Research

Using a structured design approach to reduce risks in end user spreadsheet development

Diane Janvrina,*, Joline Morrisonb

aDepartment of Accounting, University of Iowa, Iowa City, IA 52242, USA
bDepartment of MIS, University of Wisconsin, Eau Claire Eau Claire, WI 54702, USA

Received 31 July 1998; accepted 24 May 1999

Abstract

Computations performed using end-user developed spreadsheets have resulted in serious errors and represent a major control risk to organizations. Literature suggests that factors contributing to spreadsheet errors include developer inexperience, poor design approaches, application types, problem complexity, time pressure, and presence or absence of review procedures. We explore the impact of using a structured design approach for spreadsheet development. We used two field experiments and found that subjects using the design approach showed a significant reduction in the number of 'linking errors,' i.e., mistakes in creating links between values that must connect one area of the spreadsheet to another or from one worksheet to another in a common workbook. Our results provide evidence that design approaches that explicitly identify potential error factors may improve end-user application reliability. We also observed that factors such as gender, application expertise, and workgroup configuration also influenced spreadsheet error rates. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Risks in end user development; Spreadsheet errors; Structured design

1. Introduction

During the past decade, the use of end-user developed applications, especially spreadsheets, has grown exponentially. However, when identifying and evaluating risks within organizations, end-user developed computer applications are often overlooked. Studies have revealed that as many as 44% of all end-user spreadsheets contain at least one error [5]. Such errors can have catastrophic results. For example, a Florida based construction company lost US$ 254,000 by relying on an incorrect spreadsheet analysis for project costing [39]. A large mutual fund incorrectly reported a US$ 0.1 billion loss as a US$ 2.3 billion gain by placing the wrong sign on a US$ 1.2 billion spreadsheet input entry [38].

Previous effort suggests that errors in end-user developed applications may be influenced by several factors [28,30]. Here we develop a framework to examine end user risks and specifically to investigate the impact of one risk mitigating factor, a structured design approach, on spreadsheet error rates and development times.

2. Background and model development

Cotterman and Kumar [9] classify end-user computing risks into three major parts: end-user devel-
opment, end-user operation, and end-user control of information systems. We focus on the domain of end-user development. To our knowledge, no work has specifically identified and then empirically validated factors impacting outcomes in end-user developed applications, such as spreadsheets. Literature suggests that potential factors include developer attributes, development approach, developer configuration, problem/process characteristics and application domain.

2.1. Developer attributes

Our work focuses on end-users who are both producers and consumers of information. Several studies have indicated that end-users are diverse, and that this diversity leads to a need for differentiated education, training, support, software tools and risk assessment [3,12,14,19,34]. Harrison and Rainer [17] assert that specific end-user attributes include age, gender, computer experience, computer confidence, math anxiety, and cognitive style. Curtis et. al. field studies on software development [10] repeatedly confirm the importance for software developers to understand their application domain. Adelson and Soloway [1] emphasize the importance of expertise in software design. Frank, Shamir and Briggs [15] find that PC user knowledge – i.e., application expertise – positively impacted security-related behavior. Further, several researchers indicate that user training and relevant experience are important factors e.g., [2,37].

2.2. Development approach

Design activities for end-user developed applications tend to be unstructured and often non-existent. Salchenberger notes that end-users tend to develop and implement models without performing requirements analysis or developing a system design. Brown and Gould observed that their participants spent little time planning before launching into spreadsheet coding. Ronen, Palley and Lucas [35] suggest that several attributes of spreadsheets (i.e., non-professional developers, shorter development life cycles, modification ease) tend to preclude a formal spreadsheet analysis and design process.

Almost all published spreadsheet design approaches agree on one principle: the importance of keeping work areas small in order to make errors easier to find, limit the damaging effects of errors, and make spreadsheets easier to understand and maintain [33]. Markus and Keil [24] note that structured design methodologies increase the accuracy and reliability of information systems. Salchenberger proposes the use of structured design methodologies in end-user development and discusses their effectiveness in different situations. Cheney, Mann and Amoroso [7] also support the use of a structured design methodology; they found that the more structured the tasks of the end-user, the more likely the success of the application. Ronen, Palley and Lucas describe a structured spreadsheet design approach based on modifying conventional data flow diagram symbols to create spreadsheet flow diagrams. Several practitioner-directed articles encourage the use of structured design approaches and planning for spreadsheet development.

2.3. Developer configuration

Panko and Halverson [29,31] and Nardi and Miller [26] note that the accuracy of end-user developed applications might be increased by encouraging developers to work in teams. However, the former also suggest that the increased communication and coordination costs of working in groups may outweigh the benefits of reduced error rates.

