AN ABSTRACT OF THE THESIS OF

Karen J. Rothermel for the degree of Master of Science in Computer Science presented on March 31, 2000.

Title: Empirical Studies of a WYSIWYT Testing Methodology

Abstract approved: ________________________________

Curtis R. Cook

Is it possible to achieve some of the benefits of formal testing within the informal programming conventions of the spreadsheet paradigm? We investigate an approach that attempts to do so via the development of a testing methodology for this paradigm. The “What You See Is What You Test” (WYSIWYT) methodology for testing spreadsheets supplements the automatic immediate visual feedback about values with automatic immediate visual feedback about “testedness”. In this thesis, we present empirical data about the methodology’s effectiveness resulting from two controlled experiments. The first experiment provided interesting but inconclusive results which spurred us to consider ways to improve the design of our experiment. We used the Cognitive Walkthrough method to evaluate and improve our design and readministered the experiment. Our results from the redesigned experiment show that the use of the methodology was associated with significant improvement in testing effectiveness and efficiency, even with no training on the theory of testing or test adequacy that the model implements.
Empirical Studies of a WYSIWYT Testing Methodology

by

Karen J. Rothermel

A Thesis

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Completed March 31, 2000
Commencement June 2000
Master of Science thesis of Karen J. Rothermel presented on March 31, 2000

APPROVED:

Redacted for Privacy

Major Professor, representing Computer Science

Redacted for Privacy

Chair of the Department of Computer Science

Redacted for Privacy

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Redacted for Privacy

Karen J. Rothermel, Author
ACKNOWLEDGMENTS

I would like to thank my major professor Dr. Curtis Cook, minor professor Dr. Margaret Burnett, and committee members Dr. Michael Quinn and Dr. Ronald Guenther. My appreciation goes to Dr. T.R.G. Green for expanding my understanding of statistics and introducing the Cognitive Walkthrough.

My thanks also go to all the members of the Forms/3 research group, especially Tim Adams, my research associate during the first experiment, Justin Schonfeld, my research associate during the second experiment, and Dusty Reichwein, who could fix just about any Forms/3 problem we managed to create.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Background</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>Spreadsheets and Spreadsheet Errors</td>
<td>4</td>
</tr>
<tr>
<td>2.2</td>
<td>The WYSIWYT Methodology</td>
<td>6</td>
</tr>
<tr>
<td>2.3</td>
<td>The Experiment's Environment</td>
<td>9</td>
</tr>
<tr>
<td>2.3.1</td>
<td>Forms/3</td>
<td>9</td>
</tr>
<tr>
<td>2.3.2</td>
<td>The WYSIWYT Methodology in Forms/3</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>The First Testing Experiment</td>
<td>14</td>
</tr>
<tr>
<td>3.1</td>
<td>Design of the Experiment</td>
<td>14</td>
</tr>
<tr>
<td>3.1.1</td>
<td>The Participants</td>
<td>15</td>
</tr>
<tr>
<td>3.1.2</td>
<td>The Tutorial</td>
<td>16</td>
</tr>
<tr>
<td>3.1.3</td>
<td>The Materials and Task</td>
<td>18</td>
</tr>
<tr>
<td>3.2</td>
<td>Results</td>
<td>20</td>
</tr>
<tr>
<td>3.2.1</td>
<td>Effectiveness</td>
<td>20</td>
</tr>
<tr>
<td>3.2.2</td>
<td>Efficiency</td>
<td>22</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Overconfidence</td>
<td>23</td>
</tr>
<tr>
<td>3.2.4</td>
<td>Training</td>
<td>24</td>
</tr>
<tr>
<td>3.3</td>
<td>Discussion</td>
<td>24</td>
</tr>
<tr>
<td>4</td>
<td>The Cognitive Walkthrough</td>
<td>27</td>
</tr>
<tr>
<td>4.1</td>
<td>Evaluating the Design of our Experiment</td>
<td>27</td>
</tr>
<tr>
<td>4.2</td>
<td>The Cognitive Walkthrough</td>
<td>28</td>
</tr>
<tr>
<td>4.3</td>
<td>Three Cognitive Walkthroughs</td>
<td>31</td>
</tr>
<tr>
<td>4.3.1</td>
<td>The First Walkthrough</td>
<td>31</td>
</tr>
<tr>
<td>4.3.2</td>
<td>The Second Walkthrough</td>
<td>34</td>
</tr>
<tr>
<td>4.3.3</td>
<td>The Third Walkthrough</td>
<td>38</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4  Discussion</td>
<td>39</td>
</tr>
<tr>
<td>Chapter 5: The Second Testing Experiment</td>
<td></td>
</tr>
<tr>
<td>5.1  Design of the Experiment</td>
<td>41</td>
</tr>
<tr>
<td>5.1.1 The Environment</td>
<td>42</td>
</tr>
<tr>
<td>5.1.2 The Participants</td>
<td>43</td>
</tr>
<tr>
<td>5.1.3 The Tutorial</td>
<td>44</td>
</tr>
<tr>
<td>5.1.4 The Materials and Task</td>
<td>46</td>
</tr>
<tr>
<td>5.2  Results</td>
<td>46</td>
</tr>
<tr>
<td>5.2.1 Effectiveness</td>
<td>47</td>
</tr>
<tr>
<td>5.2.2 Efficiency</td>
<td>48</td>
</tr>
<tr>
<td>5.2.3 Overconfidence</td>
<td>50</td>
</tr>
<tr>
<td>5.3  Discussion</td>
<td>52</td>
</tr>
<tr>
<td>Chapter 6: Conclusions</td>
<td></td>
</tr>
<tr>
<td>6.1  Threats to Validity</td>
<td>55</td>
</tr>
<tr>
<td>6.2  Future Work</td>
<td>57</td>
</tr>
<tr>
<td>6.3  Conclusion</td>
<td>57</td>
</tr>
<tr>
<td>Bibliography</td>
<td>59</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Spreadsheet for calculating student grades. The WYSIWYT feedback can be activated by clicking the “Show Test Data” button in the left margin of the spreadsheet.</td>
<td>10</td>
</tr>
<tr>
<td>2.2</td>
<td>Spreadsheet for displaying a clock face. The WYSIWYT feedback can be deactivated by clicking the “Hide Test Data” button in the left margin of the spreadsheet.</td>
<td>11</td>
</tr>
<tr>
<td>2.3</td>
<td>“Under Construction” Clock. Note that the programmer has not yet recorded any testing decisions, the minutex, minvey and minuteHand cells have red (light gray) borders, the displayed arrows are red (light gray) and all checkboxes are empty.</td>
<td>12</td>
</tr>
<tr>
<td>2.4</td>
<td>“Under Construction” Clock after a validation. Note the checkmark in minuteHand’s check box indicating that the programmer has recorded a decision. The borders for minutex, minvey and minuteHand cells are purple (dark gray) indicating that the cells are partially tested and some of the arrows are blue indicating that those particular du-associations have been tested.</td>
<td>13</td>
</tr>
<tr>
<td>3.1</td>
<td>Spreadsheet for calculating a student’s grade. A variation of the Gradebook spreadsheet, Figure 2.1, designed to calculate a single student’s final grade based on the grading rules of a hypothetical class.</td>
<td>19</td>
</tr>
<tr>
<td>3.2</td>
<td>Effectiveness data for each group and problem. The boxplots are composed of five horizontal lines at the 10th, 25th, 50th, 75th and 90th percentiles; values above the 90th or below the 10th percentiles are plotted separately.</td>
<td>21</td>
</tr>
<tr>
<td>3.3</td>
<td>Edits and Redundancies, mean values for each group and each problem. Notice the pattern of decreasing mean values over the groups for each problem in the Edits bar graph and the lack of a similar pattern in the Redundancies graph.</td>
<td>22</td>
</tr>
<tr>
<td>3.4</td>
<td>Du-adequacy means for each group and problem. Note the pattern of increase among the groups within each problem, a slight, but encouraging trend.</td>
<td>25</td>
</tr>
<tr>
<td>5.1</td>
<td>Effectiveness data for Clock and Grades.</td>
<td>47</td>
</tr>
</tbody>
</table>
5.2 Speed of coverage for Clock and Grades. In each 5-minute time interval the WYSIWYT group shows greater du-adequacy medians. The difference is significant in interval 2 and 3 for Clock and in interval 3 for Grades. ................................. 50
LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Group demographics. Analysis shows homogeneity between the groups.</td>
<td>17</td>
</tr>
<tr>
<td>4.1</td>
<td>A Cognitive Walkthrough startup sheet [27]. Italicized text represents a summary of the notes the evaluators took during the walkthrough.</td>
<td>32</td>
</tr>
<tr>
<td>4.2</td>
<td>(Partial) Cognitive Walkthrough for a step [27]. Italicized text represents a summary of the notes the evaluators took during the walkthrough, showing the accumulation of ideas resulting in the last sentence.</td>
<td>35</td>
</tr>
<tr>
<td>4.3</td>
<td>(Partial) Cognitive Walkthrough for a step. Italicized text represents a summary of the notes the evaluators took during the walkthrough, which resulted in an important testing-related user interface change to the question mark.</td>
<td>36</td>
</tr>
<tr>
<td>5.1</td>
<td>Group demographics. Analysis shows homogeneity between the groups except in the GPA category.</td>
<td>44</td>
</tr>
<tr>
<td>5.2</td>
<td>Medians for the redundancy data.</td>
<td>49</td>
</tr>
<tr>
<td>5.3</td>
<td>Overconfidence data. For each problem, the WYSIWYT group had significantly fewer overconfident participants than expected when compared to the Ad Hoc Group.</td>
<td>52</td>
</tr>
<tr>
<td>5.4</td>
<td>WYSIWYT group helpfulness ratings: percent of participants who rated the device in each of the possible helpfulness categories.</td>
<td>53</td>
</tr>
<tr>
<td>5.5</td>
<td>Participants’ opinions of the meanings of the visual devices. Percentages reflect the number of participants who chose the correct response (shown in parentheses). The questions were multiple choice and directed the participants not to guess at the answer.</td>
<td>53</td>
</tr>
<tr>
<td>5.6</td>
<td>Learning Effects: Compare the medians of Problems 1 and 2. The Ad Hoc group achieved about the same coverage on Problem 2 as on Problem 1, but the WYSIWYT group achieved significantly higher coverage on Problem 2 than on Problem 1.</td>
<td>54</td>
</tr>
</tbody>
</table>
EMPIRICAL STUDIES OF A WYSIWYT TESTING METHODOLOGY

Chapter 1

INTRODUCTION

Perhaps the most widely used programming paradigm today is the spreadsheet paradigm. Yet, almost no work has been done to help with the software engineering tasks that arise in the creation and maintenance of spreadsheets. This inattention is rather surprising given the influential role played by spreadsheets in decisions about a variety of real-world issues, such as budgets, student grades, and taxes.