2.4. Problem/process characteristics

Several studies identify the difficulties associated with complex versus non-complex programming problems e.g., [13,36]. Studies on the effect of time pressure on programming activities indicate that due to time pressure, end-user developers may not spend sufficient time on problem definition and diagnosis e.g., [2,6]. They also note an absence of formal review and validation procedures within end-user developed applications.

2.5. Application domain

While this research covers a wide range of application domains, spreadsheet development should be a central issue, because of the widespread use of end-user developed spreadsheets in the workplace. Panko
and Halverson appropriately summarize the state of spreadsheet research as follows:

In the past, spreadsheeting has been the Rodney Dangerfield [e.g., a not well respected subject] of computer applications. Although everyone agrees that it is extremely important, few researchers have studied it in any depth. Given its potential for harm as well as for good, this situation needs to change.

Psychologists Hendry and Green [20] assert that spreadsheets may simplify programming by suppressing the complexities associated with conceptually simple operations like adding a list of numbers or handling input and output. As a result, users tend to ignore their weaknesses, such as their potential for making errors or their difficulties in debugging or reusing spreadsheets.

2.6. Outcomes

Salchenberger suggests that factors to use in judging the quality of end-user developed applications include reliability, ease of use, maintainability, and auditability. Reliability can be defined as the absence of logical and mathematical errors. One of the earliest studies on spreadsheet outcomes (conducted by Brown and Gould) find that of 27 spreadsheets created by experienced spreadsheet users, 44% contained user-generated errors. In another study, Davies and Ikin [11] audited a sample of 19 spreadsheets in an organization and discovered four worksheets contained major errors and 13 had inadequate documentation. Panko and Halverson [29] ask students to develop a spreadsheet model working individually, in groups of two, and in groups of four, and observed total error rates ranging from 53–80%. They also note that very little time (only a few minutes) was spent in design activities prior to implementation. They also note that their participants made several different types of errors. At the highest level, errors could be divided into two basic types: oversight and conceptual. They defined oversight errors as ‘dumb’ ones such as typographical errors and misreading numbers on the problem statement. In contrast, conceptual errors involve domain knowledge or modeling logic mistakes. Panko and Halverson use Lorge and Solomon’s [23] terminology to further divide conceptual errors into Eureka and non-Eureka errors. Once a Eureka error is identified, everyone can quickly agree that it exists. However, non-Eureka errors are those where the mistake cannot be easily demonstrated.

Generally, researchers evaluate the usability of end-user developed applications with satisfaction survey tools [14,22]. Salchenberger defines maintainability as the ability to change or update the system as needs change, and auditability as the ability to determine the source of data values in the model. Few researchers have evaluated end-user developed applications for either maintainability or auditability [32]. We also propose the inclusion of cost (usually measured in terms of development time) as another important outcome.

3. Structured spreadsheet design

Based on our review, we developed an initial research framework (Fig. 1) that summarizes risk factors and their relationships to potential outcomes. Our focus is to determine the impacts of development approach on spreadsheet reliability and cost.

Modern spreadsheet applications enable users to modularize large spreadsheets by creating ‘work-
sheets’ (individual spreadsheet work areas) within a larger ‘workbook’ (set of related worksheets). Users can link values from one worksheet to another within the same workbook, or link sheets among multiple workbooks. This technique readily supports scenario analysis. For example, all of the schedules leading to a master budget can be prepared on separate worksheets within a workbook, and then linked together so that the impact of a change in one parameter can be quickly evaluated. However, this feature also leads to the likelihood of ‘linking errors.’ Data on a source worksheet may be linked to the wrong cell on the target worksheet or a value on a target worksheet that should have been linked from a previous worksheet is ‘hard-coded:’ an actual value from a previous scenario is typed in. In such cases, when the value should change for subsequent scenarios, they do not and errors result. Also, values that are unknowingly linked to subsequent sheets might be deleted, thus destroying the integrity of the subsequent worksheet. To support the task of designing linked worksheets, we build upon the work of Ronen, Palley and Lucas to develop a design methodology based on data flow diagram symbols.

Data flow diagrams (DFDs) are documentation tools used by systems analysts to show organizational data sources, destinations, processes, and data inputs and outputs. Our basic modeling symbols (Fig. 2) are similar to typical symbols in DFD.

We assumed that explicitly identifying links in the design stage should reduce linking errors made during the spreadsheet coding phase. Fig. 3 shows an example design model. Worksheet pages are created to represent each of the modules shown on the design model (e.g., Input Page, Sales Budget, Production Budget, Pro Forma Income Statement). Inputs from external sources are entered on a centralized Input Page, and then flow into the Sales Budget, where they are transformed into data items needed for calculations on the other schedules. Using named variables, the specified links are created. Inputs are easily changed on the Input Page for scenario analysis.