In spite of the perceived simplicity of the spreadsheet paradigm, research shows that spreadsheets often contain faults. For example, in an early study, 44% of “finished” spreadsheets still contained faults [3]. A recent survey of other such studies reported faults in as many as 77% of spreadsheets [24, 25]. Of perhaps even greater concern, this survey also includes audits of “production” spreadsheets, those actually in use for day-to-day decision making, and as many as 26% of these spreadsheets contained faults.

Contributing to the problem of reliability is the unwarranted confidence that spreadsheet programmers seem to have in the correctness of their spreadsheets [3, 44]. A possible cause of this overconfidence may be related to findings of Gilmore and of Svendsen, who showed that too much responsiveness and
feedback, as featured in the immediate visual feedback of *values* in spreadsheet languages, can actually interfere with people's problem-solving ability in solving puzzles [12, 39], a task with much in common with programming.

To address the reliability problem associated with spreadsheets, we are investigating the possibility of bringing some of the benefits of formal testing to the informal, highly interactive, declarative spreadsheet paradigm. Our theory is that by providing feedback about the "testedness" of a spreadsheet, as opposed to just the values, we can cause programmers to have less overconfidence about the correctness of their spreadsheets. This testing feedback may motivate programmers to test their spreadsheets more thoroughly and provide guidance that helps them test more efficiently. This could lead to better testing even in this informal programming domain and to a reduction in faults before a spreadsheet is relied upon in decision making.

Rothermel, et al. have developed a "What You See Is What You Test" (WYSIWYT) methodology for testing spreadsheets [34]. The methodology is designed to accommodate the declarative evaluation model of spreadsheet formulas, the incremental style of development, and the immediate visual feedback expected of spreadsheet languages. The methodology is designed for use by the wide range of programmers using spreadsheet languages, taking into account the lack of formal software engineering background of many of these programmers.

Given this methodology, determining whether it can be used in a way that brings any benefit to programmers requires answers to three questions:

First, is the methodology efficient enough to coexist with the immediate cell redisplay expected after each formula edit? In [34] Rothermel, et al. showed that most of their algorithms can be implemented in ways that add only $O(1)$ to the existing cost of maintaining the interactive environment.
Second, will the methodology uncover faults in programs? The methodology guides programmers (though they need not be aware of it) in meeting a dataflow test adequacy criterion, which will be described in Chapter 2. Rothermel et al. empirically studied the fault detection characteristics of test suites that meet this criterion. Their results suggest that such test suites can provide fault detection rates for spreadsheets at a significantly higher rate than equivalently sized randomly generated test suites [34, 31].

Third, will programmers who use the methodology be less overconfident and be more effective, more efficient testers than programmers who do not use the methodology? To investigate this third question, we have begun a series of empirical studies with human participants. The first of these studies is the subject of this thesis.

In Chapter 2, we will review the literature on spreadsheet testing, errors and over-confidence, followed by a discussion of the testing methodology we employed in our study. We will describe our first experiment in Chapter 3, and in Chapter 4 explain the Cognitive Walkthrough technique we used to help redesign the experiment. Chapter 5 describes the redesigned experiment and its results. In Chapter 6, we conclude this thesis and discuss some of the questions remaining to be answered by future empirical studies.
Chapter 2

BACKGROUND

In this chapter we will present background material on the occurrence of errors in spreadsheets and the overconfidence of spreadsheet programmers. We then review the WYSIWYT methodology which proposes to help reduce those errors and the associated overconfidence. Finally we describe the Forms/3 spreadsheet language in which the methodology was prototyped for our experiments. Much of the information in this chapter can also be found in [35].

2.1 Spreadsheets and Spreadsheet Errors

Spreadsheet languages differ from most other commonly used programming languages in that they provide a declarative approach to programming, characterized by a dependence-driven, direct-manipulation working model [1]. Users of spreadsheet languages create cells and define formulas for those cells. These formulas reference values contained in other cells and use them in calculations. As soon as a cell’s formula is defined, the underlying evaluation engine automatically calculates the cell’s value and the values of affected cells (at least those that are visible), and immediately displays new results. Spreadsheet languages are used for computational tasks ranging from simple “scratchpad” applications developed by single users to large-scale, complex systems developed by multiple users [24].
The spreadsheet paradigm is considered “easy to understand and use”, as evidenced by the millions of non-programmers who create spreadsheets. Unfortunately, there are problems that plague spreadsheets and spreadsheet programmers: errors, over-confidence about accuracy, poor design, and lack of documentation [3, 8, 24, 30, 40]. The following are just a few of the more sensational stories of the effects of spreadsheet errors [26]:

- A Florida construction company used @sum to sum a range. The range was not changed when an item was added, so the added item was not summed. This caused the company to underbid the project by a quarter of a million dollars. The company sued Lotus.

- Eric Klasson, a Houston consultant with Price Waterhouse audited 4 spreadsheet model. He found 128 errors covering 120 line items. Some formulas were applied differently to two subsidiaries. The spreadsheets had already been in use for months.

- A Dallas Oil and Gas company’s spreadsheet error resulted in millions of dollars being lost. Several executives were fired.

The reliability problem of spreadsheets has been well documented over the years: Brown and Gould [3] conducted a study of 9 experienced spreadsheet users in which each user created 3 spreadsheets; 44% of the spreadsheets created had at least one error and every participant introduced at least one error. But, the participants were “quite confident” about the accuracy of their spreadsheets, including the spreadsheets that contained errors. In a recent survey of the literature on spreadsheet errors, Panko [24] found the following: in 4 field audits of operational spreadsheets, errors were found in 10.7%, 21%, 25% and
26%, respectively, of spreadsheets audited; in 11 experiments in which participants created spreadsheets, errors were found in an average of 60.8% (median 57.5%) of the spreadsheets; in 4 experiments in which the participants inspected spreadsheets for errors, the participants missed 33%, 54%, 55%, and 81%, respectively, of the errors.

Panko also investigated the associated problem of overconfidence of spreadsheet developers in his survey on spreadsheet errors [24]. He cites the findings of Brown and Gould noted above, findings from one of the field audits in which 21% of the spreadsheets contained errors and yet the developers were “extremely confident in the accuracy of their spreadsheets”, as well as findings from several other experiments in which the participants had high confidence in accuracy despite the presence of errors in their spreadsheets. The issues of design and documentation are also well documented in the literature [30], but since our focus is on errors and overconfidence, we will not pursue this issue.

Our basic hypothesis is that by providing feedback about the testedness of a spreadsheet, rather than just values, we can cause spreadsheet users and developers to have less overconfidence about the accuracy of the spreadsheet and thereby cause them to test their spreadsheets more thoroughly and reduce the number of errors. This thesis will report on two experiments that investigated this hypothesis.

2.2 The WYSIWYT Methodology

The literature on program testing primarily addresses the testing of imperative programs (e.g. [9, 11, 14, 20, 28, 42]), with a few attempts to address the testing of functional and logic programs (e.g. [2, 16, 19, 23]). However, there are three
major differences between the spreadsheet and imperative language paradigms that directly affect the development of a testing methodology for spreadsheets. First, evaluation of spreadsheets is driven by data dependencies between cells, and spreadsheets contain explicit control flow only within cell formulas. This dependence-driven evaluation model allows evaluation engines for spreadsheets flexibility in the scheduling algorithms and optimization devices they employ to perform computations. A methodology for testing spreadsheets must be compatible with this flexibility and not rely upon any particular evaluation order. Second, spreadsheets are developed incrementally and there is an immediate visual response after each addition of or modification of a formula. A testing methodology for spreadsheets must be flexible enough to operate upon partially-completed programs and efficient enough to support responsiveness. Third, and most critical, whereas most imperative programs are developed by professional programmers, spreadsheets are developed by a wide variety of users, many of whom have no training in formal testing principles. Rothermel et al.’s “What You See Is What You Test” (WYSIWYT) methodology for testing spreadsheets takes these factors into account. We briefly describe the foundations of that methodology here; a detailed presentation can be found in [33, 34].

Spreadsheet programmers are not likely to write specifications for their spreadsheets; thus, the WYSIWYT methodology relies, behind the scenes, on code-based test adequacy criteria. Code-based test adequacy criteria provide help in selecting test data and in deciding whether a program has been tested “enough” by relating testing effort to coverage of code components such as program statements or flow of control branches. Such criteria have been well researched for imperative languages (e.g. [11, 20, 28]), and several empirical studies (e.g. [10, 14, 42]) have demonstrated their usefulness.
The WYSIWYT methodology incorporates a test adequacy criterion adapted from the *output-influencing-all-du-pair* dataflow adequacy criterion defined originally for imperative programs [9]. This criterion, called *du-adequacy* for brevity, focuses on the definition-use associations (du-associations) in a spreadsheet, where a du-association links an expression in a cell formula that defines a cell's value with expressions in other cell formulas that use (reference) the defined cell. The criterion requires that each executable du-association in the spreadsheet be exercised by test data in such a way that the du-association contributes, directly or indirectly, to the display of a value that is subsequently pronounced correct (validated) by the programmer.

It is not always possible to exercise all du-associations; those that cannot be exercised are called *nonexecutable*. Determining whether a du-association is executable is provably impossible in general and frequently infeasible in practice [11, 42]. Data flow test adequacy criteria typically require that test data exercise (cover) only executable du-associations. In this respect, the WYSIWYT criterion is no exception. In our experience with spreadsheets, however, most of the nonexecutable du-associations we have encountered have involved direct contradictions between conditions that are relatively easy for persons capable of creating those spreadsheets to identify.

The appropriateness of the du-adequacy criterion for spreadsheets stems from relating test coverage to interactions between definitions and uses of cells by requiring these interactions to be exercised; this requirement is important since such interactions are a primary source of faults in spreadsheets. Moreover, the criterion does not enforce any expectation of a particular cell execution order: the du-associations in a spreadsheet are the same regardless of the order in which the evaluation engine executes cells. Further, by linking test coverage to
cell validation, the criterion avoids problems in which du-associations influencing only hidden or off-screen values are considered exercised simply by applying test inputs. A du-association is exercised only when it participates in producing a visible result that is validated by the programmer. Finally, the criterion facilitates the incremental testing of spreadsheets, allowing a programmer to focus on a small subset of the potentially large set of cells in a spreadsheet.

2.3 The Experiment’s Environment

We prototyped the WYSIWYT testing methodology in the research language Forms/3 [4], one of many spreadsheet language research systems (e.g. [4, 6, 17, 21, 38, 41]). This choice is motivated primarily by the fact that we have access to the implementation of Forms/3, and thus, we can implement and experiment with various testing technologies within that environment. Additionally, by working with Forms/3, we can investigate language features common to commercial spreadsheet languages as well as advanced language features found in research spreadsheet languages.

2.3.1 Forms/3

As in other spreadsheet languages, a Forms/3 spreadsheet is a collection of cells; each cell’s value is defined by the cell’s formula. The programmer enters a cell’s formula and receives immediate visual feedback as to the cell’s value. In Forms/3, ordinary formulas can be used for both numeric and graphical computations. Figure 2.1 shows a gradebook spreadsheet that computes averages for each student and for the class. Figure 2.2 shows a spreadsheet that translates numeric input values for hour and minute into a graphical clock face with hour
FIGURE 2.1: Spreadsheet for calculating student grades. The WYSIWYT feedback can be activated by clicking the “Show Test Data” button in the left margin of the spreadsheet.

and minute hands. In Forms/3, a programmer can freely reposition a cell and can choose to display or hide the cell’s formula. In the Gradebook spreadsheet, all formulas are hidden while in the Clock spreadsheet, the programmer has chosen to display all of the non-constant formulas.

2.3.2 The WYSIWYT Methodology in Forms/3

Forms/3 incorporates the WYSIWYT methodology by giving immediate visual feedback about the testedness of a spreadsheet. Suppose that a programmer has begun creating the graphical clock, shown in Figure 2.2, and has entered a few of the cells and formulas. During this process, the underlying evaluation engine has not only been displaying cell values, but has also been calculating
the du-associations that come into existence as new formulas are created, and tracking the du-associations that influence calculations. Using this information, visual devices keep the programmer continually informed as to testedness status. To draw attention to untested sections, cell border colors reflect the extent to which a cell has been tested\(^1\), a red (light gray in this black and white paper) cell border means the cell is untested, a blue (black) border means the cell is fully tested, purple (dark gray) means partially tested. Information about testedness at a finer granularity is available via arrows depicting definition-use associations

\(^1\) Formally, du-adequacy does not define test adequacy at the granularity of cells. When we refer to a cell being tested it is shorthand for saying that all the du-associations whose uses are in that cell have contributed to values that the programmer pronounced correct.
FIGURE 2.3: "Under Construction" Clock. Note that the programmer has not yet recorded any testing decisions, the \textit{minute}, \textit{minutey} and \textit{minuteHand} cells have red (light gray) borders, the displayed arrows are red (light gray) and all checkboxes are empty.

(which in user terms are simply interactions between portions of cell formulas) that the programmer may display or hide. Following the same color scheme as for the cell borders, a red arrow indicates that a du-association has not been tested and a blue arrow indicated that a du-association has been tested.

For example, in Figure 2.3, the spreadsheet programmer has created a portion of the Clock spreadsheet and has displayed arrows for the cells. Suppose the she now decides that cell \textit{minuteHand}'s displayed graphical value is correct, given the \textit{minute} cell's current value. To record the decision that the value is correct, she clicks on the check box in the upper right corner of \textit{minuteHand} cell. The system responds with immediate visual feedback as to the new testedness of each visible cell and arrow, as shown in Figure 2.4. The underlying validation algorithm is given in [34]; the overall notion is that it recurses back through the du-associations that affect, directly or indirectly, the currently computed value of \textit{minuteHand}, and marks them tested (covered).
FIGURE 2.4: "Under Construction" Clock after a validation. Note the check-mark in \( \text{minuteHand} \)'s check box indicating that the programmer has recorded a decision. The borders for \( \text{minutex}, \text{minutey} \) and \( \text{minuteHand} \) cells are purple (dark gray) indicating that the cells are partially tested and some of the arrows are blue indicating that those particular du-associations have been tested.

The methodology also accounts for the retesting that may be required whenever the spreadsheet programmer edits a non-input formula. This aspect was not included in the experiment, so we do not discuss it here.
Chapter 3

THE FIRST TESTING EXPERIMENT

In this chapter we will present material on our first testing experiment using the WYSIWYT methodology. We will describe the design of the experiment: the participants, the tutorial, the task and the materials. We then report the results of the experiment. Much of the information in this chapter can also be found in [35].

3.1 Design of the Experiment

The objectives of our study were to investigate the following research questions:

RQ 1: Do programmers who use the WYSIWYT testing methodology create test suites that are more effective in terms of du-adequacy than programmers who use an ad hoc approach?

RQ 2: Do programmers who use the WYSIWYT testing methodology create test suites more efficiently than programmers who use an ad hoc approach?

RQ 3: Are programmers who use the WYSIWYT testing methodology less overconfident about the quality of their testing than programmers who use an ad hoc approach?

These questions were translated directly into hypotheses. We also took care that the design of our experiment would provide insight into the following question:
RQ 4: Is training in the underlying test adequacy criterion and its relationship to the visual devices needed in order for spreadsheet programmers to gain testing effectiveness or efficiency from using the WYSIWYT methodology?

To investigate these questions, we conducted a controlled laboratory experiment in which the participants tested two spreadsheets. The following sections will describe details of the experiment and our analysis of the data collected during the experiment.

3.1.1 The Participants

We have previously pointed out that the spreadsheet paradigm serves a range of audiences, from end users to professional programmers. In this experiment we used advanced computer science (CS) students as our participants. CS students are one segment of the population served by spreadsheets, and are an especially interesting group of participants for exploration of our research questions because of their experience with testing. That is, CS students are more experienced in testing than are most end users; therefore, because of their experience, they might already be efficient and effective at testing without much room for the improvement that could be otherwise gained from using the methodology. Thus, if CS students show a significant improvement, the method shows promise for spreadsheet programmers with less experience in testing.

Sixty-seven students participated in the experiment, drawn from students enrolled in three computer science courses: a junior-level, a senior-level, and a graduate-level course. The participants were randomly divided into three groups, subject to the constraint that each group contained approximately equal numbers of students from each course. One group, the Ad Hoc group, did not
have access to the WYSIWYT methodology described in Section 2.3.2; this group represents programmers who test in an ad hoc fashion. The other two groups, the WYSIWYT-No-Training group and the WYSIWYT-With-Training group, did have access to the WYSIWYT methodology. We refer to participants in the combination of the latter two groups as WYSIWYT participants.

Of the 67 participants, the Ad Hoc group, WYSIWYT-No-Training group, and WYSIWYT-With-Training group contained 21, 24 and 22 participants, respectively. The difference in group sizes was due to a few students failing to arrive for their appointments and to computer system problems. The group sizes were subsequently reduced when we decided to omit data for participants whose measurable testing activity level was zero (as revealed by transcripts of the sessions), whose computer crashed during the experiment, or who inadvertently corrupted their data in other ways. The removal of these participants reduced the number of participants in the Ad Hoc and WYSIWYT-No-Training and WYSIWYT-With-Training groups to 17, 23 and 21, respectively, for a total of 61 participants.

To ascertain whether the participants in the three groups had reasonably similar backgrounds, we administered a background questionnaire to each participant and analyzed the data. Results, summarized in Table 3.1, showed homogeneity between the groups.

### 3.1.2 The Tutorial

The experiment was conducted in a computer lab with participants seated one per workstation, using Forms/3. It began with a 20-minute tutorial of Forms/3, during which each student actively worked with the example spreadsheets on
their workstations following instructions given by the lecturer. Throughout the tutorial the participants had access to a handout containing a quick reference of the Forms/3 features they were being taught. They could make notes on these handouts which remained available to them throughout the experiment.

The first part of the tutorial introduced the participants to environmental features (e.g., how to edit cells, display formulas, etc.) that would eventually be needed in testing the spreadsheets. All participants received the same instructions in the first part of the tutorial.

The second part of the tutorial described how to test, record testing decisions and interpret the testing feedback. Testing was described as trying various input values and recording decisions made about the correctness of values in the other cells. All participants were instructed to use the check boxes to record decisions about correct cell values and to record information about incorrect cell values in a special cell named *OutputErrors*. As a result of recording decisions about correct values, the WYSIWYTYT participants received the methodology’s feedback of checkmarks and cell and arrow colors as described in Section 2.3.2, while the Ad Hoc participants’ feedback was that the background of the spreadsheet blinked several times to indicate that the decision was recorded.

TABLE 3.1: Group demographics. Analysis shows homogeneity between the groups.
To this point, all groups received the same general tutorial on how to use Forms/3: after this, both WYSIWYT groups received instruction on how to use the WYSIWYT feedback. The WYSIWYT-With-Training group's tutorial was also supplemented by a brief explanation of du-adequacy and the relationship of testing feedback to this concept. However, regardless of which group a participant was in, the total training time was identical; participants not receiving training on the testing feedback and methodology were given more time to practice using Forms/3.