4. Evaluation studies

Based on the research framework, we designed a field experiment to explore the impacts of the structured design approach on spreadsheet errors. The reduced research model guiding the study is shown in Fig. 4. Harrison and Rainer note a positive relationship between developer confidence and improved program outcomes. Other researchers identify a positive relationship between domain and application expertise and improved outcomes. Therefore, we hypothesized that both developer attributes and development approach would influence error rate and development time.

Structured systems design approaches e.g. [8,16] have successfully been used in systems development
for over 25 years. They also help in making reusable components. This approach can be applied to spreadsheet design by identifying different worksheets and their corresponding data inflows and outflows. Thus, each individual worksheet effectively becomes a structured module that takes a defined set of inputs and translates it into a prescribed set of outputs. Since the worksheet links are explicitly defined in the design, we hypothesized that the result should be a decreased error rate for linking errors.

4.1. Study #1

4.1.1. Subjects, treatment, and measures

Sixty-one upper- and masters-level accounting and business administration majors in an accounting information systems course at a large public university were randomly assigned to two groups: treatment (structured design) and control (ad hoc design). All subjects were given 1 hour of formal training in spreadsheet development and told to develop a single spreadsheet workbook containing a series of linked worksheets using Microsoft Excel. The assigned problem contained 10 separate worksheets with a total of 51 linked data values among the worksheets. A paper template of the worksheets with one complete data set (from Moriarity and Allen [25]) was given to all subjects to provide check figures and mitigate non-linking errors. (Only numerical values were specified, and the subjects still had to develop and correctly implement the underlying design in order to successfully perform scenario analysis.) All subjects were required to maintain logs recording time spent on design and development activities. The treatment group was required to submit their spreadsheet design prior to being assigned a password to provide computer data entry.

A pre-test questionnaire was administered to identify gender, age, and class level, as well as self-reported pre-test application expertise (in terms of both spreadsheets in general as well as the specific spreadsheet application, Microsoft Excel), DFD development expertise, and design confidence. Development time was determined by combining two measures: (1) self-reported design logs covering two- or three-day periods during the experiment; and (2) system-recorded measures of actual implementation times. The subjects were given 16 days to complete the assignment. At the end of that period, questionnaires were administered to ascertain post-test spreadsheet and Excel expertise and design and coding confidence. Error rates were determined by examining the spreadsheet data files.

4.2. Analysis and results

Due to an unexpected shortage of software and hardware to complete the assignment, we were forced to allow subjects in Study #1 to work either individually or in self-selected pairs within treatment groups. In Study #2 all subjects worked alone. Means and standard deviations for the independent and dependent variables for all subjects as well as results sorted by configuration and treatment group are shown in Table 1. Analysis was performed on a per-subject basis, with scores for time and error rate replicated for each subject in a given dyad. A selection internal validity threat exists, since individuals were assigned to the treatment group based on their class section. Pre-test questionnaire data was used to determine the impact of the selection problem. The treatment group was significantly older than the control group ($p < 0.03$), but no significant differences were noted for pre-test spreadsheet expertise, Excel expertise, or DFD expertise.

We used least squares multiple regression to test the research model components and proposed linkages. Examination of the correlations among the independent variables revealed no evidence of multicollinearity: that is, $r \leq 0.8$ [4]. Table 2 shows that gender, age group, class level, and pre-test Excel expertise had no impact on any of the outcomes, while configuration
(singles versus pairs), treatment group (structured versus ad hoc design), and pre-test spreadsheet expertise all displayed significant correlations with one or more of the dependent variables.

Following Heise [18], the independent variables that did not explain a significant variation in the outcomes were excluded from further analysis. Table 3 shows that the variations in the dependent variables are explained, to some extent, by the independent variables in the reduced model. MANOVA analysis was used to detect potential interactions between the configuration and treatment variables on error rate.
This revealed a weak significant difference (p 10) between the control to the treatment group, as well as a weak significant interaction (p 10) among the configurations and the treatment.

4.3. Summary and discussion

Overall, subjects using the structured design technique exhibited significantly lower error rates: of the 51 possible links in the spreadsheet, the structured design subjects had an average of 3.57 (7%) linking errors, while the ad hoc design subjects averaged 4.1 (10%) linking errors. We believe this was because the design methodology forced the developers to identify the links between worksheets prior to implementation. This explicit identification was then carried over into the implementation. Virtually all of the linking errors were from omitted links (where a value had been ‘hard coded’ instead of linked) rather than incorrect linkage. Possibly subjects were relying heavily on the check figures provided for the first scenario instead of examining the reasonableness of their answers for the subsequent cases.