3.1.3 The Materials and Task

Following the tutorial, the participants were asked to test two different spreadsheets, Clock and Grades. The Clock spreadsheet, shown in Figure 2.2, converts numeric hour and minute values into a graphical clock face with hour and minute hands. The Grades spreadsheet, Figure 3.1, is a single-student variant on the one presented in Figure 2.1. It calculates the final letter grade for one student based on the grading rules of a hypothetical class. The differences between Clock and Grades are that (1) they represent different problem domains (graphic and numeric), (2) the formulas for Clock are relatively difficult to understand, whereas those of Grades are relatively easy to understand (lending themselves to straightforward reasoning by examining the code), and (3) the oracle problem—determining whether the displayed “final” output is correct—may be easier for Clock than for Grades. We designed each spreadsheet to be fault-free\(^1\) and took care to reduce the number of nonexecutable du-associations;

\(^1\) Though we designed the spreadsheets to be fault-free, the Grades spreadsheet contained an error: ExtraCredit was not checked for inappropriate values in the \(\text{ErrorsExist?}\) cell.
FIGURE 3.1: Spreadsheet for calculating a student’s grade. A variation of the Gradebook spreadsheet, Figure 2.1, designed to calculate a single student’s final grade based on the grading rules of a hypothetical class.

however, these facts were not revealed to the participants, who were informed instead that the spreadsheets “might or might not” contain faults and no mention of nonexecutable du-associations was made.

Both spreadsheets were tested by all participants. The participants had 15 minutes per spreadsheet to perform their testing. The experiment was counterbalanced with respect to problem type in that participants in each group worked both problems, but half the participants in a group tested the Grades spreadsheet first, whereas the other half tested the Clock spreadsheet first. At the beginning of each testing session we instructed the subjects to read a description of the spreadsheet they were about to test, the descriptions included details of what the spreadsheets was to accomplish and indicated which cells
were the “input” cells. The participants were able to enter different values in the input cells, but were prohibited by the environment from changing non-input cells' formulas, in order to maintain the integrity of the spreadsheets throughout the study.

3.2 Results

We collected the following data: (1) the actions (edits, decisions, etc) of each participant throughout the experiment’s sessions were automatically recorded in transcript files, (2) the test suites created by the participants’ actions, derived from the information in the transcript files, and (3) participants’ answers to three questionnaires. The first questionnaire collected information about participants’ backgrounds, as noted in Section 3.1.1. A questionnaire at the end of each testing session collected subjective information about the participants’ perceived testing effectiveness. In the third questionnaire, the WYSIWYT participants were also questioned about their understanding and use of the WYSIWYT visual features.

3.2.1 Effectiveness

Our first research question was to determine if the two WYSIWYT groups created more effective test suites than the Ad Hoc group. From their test suites we determined the percent of executable du-associations covered (du-adequacy percent). An overview of the effectiveness data for each group and problem is presented in the form of boxplots in Figure 3.2.

We examined the effectiveness data to see whether it satisfied the requirements for using normal-theory analyses such as t-tests and analysis of variance.
FIGURE 3.2: Effectiveness data for each group and problem. The boxplots are composed of five horizontal lines at the 10th, 25th, 50th, 75th and 90th percentiles; values above the 90th or below the 10th percentiles are plotted separately.

There were some indications of skew, but the analysis of variance is robust against moderate skew; because our data sets are small, we also used more conservative non-parametric alternatives where possible, which gave identical patterns of results.

We analyzed effectiveness using analysis of variance with two factors, Environment (Ad Hoc, WYSIWYT-No-Training, WYSIWYT-With-Training) and Problem (Clock or Grades). Each subject experienced only one environment and attempted both problems, so the Environment factor was treated as independent groups and the Problem factor was treated as having repeated measures.

The results of the ANOVA were inconclusive, we were unable to reject the null hypothesis of no difference in du-coverage between the data sets, $p=0.5821$. Using both multiple comparison tests such as Bonferroni/Dunn and non-parametric tests such as Mann-Whitney we considered the various combinations using two of the three groups but we were unable to reject the null hypothesis for any of the combinations.
FIGURE 3.3: Edits and Redundancies, mean values for each group and each problem. Notice the pattern of decreasing mean values over the groups for each problem in the Edits bar graph and the lack of a similar pattern in the Redundancies graph.

3.2.2 Efficiency

To address our second research question, do the two WYSIWYT groups create test suites more efficiently than the Ad Hoc group, we considered two measures: the number of edits made by the participants and the number of redundancies in their test suites. The mean number of edits for each group can be seen in Figure 3.3. An analysis of variance, $p = 0.0093$, shows there is a significant difference between the number of edits made by the groups. To determine which combinations of the groups exhibit a significant difference in the number of edits, we used both multiple comparison tests such as Bonferroni/Dunn and non-parametric tests such as Mann-Whitney. We found a significant difference in the number of edits between the Ad Hoc and WYSIWYT-With-Training groups, Bonferroni/Dunn $p = 0.0023$; verifying this result with Mann-Whitney, Clock $p = 0.005$ and Grades $p = 0.032$.

The number of redundancies in a test suite was calculated by summing over all du-associations one less than the number of times the du-association was covered. This was perhaps an overly simplistic view of redundancy since two
different tests may cover some but not all of the same du-pairs. This measure also implies that multiple test cases exercising the same du-association do not increase testing effectiveness which is again overly simplistic [32]; however, this assumption is common to the use of structural coverage criteria.

The results of our analysis of the redundancy data were inconclusive. Using analysis of variance we were unable to reject the null hypothesis of no difference in redundancy between the data sets, \( p = 0.914 \) Using both multiple comparison tests such as Bonferroni/Dunn and non-parametric tests such as Mann-Whitney we considered the various combinations using two of the three groups but we were unable to reject the null hypothesis for any of the combinations.

At this point in our investigation we can make one important statement based on the results of our analysis of effectiveness and efficiency: The WYSIWYT-With-Training group made significantly fewer edits than the Ad Hoc group while achieving at least the same du-coverage; they were more efficient without loss of effectiveness.

### 3.2.3 Overconfidence

The third research question dealt with overconfidence: are the two WYSIWYT groups less overconfident about the quality of their testing than the Ad Hoc group? To investigate this question we compared the participants' du-coverage to their self-reported evaluation. The participants rated themselves as having tested each spreadsheet “Very Thoroughly”, “Adequately”, or “Poorly”. Since the participants are all students and were told to “do their best” during the experiment, we mapped the given categories to du-coverage percentages as follows: Very Thoroughly = 100 - 90\%, Adequately = 80 - 89\%, Poorly = 0 - 79\%.
Then comparing their self-evaluation to their actual du-coverage, we categorized each participant as overconfident or not overconfident for each spreadsheet. Using Fisher’s Exact Test the results were inconclusive. With \( p = 0.783 \), we were unable to reject the null hypothesis of no difference in numbers of participants overconfident among the groups.

### 3.2.4 Training

The fourth research question explores the need for training in the methodology beyond a very basic level: is training in the underlying test adequacy criterion and its relationship to the visual devices needed in order for users to gain testing effectiveness or efficiency? If we review each of the previous analyses, we find that there were no significant differences between the WYSIWYT-No-Training and WYSIWYT-With-Training groups. We consider this to be an important result since eventually we want to work with end users who may have little or no background in software engineering. If extensive training in testing is needed before the methodology is at all effective, it will be of little use to end users.

### 3.3 Discussion

As we noted in Section 3.2.2, the WYSIWYT-With-Training group was more efficient than the Ad Hoc group, without loss of effectiveness. In Section 3.2.4 we concluded that there were no significant differences between the WYSIWYT-No-Training and the WYSIWYT-With-Training groups. These results and the points noted in the following paragraphs led us to believe the WYSIWYT methodology has potential to benefit spreadsheet programmers and needs further investigation.
In our consideration of the WYSIWYT methodology's potential, we were encouraged by some interesting trends in the data for effectiveness and redundancy. Comparing the du-adequacy percent means for each group and problem using the bar graph of Figure 3.4, we see a pattern of increasing means from the Ad Hoc group to the WYSIWYT-No-Training group to the WYSIWYT-With-Training group for both the Clock and Grades spreadsheets. The differences shown in the graph are small but the trend is encouraging. The boxplots of Figure 3.2 show a pattern of decreasing standard deviations from the Ad Hoc group to the WYSIWYT-No-Training group to the WYSIWYT-With-Training group for the Clock spreadsheet and a decrease in standard deviation from the Ad Hoc to the WYSIWYT-With-Training group for the Grades spreadsheet.

The redundancy data, shown in Figure 3.3, gave us mixed results; to simplify the following discussion we will consider the two WYSIWYT groups combined. The ANOVA results show an interaction between problem type and redundancy (p= 0.0188), that is, the pattern of redundancy was different for Clock and Grades. For the Clock spreadsheet, redundancy was higher for the Ad Hoc group than for the WYSIWYT groups, but for the Grades spreadsheet the opposite was true. To try to make sense of this data, we looked at some of
the factors that may have affected it: Participants reached higher du-adequacy percents on the Clock spreadsheet than the Grades spreadsheet (ANOVA, \( p < 0.0001 \)) which leads us to conjecture that the Grades spreadsheet is "harder" to test in some way; the Ad Hoc group created significantly more tests for the Clock spreadsheet than for the Grades spreadsheet (Wilcoxon, \( p = 0.0319 \)) while the WYSIWYT groups created about the same number of tests for each spreadsheet. We conjecture that the WYSIWYT participants worked "harder" in some way on the "hard" problem of testing the Grades spreadsheet; their greater effort led to more redundancy. With a similar logic, we conjecture that the WYSIWYT participants worked "smarter" on the "easy" problem of testing the Clock spreadsheet; their "smarter" work led to less redundancy.

Final points of encouragement for us were the differences between the first and second problem. Recall that the experiment was counterbalanced for problem type, half the participants in a group tested the Clock spreadsheet first while the other half tested the Grades spreadsheet. Again, to simplify the discussion, we consider the two WYSIWYT groups combined. The Ad Hoc group achieved about the same du-adequacy on problem 2 as on problem 1 (Wilcoxon, \( p = 0.46 \)), while the WYSIWYT groups achieved significantly higher du-adequacy on problem 2 than on problem 1 (Wilcoxon, \( p = 0.0423 \)). This "learning effect" is important; after a brief tutorial on the methodology, the WYSIWYT participants were able to continue learning on their own how to use the methodology more effectively.

Thus, although some points of our investigation had inconclusive results, the WYSIWYT methodology appeared promising and we considered ways to continue and improve our investigation.
Chapter 4
THE COGNITIVE WALKTHROUGH

In this chapter we present the Cognitive Walkthrough as a means of reviewing and improving our experiment’s design. We consider the appropriateness of the cognitive walkthrough method, then apply the method several times as we make changes to our design. Much of this chapter was originally written by T.R.G. Green, Andy Ko, and Margaret Burnett, and can be found in [13] with additional details on the Cognitive Walkthrough method and our use of it.

4.1 Evaluating the Design of our Experiment

Our experiment produced some interesting, but for the most part, not statistically significant results. However, we began to realize in performing the investigation that there may have been issues with the experiment’s design which translated to uncontrolled variables and clouded our results. Due to the possibility of such variables, we could not be sure whether the results could truly be interpreted as evaluations of the WYSIWYG methodology itself.

To address questions about the experiment’s design, on the advice of a Human-Computer Interaction (HCI) expert¹, T.R.G. Green, we decided to use

¹ The designers and administrators of this experiment are all computer science researchers without expertise in psychology or human-computer interaction though some members of the team are experienced in experiment design.
the Cognitive Walkthrough [18, 27, 29, 37] to identify and remove as many un-controlled variables as possible. In this chapter we report on our use of the Cognitive Walkthrough, the changes it led us to make, and try to answer the following questions:

1. The Cognitive Walkthrough has been used almost exclusively for evaluation of interfaces. Can it be applied to the evaluation of a design for an experiment?

2. The Cognitive Walkthrough method was created under the assumption that an HCI specialist would be part of the team conducting the walkthrough. Can it be used successfully without an HCI specialist?

4.2 The Cognitive Walkthrough

Although HCI evaluation methods have been developed to evaluate the usability of interfaces, no studies, to our knowledge, have addressed our Question 1 above, i.e. whether the Cognitive Walkthrough method, or indeed any other usability evaluation method, can assist in the design of an experiment.²

Why then base evaluating an experiment’s design on the Cognitive Walkthrough rather than on some other evaluation method? One answer relates to the context of use, particularly from the perspective of an experiment participant. Most evaluation methods focus on a mixed bag of questions, as in Heuristic Evaluation [22], or focus solely on how easily each step is carried out—without considering how the user finds the appropriate step—as in the Keystroke Level Model [5]. Thus, Heuristic Evaluation is effective at catching

²Details of the Cognitive Walkthrough and other HCI evaluation methods were provided by T.R.G. Green an HCI expert consulting with our team.
inconsistencies of wording, for instance, and the Keystroke Level Model is good for predicting how fast a user can perform a routine task, but neither of these is a good fit for programming tasks, because those tasks are non-routine and require an element of reasoning. In contrast to these methods, the Cognitive Walkthrough focuses on how a user knows, or discovers, what action is appropriate and whether an action resulted in progress towards the goal. It is therefore good at catching cases where the “user” is likely to say “Help! I don’t know what to do next”!

A second answer is that the Cognitive Walkthrough shows potential for being usable by computer scientists without the assistance of an HCI or cognitive science expert. On one hand, shortly after the tool’s introduction, Wharton, Bradford, and Franzke [43], after evaluating three user interfaces, identified a few important weaknesses of the process, and claimed “it will be difficult to eliminate the need for a cognitive science background both to make sense and to take full advantage of the technique.” However, John and Packer [15] reported a case study in which “the Cognitive Walkthrough evaluation technique [was] learnable and usable for a computer designer with little psychological or HCI training.”

The HCI or cognitive science expertise prerequisite then, has been demonstrated to be open to question. Other candidate evaluation methodologies, in contrast, explicitly require the cooperation of such an expert. This issue is of interest because such expertise is often not available to a research team in the areas of programming languages and software engineering.

The Cognitive Walkthrough inspects the steps the user is expected to perform in carrying out a task. It rests on an acceptable cognitive model of user activity during the phase of exploratory learning. That model describes four phases of activity:
• The user sets a goal to be accomplished;
• The user searches the interface for currently available actions;
• The user selects an action that seems likely to make progress toward the goal;
• The user performs the action and checks to see whether the feedback indicates that progress is being made towards the goal.

Before a walkthrough can take place, the evaluation team must prepare demographics of the expected users and an estimate of their prior knowledge, a fairly detailed description of one or more tasks for which the system is capable, and a list of action-steps comprising the optimal sequence of execution. Once this preliminary setup is complete, the actual walkthrough follows. At each step of the execution sequence, the team follows a printed form to answer preset questions covering four points relevant to the four phases of activity:

• Will the user form the right goal?
• Is an appropriate action readily available?
• Will the user find that action?
• Will the user know that progress has been made?

As the team “walks through” each step in the execution sequence, they note where the user may not take the correct action and, using the form, deduce why. At the end of the walkthrough, the team should have identified a set of problems associated with the system. In our adaptation of this method, we instead used the walkthrough to identify a set of problems associated with our experiment, including the combination of a system, a set of tasks, a tutorial, and other experiment materials.
4.3 Three Cognitive Walkthroughs

Our team did not have previous experience using the Cognitive Walkthrough method. T.R.G. Green, an HCI expert consulting for our team, taught us the method using a brief tutorial (about 90 minutes), which introduced the ideas behind the Cognitive Walkthrough, the forms used during a walkthrough and presented example walkthroughs.

4.3.1 The First Walkthrough

The HCI expert monitored the first walkthrough, but did not participate in it. His monitoring activities were (1) to answer questions about the Cognitive Walkthrough method, and (2) to observe whether the team seemed to be able to use the Cognitive Walkthrough method effectively.

First, the team performed the setup process required by the Cognitive Walkthrough forms; see Table 4.1. The form requires a description of the task, the list of action-steps for the optimal sequence of execution of the task, a description of the anticipated users, and the users initial goals. The task description was the same one the participants would receive during the experiment: test a spreadsheet. We assumed the user’s initial goal would coincide with the task. Refer to Table 4.1 for the list of action-steps. We chose end users as our anticipated user population: although our earlier experiment used computer science students, we were looking ahead to future work with end users who would not be as tolerant of imperfections as are computer science students. After completing the startup sheet, we began the main part of the walkthrough.

During the walkthrough we discovered many design issues of concern. A portion of the walkthrough is shown in Table 4.2 and the following is a list
Cognitive Walkthrough Form: Start up sheet.

Experiment: WYSIWYGT experiment

Task: Test the Grades spreadsheet by validating the outputs corresponding to various sets of input value

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 to work.

The task is to test the spreadsheet (this is what they’ll be told). The system will be in a state such that someone could immediately start testing.

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

The optimal sequence of actions: (1) Change an assignment grade to a different value by (a) double-clicking on the formula tab; (b) changing the window focus; (c) entering a value; (d) accepting; (2) Validate output by checking the final output box (hopefully the user will choose the “final grade” cell); (3) Repeat for different inputs.

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.

People who have experience with spreadsheet basics, but limited experience inventing spreadsheet formulas. They should have basic algebra skills, and will have gone through the tutorial, but will not have had other Forms/3 training.

User’s Initial Goals. List the goals the user is likely to form when starting the task. If there are likely goal structures list them.

We think it’s going to be “test the spreadsheet,” rather than something more concrete like “change an input value.”

TABLE 4.1: A Cognitive Walkthrough startup sheet [27]. Italicized text represents a summary of the notes the evaluators took during the walkthrough.

of design problems and potential solutions that we discussed. Some of those problems and solutions had been considered previous to the walkthrough and are marked with a * in the list.
1. Problem Design or Forms/3 Language Design Problems\(^3\)
   - nested conditionals in “if” expressions
     Solution: alter problems to eliminate or reduce
   - nested parenthesis
     Solution: rewrite language parser or alter problem formulas
   - indentation
     Solution: indent formulas in a consistent and readable way
   - min/max operators
     Solution: implement min/max operators in language or alter problems

2. Tutorial Design Problems:
   - too long
     Solution: eliminate unnecessary details and combine some explanations
     Solution: eliminate “With-Training” group and associated details *
   - participants may not acquire all necessary skills
     Solution: provide more time to practice skills at the end of the tutorial

3. Testing-Related User Interface Problems:
   - no specific long term progress feedback
     Solution: implement a spreadsheet testedness indicator
   - no way to “undo” a validation *
     Solution: implement an undo feature
   - no feedback as to what inputs to try next *
     Solution: implement an alternate version of cell relation arrows

---

\(^3\) Some of the Spreadsheet Design or Forms/3 Language Design issues that were found can be seen in the Grades spreadsheet in Figure 3.1. Note that each issue introduces distractions:
- cell LetterGrade on the right exhibits several nested conditionals, requiring participants to focus on the flow of logic;
- the ErrorsExist? cell in the bottom-left contains extensively nested parentheses, requiring participants to expend extra energy on parsing;
- several of the “if” expressions are indented in different ways;
- cells min, min1, and min2, would normally be handled by a single minimum operator.
4. Other User Interface Problems:

- formula display is difficult *
  *Solution: implement a new formula display method*

- formula editing *
  - mechanism has no useful label
  - edit window can “get lost” behind other windows
  - contains an “and-then” structure
  *Solution: implement a new formula editor*

- fractional results are shown in ratio format
  *Solution: change output format to decimal style*

- slow response time
  *Solution: reduce number of client applications per server, implement “smart” garbage collection*

A natural question regarding the issues that were discussed during the walkthrough is why these deficiencies had not been noticed before the experiment. In fact, the team had already been aware that there were imperfections in our system and, as previously noted, had discussed some problems before we did the walkthrough. However, the Cognitive Walkthrough helped assign importance to problems that the team had noticed but had considered unimportant in designing a successful experiment. For example, the team had always thought that editing a cell formula was unintuitive and physically tricky. Because Forms/3 is a research prototype, these types of imperfections were expected, and presumed unrelated to a successful experiment design. The walkthrough dispelled this idea by illustrating their impact on the experiment task.

### 4.3.2 The Second Walkthrough

One change we made after the first walkthrough was to add an “undo validation” feature, letting participants “undo” a previous testing decision and remove the associated validation checkmark.
Cognitive Walkthrough Form: A Single Step
Task: Validate output corresponding to a set of inputs on the Grades spreadsheet
Action#: 5

2. Choosing and executing the action.
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)

There's nothing obvious about the checkbox. Our anticipated users should, however, understand the concept of the checkbox (which is fairly common in user interfaces these days) and realize that the box in the upper right of each cell is exactly that. Furthermore, the correct action lies on a continuum: if they click on a cell in the middle of the program it's not as great as if they'd clicked on the final output cell. Does that matter though, as long as they make progress?

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

There is an empty box, which unfortunately doesn't indicate much: there is no label?

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

The users will have the tutorial and the Forms/3 quick reference sheet. The checkbox is also a common user interface concept. A fairly small percentage will have a problem if the tutorial is good enough. Maybe we should have a better indicator.

TABLE 4.2: (Partial) Cognitive Walkthrough for a step [27]. Italicized text represents a summary of the notes the evaluators took during the walkthrough, showing the accumulation of ideas resulting in the last sentence.

In the first walkthrough, we had discussed three cases where an undo would be necessary: a participant accidentally validates a cell, a participant does not notice what changed on the screen after a validation and wants to redo the validation, or a participant wants to undo a sequence of validations possibly to compare the results of a different sequence. While the proposed solution was straightforward in the first two cases, it was less clear in the third: undoing
Cognitive Walkthrough Form: A Single Step

Task: Undo a past validation
Action: 3

3. Modification of goal structure.
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).

The checkmark will disappear, but if some other cell has automatically validated this one, the border will not change, which is unexpected. User may be confused and try to undo the other validations they made ... Maybe we can allow an undo of only the last validation.

3.2 Accomplished Goals 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. (%, 0, 25, 50, 75, 100)

If system is slow to respond they may not notice. Maybe in order to make their goal and feedback clearer, we could have three checkbox states: one which indicates a validated cell, one which indicates that if clicked, something will change, and one which indicates that if clicked, nothing will change.

TABLE 4.3: (Partial) Cognitive Walkthrough for a step. Italicized text represents a summary of the notes the evaluators took during the walkthrough, which resulted in an important testing-related user interface change to the question mark.

the first sequence of validations could cause inconsistencies if the sequence was removed in a order different from which it was applied. Worried about this inconsistency confusing the participants (therefore confounding the statistics), we performed a walkthrough on this specific task in order to discover whether there was a feasible solution, or whether a solution was required at all. A portion of this walkthrough is presented in Table 4.3.

The walkthrough revealed that the case of undoing a sequence of validations was indeed problematic; therefore we modified the “undo validation” feature to allow removal of only the last validation, and then only if no edits were made.
to input cells that directly or indirectly affect the last cell validated. The walk-
through also led to a discussion of possible testing strategies that participants
might follow, revealing the lack of a visual mechanism in the WYSIWYGT inter-
face to suggest some reasonable testing action to take after validating an output
value.

To suggest such actions to participants, the team decided to change the ex-
isting checkbox model (see the last sentence of Table 4.3). At the time of the
walkthrough, a checkmark represented a validation based on current inputs, a
blank meant no validation had ever been done, and a question mark meant a
cell had been previously validated but not for the current inputs. The team’s
change was to introduce an indication of potential progress. The checkmark’s
meaning remained the same. Both an empty checkbox and a question mark
would indicate that a cell had not been validated for the current inputs. How-
ever, an empty checkbox would also indicate that validating the cell would not
test anything that had not been previously covered, that is, no progress would
be made towards the goal of testing the spreadsheet. A question mark would
also indicate that validating the cell would test something that had not been
previously covered, that is, some progress would be made toward the goal.

The HCI expert did not participate in this walkthrough, but he observed
a portion of it to ascertain whether the team was executing it in a reasonably
correct fashion. His observation was that, although an HCI expert would have
gotten more data from the technique than the team did without an HCI expert’s
participation, the team’s use of the CW was still appropriate and within the
boundaries of the Cognitive Walkthrough method.
4.3.3 The Third Walkthrough

The revisions to the experiment design could have themselves introduced new variables. For example, during each of the previous walkthroughs, a number of questions on the walkthrough forms were answered positively under the assumption that the user would retain all of the knowledge contained within the tutorial. Yet, we had since made changes to the tutorial. Furthermore, some of the changes identified in the first walkthrough, which were expected to solve certain problems with the experiment’s design, were not implemented (such as an alternate version of dataflow arrows and the spreadsheet design issues) because they had no practical solution that could be implemented within our time constraints. In an effort to ascertain whether our revised design had solved as many of the problems identified in the first walkthrough as practical without introducing new problems, the team performed a new walkthrough on the same task used in the first walkthrough. (In this walkthrough, the HCI expert was not present.) This walkthrough did not lead to any further changes in the experiment’s design.

We made one change to the way the walkthrough itself was done. Recalling difficulties doing the first walkthrough without having the spreadsheet system present, we performed this walkthrough while actually performing each action on a computer running Forms/3, to be sure we had included all the options that would actually be available to the participants. This is not traditional in the use of Cognitive Walkthroughs for user interfaces, since they are normally used at a stage of user interface design in which there is no executable user interface to use. However, it is viable in the design of an experiment involving an existing system, and we found that doing so added accuracy and completeness to our responses to the questions.
4.4 Discussion

In Section 4.1 we posed two questions about the application and use of the Cognitive Walkthrough: The first question asked whether the CW method could be applied to the problem of evaluating the design of an experiment. Though it was not originally specified to evaluate the design of an experiment, we found the CW to be applicable to this task. We observed several advantages to evaluating the experiment via the CW as opposed to pilot studies and protocol analyses, which are other experiment evaluation mechanisms we have used. First, the process was relatively quick; with a small group of evaluators, our first walkthrough spanned a total of about four hours while subsequent walkthroughs took less time. This is much faster than preparing an “almost polished” version of the experiment, as required in a pilot study or protocol analysis. Second, the process was focused on the probability that the participants could perform each specific subtask, which led to a list of specific design issues. A pilot study can point out such problems, but, because a pilot study is limited to very few participants, the range of potential problems is not likely to be completely exposed. A third advantage was that the very wording of the Cognitive Walkthrough’s questions, while not prescriptive, was relatively constructive, pointing out where better information would help participants to perform their task. For example, in Table 4.2, question 2.5 suggested either user interface or tutorial changes for a case where there is no label associated with a correct action.

The second question posed in Section 4.1 concerned the use of the Cognitive Walkthrough without the assistance of HCI specialists. The CW method was created with the assumption that an HCI expert would be part of the team using the method and, as previously noted, there have been contradictory reports
regarding the need for such an expert. After our initial tutorial on the method, our HCI consultant observed us as we performed the first two walkthroughs described above. He found that we used the CW effectively, though probably not as well as a group of HCI experts.

We administered the redesigned experiment on a new group of participants. Chapter 5 provides details of the redesigned study and its results.
This chapter presents material on our redesigned testing experiment using the WYSIWYT methodology. We describe changes to the design of the experiment: the environment, the new participants, the tutorial, the task and the materials. We then report the results of the redesigned experiment. Much of the information in this chapter can also be found in [36].

5.1 Design of the Experiment

The objective of our study remained unchanged from the previous experiment. That is, we want to investigate the following research questions:

*RQ 1:* Do programmers who use the WYSIWYT testing methodology create test suites that are more effective in terms of du-adequacy than programmers who use an ad hoc approach?

*RQ 2:* Do programmers who use the WYSIWYT testing methodology create test suites more efficiently than programmers who use an ad hoc approach?

*RQ 3:* Are programmers who use the WYSIWYT testing methodology less overconfident about the quality of their testing than programmers who use an ad hoc approach?

*RQ 4:* Is training in the underlying test adequacy criterion and its relationship to the visual devices needed in order for spreadsheet programmers to gain
testing effectiveness or efficiency from using the methodology?

Many aspects of the redesigned experiment remained the same as in the first experiment; the following sections provide details.

5.1.1 The Environment

The Forms/3 environment described in Section 2.3.2 remained, for the most part, the same: red cell borders or arrows continue to indicate that the cell or du-association is not tested, blue continues to indicate tested and purple indicates partially tested. Though the basic environment remained the same, some features were added or modified.

The most important change modified the methodology significantly:

- Changes to the checkbox feedback as described in Section 4.3.2 were implemented to help the participants determine the next action to take.

Other changes were:

- A “remove the last validation” feature was added, based on the discussion generated during our second Cognitive Walkthrough, partially described in Section 4.3.2.

- A “Tested %” indicator was added so that testers could easily determine what percent of the spreadsheet has been tested and to reinforce the progress made as new decisions were recorded.

- Changes were made to the shades of purple used to indicate partial testedness to make them more distinct from the red used to indicated not tested and the blue used to indicate fully tested.

- The process used to edit an input value was simplified and instructions were added to the edit window. (This change affected both the WYSIWYG and Ad Hoc environments.)
The formula display mechanism was simplified. (This change affected both the WYSIWYT and Ad Hoc environments.)

5.1.2 The Participants

Seventy-eight students participated in the experiment, drawn from the same three Computer Science courses which supplied participants in the previous experiment: two upper-division undergraduate courses and one graduate course in Computer Science. We were careful to screen out any students that had participated in the previous experiment. The participants were randomly divided into two groups, subject to balancing the number of undergraduate and graduate students between the groups. The control (Ad Hoc) group did not have access to the WYSIWYT methodology and represents programmers who test in an ad hoc fashion. The treatment (WYSIWYT) group did have access to the methodology. By reducing the number of groups from 3 to 2, we increased the sample size of each group by more than 50%.

Of the 78 participants, the Ad Hoc group and WYSIWYT groups contained 37 and 41, respectively. The difference in group size was due to a few students failing to arrive for their appointments. The group sizes were subsequently reduced when we decided to omit data for participants whose measurable activity level was zero (as revealed by transcripts of the sessions), whose computer crashed during the experiment, or who inadvertently corrupted their data in other ways. The removal of these participants reduced the number in the Ad Hoc and WYSIWYT groups to 30 and 39, respectively.

To ascertain whether the participants in the two groups had similar backgrounds, we administered a background questionnaire to each participant and
analyzed the data. Results are summarized in Table 5.1. Our analysis showed homogeneity between the groups except in the GPA category, where we found that the Ad Hoc participants had significantly higher GPAs than the WYSIWYT participants. Since the GPAs were self-reported and approximately one-third of the participants did not report a GPA, this non-homogeneity of GPAs is tenuous.

### 5.1.3 The Tutorial

The tutorial remained nearly the same as that used in the previous experiment. We shortened it by removing unnecessary details about moving and resizing cells, and replacing the formula display and edit instructions with shorter, simplified instructions for the redesigned display and edit mechanisms. For the WYSIWYT group, we removed from the testing instructions one example spreadsheet that did not add any useful information.

The first two portions of the tutorial were the same as the previous tutorial except for details removed or replaced as noted above. The first portion introduced the Forms/3 environment (e.g., how to edit cells, display formulas, etc.) The second portion described testing as trying various input values and

<table>
<thead>
<tr>
<th></th>
<th>Ad Hoc</th>
<th>WYSIWYT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number in Group</td>
<td>30</td>
<td>39</td>
</tr>
<tr>
<td>Number of Graduate Students in Group</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>Overall GPA</td>
<td>3.45</td>
<td>3.2</td>
</tr>
<tr>
<td>CS GPA</td>
<td>3.7</td>
<td>3.5</td>
</tr>
<tr>
<td>Number of Programming Languages Known</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Number with Spreadsheet Experience in Group</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>Number with Professional Experience in Group</td>
<td>11</td>
<td>20</td>
</tr>
</tbody>
</table>

**TABLE 5.1: Group demographics.** Analysis shows homogeneity between the groups except in the GPA category.
recording decisions made about other cells’ values. All participants were instructed to use the check boxes to record decisions about correct cell values and to record information about incorrect cell values in a special cell named BugRecorder (formerly named OutputErrors).

In redesigning the third tutorial part for the WYSIWYT group, we were faced with an important decision: whether to provide a technical explanation of the underlying concepts of such things as du-associations and their relationship to the methodology’s feedback. Because one of our goals is to show that the testing methodology does not require an understanding of the underlying theory, we chose to explain only that red means “not tested”, blue means “tested”, and purple means “partially tested”. The question marks were described as meaning “recording a decision here will help test part of the spreadsheet that has not been tested”, check marks as “you have recorded a decision here”, blanks as “you have previously recorded a decision that covers this case”. These explanations were integrated in several examples of trying different inputs and recording decisions. As part of the examples, we gave instructions on how to display the colored dataflow arrows at both the cell level and subexpression level. We did not mention the underlying concepts of du-associations, nor did we describe nonexecutable du-associations.

At the end of the tutorial, everyone was given unstructured time to practice their Forms/3 skills. Regardless of group, the total training time was identical; participants not receiving training on the WYSIWYT methodology’s feedback were given more unstructured time to practice using Forms/3. This was necessary; unequal training in Forms/3 could have confounded the results and the training on the WYSIWYT methodology also provided practice using Forms/3.
5.1.4 The Materials and Task

We reused the Clock and Grades spreadsheets after minor changes were made: In the Clock spreadsheet, the clock face was slightly increased in size so that participants would not erroneously report a bug about the length of the minute hand, the omission in the ErrorsExist? cell of the Grades spreadsheet was corrected, and the OutputErrors cell was renamed BugRecorder for both spreadsheets.

The task remained the same, the participants were asked to test the spreadsheets and were given 15 minutes per spreadsheet to perform their testing. The experiment was counterbalanced with respect to problem type: participants in each group worked both problems, but half the participants in each group tested Grades first, whereas the other half tested Clock first. At the beginning of each testing session we instructed the participants to read a description of the spreadsheet they were about to test. The descriptions were modified to included clearer details of what the spreadsheet was to accomplish, the range of accepted input values and the error messages expected for out-of-range inputs.

5.2 Results

As in the previous experiment, we collected the following data: (1) the actions (edits, decisions, etc.) of each participant throughout the experiment's sessions were automatically recorded in transcript files, (2) the test suites created by the participants' actions, derived from the information in the transcript files, and (3) the participants' answers to three questionnaires. The first questionnaire collected information about participants' backgrounds, as noted in Section 5.1.2. At the end of each testing session the participants completed a
questionnaire which collected subjective information about their perceived testing effectiveness. In the third questionnaire, the WYSIWYT participants were also questioned about their understanding and use of the WYSIWYT visual features. The post-testing-session questionnaires were substantially rewritten to more carefully gather the data we wanted as will be seen in Sections 5.2.3 and 5.3.

The overall results were strongly in favor of the WYSIWYT methodology. The following sections detail results for each of our hypothesis.

5.2.1 Effectiveness

Our first research question considers whether using the WYSIWYT methodology increased testing effectiveness. We measured effectiveness in terms of du-adequacy, the percent of executable du-associations covered.

We examined the data (Figure 5.1) to see whether it satisfied the requirements for using normal-theory analyses such as t-tests and analysis of variance. There were some indications of skew, but the analysis of variance is robust against moderate skew; however, conservative non-parametric alternatives were also used where possible, and gave identical patterns of results.
We analyzed effectiveness using analysis of variance with two factors, Environment (Ad Hoc or WYSIWYTY) and Problem (Clock or Grades). Each participant experienced only one environment and attempted both problems, so the Environment factor was treated as independent groups and the Problem factor was treated as having repeated measures.

Analysis of variance showed that the effectiveness of the WYSIWYTY group was very significantly higher than the effectiveness of the Ad Hoc group ($F = 8.56, df = 1,67, p = 0.0047$). There was a significant difference between effectiveness on the problems ($F = 9.632, df = 1,67, p = 0.0028$) but no significant interaction effect, that is, the WYSIWYTY group showed the same pattern of greater effectiveness on both Clock and Grades. Independent non-parametric tests were performed on the problems considered separately and on the pooled data, confirming the significant differences between environments (Mann-Whitney, Clock: $p = 0.0001$; Grades: $p = 0.0083$; Pooled Data: $p < 0.0001$).

Thus, in terms of our first research question, programmers who used the WYSIWYTY testing methodology during the experiment created test suites that are more effective, as measured by du-adequacy, than programmers who used an ad hoc approach.

5.2.2 Efficiency

Our second research question considers whether using the methodology increased efficiency. One view of efficiency measures "wasted effort" in running

---

1 The results from two of the participants appeared to be outliers, and further investigation showed that one was not following the instructions and the other had limited experience with English. This analyses includes results from these two participants, but the same pattern of results emerged from analyses in which their scores were eliminated.
redundant tests that do not increase coverage\textsuperscript{2}. As we noted in Section 3.2.2 this measure implies that multiple test cases exercising the same du-association do not increase testing effectiveness and some evidence exists to suggest that this assumption is overly simplistic [32]; however, this assumption is common to the use of structural coverage criteria.

An analysis of variance on the redundancy data showed that the percentage of redundant test cases created by the WYSIWYT group was significantly lower than the percentage of redundant test cases in the test suites created by the Ad Hoc group ($F = 47.987$, df= 1.67, $p< 0.0001$). There was a significant difference between the percentage of redundant test cases on the two problems ($F = 8.37$, df= 1.67, $p= 0.0045$) but no significant interaction effect, that is, the WYSIWYT participants showed the same pattern of lower redundancy on both Clock and Grade, see Table 5.2. As with the effectiveness data, we performed independent non-parametric tests (Mann-Whitney) on the two problems considered separately and on the pooled data, again confirming the significant differences between environments.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
 & Ad Hoc & WYSIWYT \\
\hline
Clock & & \\
# Tests & 20 & 19 \\
% Redundant & 61.3 & 15.4 \\
\hline
Grades & & \\
# Tests & 14 & 18 \\
% Redundant & 44.0 & 4.3 \\
\hline
\end{tabular}
\caption{Medians for the redundancy data.}
\end{table}

\textsuperscript{2}This is a more reasonable view of redundancy than we used in the previous experiment in which we considered the redundancy in terms of individual du-associations.
Another view of efficiency is the speed with which coverage was obtained. The data for speed of coverage was derived by dividing each of the 15 minute testing sessions into three 5-minute intervals and determining the du-adequacy at the end of each interval. Figure 5.2 shows that the WYSIWYT group achieved coverage faster than the Ad Hoc group. For the Clock spreadsheet, the difference is significant in the second and third time periods (Mann-Whitney, 2nd period: $p= 0.0497$, 3rd period: $p= 0.0001$). For the Grades spreadsheet, the difference is significant in the third time period (Mann Whitney, 3rd period: $p= 0.0083$).

Thus, in terms of our second research question, programmers who used the WYSIWYT testing methodology during the experiment created test suites more efficiently, as measured by redundancy and speed, than programmers who used an ad hoc approach.

5.2.3 Overconfidence

The WYSIWYT participants’ higher coverage achieved more efficiently are two benefits that the above results show. However, earlier in this paper, we discussed
several studies that indicate that spreadsheet programmers have unwarranted confidence in the accuracy of their spreadsheets. The methodology may be of little use in practice if overconfidence causes spreadsheet programmers to stop working on their spreadsheets before making much use of its guidance. For this reason, reducing overconfidence has been an important goal of the WYSIWYT methodology for the spreadsheet domain.

At the end of each 15-minute testing session we asked the participants to rate how well they thought they tested the spreadsheet by answering the following question:

1. How well do you think you tested the spreadsheet?
   a) Really, really well. (If you graded it, I’d get an A)
   b) Better than average. (If you graded it, I’d get a B)
   c) About average. (If you graded it, I’d get a C)
   d) Worse than average. (If you graded it, I’d get a D)
   e) Poorly. (If you graded it, I’d get an F)

We compared each participant’s answer to this question to our “grading” of their du-adequacy. We assigned grades of A - F based on a standard grading scale, coverage of 90-100%⇒A, 80-89%⇒B, ... 0-59%⇒F. If a self-reported grade was higher than our assigned grade for the participant, he or she was categorized as “overconfident”, otherwise he or she was categorized as “not overconfident”; see Table 5.3. We analyzed the overconfidence data using Fisher’s Exact Test on the two problems considered separately. The WYSIWYT group had significantly fewer than expected participants in the overconfident category while the Ad Hoc group had significantly more than expected participants in the overconfident category (Clock: p= 0.025; Grades: p= 0.0155).
TABLE 5.3: Overconfidence data. For each problem, the WYSIWYT group had significantly fewer overconfident participants than expected when compared to the Ad Hoc Group.

<table>
<thead>
<tr>
<th></th>
<th>Ad Hoc</th>
<th>WYSIWYT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overconfident</td>
<td>16</td>
<td>10</td>
</tr>
<tr>
<td>Not Overconfident</td>
<td>14</td>
<td>29</td>
</tr>
<tr>
<td>Grades</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overconfident</td>
<td>20</td>
<td>14</td>
</tr>
<tr>
<td>Not Overconfident</td>
<td>10</td>
<td>25</td>
</tr>
</tbody>
</table>

In terms of our third research question, programmers who used the WYSIWYT testing methodology during the experiment were less overconfident about the quality of their testing than programmers who used an ad hoc approach.

5.3 Discussion

The results of our analyses showed that participants using the WYSIWYT methodology during the experiment created more effective test suites faster, with less redundancy of coverage, and with less overconfidence about the quality of their testing than participants who used an ad hoc approach. Given such strong results, a natural question that arises is whether particular portions of the methodology’s communication devices are key to the results. For example, would it be possible to attain the same results without the colors to attract attention to untested areas, using only the check marks and question marks to guide the user through the testing activities? Although we do not have a rigorous answer to this question, we do have the participants’ opinions about which aspects were the most helpful, as reported on their questionnaires. These are given in Table 5.4, in descending order of the participants’ votes in the
TABLE 5.4: WYSIWYT group helpfulness ratings: percent of participants who rated the device in each of the possible helpfulness categories.

<table>
<thead>
<tr>
<th>How helpful were:</th>
<th>Very Helpful</th>
<th>Helpful</th>
<th>Not Helpful</th>
</tr>
</thead>
<tbody>
<tr>
<td>question marks</td>
<td>69%</td>
<td>31%</td>
<td>0%</td>
</tr>
<tr>
<td>clicking to validate</td>
<td>64%</td>
<td>36%</td>
<td>0%</td>
</tr>
<tr>
<td>colored cell borders</td>
<td>56%</td>
<td>44%</td>
<td>0%</td>
</tr>
<tr>
<td>colored arrows</td>
<td>51%</td>
<td>41%</td>
<td>8%</td>
</tr>
<tr>
<td>check marks</td>
<td>44%</td>
<td>49%</td>
<td>8%</td>
</tr>
<tr>
<td>“Tested” indicator</td>
<td>36%</td>
<td>56%</td>
<td>8%</td>
</tr>
<tr>
<td>blanks</td>
<td>23%</td>
<td>51%</td>
<td>26%</td>
</tr>
</tbody>
</table>

TABLE 5.5: Participants’ opinions of the meanings of the visual devices. Percentages reflect the number of participants who chose the correct response (shown in parentheses). The questions were multiple choice and directed the participants not to guess at the answer.

A red cell border indicates that the cell is (not tested) 100%
A question mark in a cell’s check box indicates that (the current input tests part of the spreadsheet not previously tested) 87%
A blue arrow between cells indicates (the relationship between the two cells is fully tested) 64%

“Very Helpful” category. In the table, the choices listed are abbreviations of the questionnaire wording: to distinguish ideas of feedback from user actions in the questions, we worded the choices about output devices with the words “seeing the...” (e.g., “seeing the colored cell borders”), as opposed to actions the user could take such as “clicking to validate.”

Still, the participants’ opinions of helpfulness could be misleading if they badly misunderstood what the visual devices are intended to communicate. Given that they had only 20 minutes to learn the entire environment and receive their task instructions, this was a possibility. To assess their understanding, and provide some insight into the understandability of the methodology, we
TABLE 5.6: Learning Effects: Compare the medians of Problems 1 and 2. The Ad Hoc group achieved about the same coverage on Problem 2 as on Problem 1, but the WYSIWYT group achieved significantly higher coverage on Problem 2 than on Problem 1.

<table>
<thead>
<tr>
<th>Problem 1</th>
<th>Ad Hoc</th>
<th>WYSIWYT</th>
<th>Problem 2</th>
<th>Ad Hoc</th>
<th>WYSIWYT</th>
</tr>
</thead>
<tbody>
<tr>
<td>% Tested</td>
<td>69.0</td>
<td>82.7</td>
<td>71.6</td>
<td>97.8</td>
<td></td>
</tr>
<tr>
<td># Tests</td>
<td>13</td>
<td>20</td>
<td>22</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>% Redundant</td>
<td>51.3</td>
<td>11.1</td>
<td>56.3</td>
<td>7.7</td>
<td></td>
</tr>
</tbody>
</table>

asked several questions about the meanings of the different devices. The results, summarized in Table 5.5, suggest that WYSIWYT participants understood the features reasonably well.

Our methodology’s usefulness would be rather limited if a steep learning curve prevented benefits without a time-consuming initial effort. To gain some insights into the possibility of this problem, we compared the Ad Hoc and WYSIWYT participants’ performance on the first problem (shown in the left half of Table 5.6). Despite whatever learning curve is associated with the methodology, WYSIWYT group still had greater effectiveness and greater efficiency in the first problem than did the Ad Hoc group (Mann-Whitney, % Tested: p = 0.0083; Redundancy: p < 0.0001). Additional benefits came with experience for the WYSIWYT group. In particular, the WYSIWYT group seemed to learn to improve their test selections on the second problem. They generated about the same number of test cases for each problem but significantly increased their coverage (by 15%) from the first problem to the second problem, whereas the Ad Hoc group, despite generating about one third more tests, did not significantly increase their coverage (Wilcoxon, Ad Hoc: p = 0.26; WYSIWYT: p = 0.0005). Additionally, from problem 1 to problem 2, the WYSIWYT participants decreased their percent of redundant tests, see Table 5.6.
Chapter 6

CONCLUSIONS

In this chapter we consider some of the threats to the validity of our investigation, future work needed to complete the picture begun by our investigation and present the conclusion of this thesis.

6.1 Threats to Validity

In our experiment we attempted to address threats to internal validity by balancing the two groups of participants according to year in school and course, by counterbalancing with respect to problem type, by equalizing training time, and by selecting problems from familiar domains. However, as in most controlled experiments, threats to external validity are more difficult to address given the need to control all other factors. For example, computer science students may not be representative of any sizable segment of the population of spreadsheet programmers. In particular, they cannot be said to be representative of end-user spreadsheet developers. Similarly, the spreadsheets used in the experiment may not be representative of the population of spreadsheets. However, although the spreadsheets may seem rather simple, given the limited testing time, few participants reached 100% du-adequacy (Clock: 21.7%; Grades: 1.4%).

As we have mentioned before, the WYSIWYT methodology does not currently handle non-executable du-associations in a way that is helpful to the task
of testing; yet, non-executable du-association do occur in spreadsheets. A large number of non-executable du-associations in a spreadsheet would be a barrier to the effectiveness of the methodology, and the experiment did not address this issue. Instead, we circumvented it to the extent possible by attempting to minimize the number of non-executable du-associations in each spreadsheet: Grades contained 2 (out of 95), and Clock contained 10 (out of 83).

Because the focus of the study was on effectiveness and efficiency of testing, the spreadsheets contained no faults. This may be unrealistic; however, including faults in the spreadsheets would have confounded the data about testing effectiveness and efficiency since the participants would not be focused on the single task of testing the spreadsheets. Moreover, to enable a clear analysis of this task, we did not allow participants to change formulas; including faults without allowing corrections would not be realistic. A separate study of debugging tasks can be found in [7]. To motivate the participants, however, they were informed that their spreadsheets “might or might not” contain faults. In fact, several participants did report faults of a cosmetic nature.

Testing effectiveness was measured by the percentage of the du-associations covered by the test cases, but this is not the only possible measure. Another measure of effectiveness is the number of faults detected; however, as discussed above, that measure also presents threats to validity. A correlation between effectiveness in terms of du-adequacy and effectiveness at finding faults is supported by evidence in empirical studies of imperative programs [10, 14, 42] and in previous empirical work we performed in the spreadsheet paradigm [34].
6.2 Future Work

This testing experiment is just a small part of a much larger investigation of the problems involved in bringing some of the principles of software engineering to the informal environment of spreadsheet languages. There are questions remaining about the WYSIWYT methodology:

1. Will end users derive the same benefits from the methodology as the participants of this study, i.e., do our results generalize to a larger population?
2. Will the methodology still be valuable when testing larger spreadsheets, i.e., do our results scaled up?
3. Will the WYSIWYT methodology help users find errors in existing spreadsheets?
4. Will the methodology help users write spreadsheets that have fewer errors?

Some of these questions will be answered by additional experiments that we are planning. We have completed an experiment investigating whether the WYSIWYT methodology helps in finding errors in spreadsheets (Question 3). The results of that investigation, detailed in [7], showed that participants using the WYSIWYT methodology generally did a better job of debugging a spreadsheet seeded with errors.

6.3 Conclusion

The software engineering properties of spreadsheet languages have rarely been studied; this is a serious omission because these languages are being used to create production software upon which real decisions are based. Further, research shows that many of the spreadsheets created with these languages contain faults.
For these reasons, it is important to provide support for mechanisms, such as testing, that can help spreadsheet programmers determine the reliability of values produced by their spreadsheets.

In this paper we reported empirical results about a testing methodology aimed at this need. The results were:

- Participants using the WYSIWYT methodology performed significantly more effective testing than did the Ad Hoc participants, as measured by du-adequacy.
- Participants using the WYSIWYT methodology were significantly more efficient testers than the Ad Hoc participants, as measured by redundancy and speed.
- Participants using the WYSIWYT methodology were significantly less overconfident than were the Ad Hoc participants.

Further, it was possible for the WYSIWYT group to achieve these benefits without training on the underlying testing theory. These results are encouraging, because they suggest that it is possible to achieve at least some benefits of formal notions of testing without formal training in the testing principles behind a testing methodology. However, this experiment is the first that we have completed, and thus includes only one segment of the population using spreadsheet languages. Future experiments will be required before it will be clear whether the methodology brings similar benefits to other kinds of users, especially end users.
BIBLIOGRAPHY