Improvement in linking errors was mitigated by subject configuration: subjects working alone within the ad hoc design group averaged 7.44 linking errors per spreadsheet (14%), while subjects working alone in the design group and both sets of paired subjects all averaged approximately 3.8 linking errors per spreadsheet (7%).

For subjects working alone, the MANOVA analysis results imply that the use of the design methodology had a significant impact. Unfortunately, the advantage of using the structured design methodology was weakened when the subjects were working in pairs. Two potential explanations occurred to us: (1) subjects working in pairs gained other advantages that offset the impact of the design approach; or (2) another external factor induced some subjects to elect to work in pairs and it allowed them to create spreadsheets with fewer linking errors.

Our results also indicated that subjects working in pairs reported higher post-test spreadsheet expertise but lower design confidence. The latter result seemed inconsistent with the prior literature. The significantly lower overall time investment for the structured design subjects was unexpected but encouraging. (This result does not consider the time spent initially learning the design methodology, however). Finally, as expected, pre-test spreadsheet expertise was a significant predictor of post-test spreadsheet expertise, reduced error rates, and reduced time investments. The results of this study suggested that developer attributes, configurations, and the development approaches are all potential factors impacting end-user development outcomes. However, the unexpected results for time and post-test confidence as well as the unexpected configuration variable induced us to perform a second study with revised procedures and measures.

4.4. Study #2

To confirm the results of Study #1 and remove the possible confound involved in allowing subjects to work in self-selected pairs, a second study was performed. The problem used was slightly more complex than the one in Study #1, with 10 different worksheets and a total of 66 linked cells. All subjects were
required to work individually so as to increase sample size and allow a more isolated investigation of the impacts of the design methodology and developer attributes. Error types were expanded to include conceptual and oversight errors as well as linking errors. A blank template (from [21]) with only a single check figure was given. Explicit post-test measures were introduced to assess design confidence, coding confidence, and confidence that the spreadsheet was error-free. Additionally, a measure was included to explore the impact of potential domain expertise: subjects who were accounting majors had additional experience and coursework in preparing financial reports and other master budgeting topics than the non-accounting majors. Subjects in the treatment group were also asked to rate their post-treatment DFD development confidence, and to rate their satisfaction with the development approach. Specifically, they were asked the degree to which they agreed or disagreed with statements regarding: (1) whether the DFD design methodology helped them develop a better spreadsheet; (2) if using the methodology made the development effort take longer; and (3) whether they would use the methodology the next time they developed a complex spreadsheet.

4.5. Analysis and results

Table 4 shows overall means and standard deviations for all measures. No significant differences in pre-test questionnaire information were noted between treatment and control groups, thus reducing the selection internal validity concern. Correlation analysis indicated a high degree of correlation between pre-test Excel expertise and overall spreadsheet expertise. Since the Excel expertise measure had

<table>
<thead>
<tr>
<th>Variable</th>
<th>Overall</th>
<th>Ad hoc design (n=30)</th>
<th>Struct. design (n=58)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>SD</td>
<td>Mean</td>
<td>SD</td>
</tr>
<tr>
<td>Gender (1 Male, 2 Female)</td>
<td>1.38</td>
<td>0.49</td>
<td>1.37</td>
</tr>
<tr>
<td>Age group (1 20–25, 2 26–30, 3 31–40, 4 40–50)</td>
<td>1.18</td>
<td>0.54</td>
<td>1.19</td>
</tr>
<tr>
<td>Class level (1 Jr., 2 Sr., 3 Grad)</td>
<td>1.33</td>
<td>0.54</td>
<td>1.33</td>
</tr>
<tr>
<td>Domain expertise (1 Other, 2 Accounting)</td>
<td>1.93</td>
<td>0.25</td>
<td>1.93</td>
</tr>
<tr>
<td>Treatment group (1 Control, 2 Treat.)</td>
<td>1.66</td>
<td>0.48</td>
<td>1</td>
</tr>
<tr>
<td>Pre-test SS expertise a</td>
<td>2.86</td>
<td>0.75</td>
<td>2.91</td>
</tr>
<tr>
<td>Pre-test Excel expertise a</td>
<td>3.02</td>
<td>0.89</td>
<td>3.00</td>
</tr>
<tr>
<td>Pre-test DFD expertise b</td>
<td>2.35</td>
<td>0.79</td>
<td>2.41</td>
</tr>
<tr>
<td>Pre-test design confidence b</td>
<td>3.11</td>
<td>0.86</td>
<td>3.04</td>
</tr>
<tr>
<td>Post-test SS expertise a</td>
<td>3.56</td>
<td>0.78</td>
<td>3.67</td>
</tr>
<tr>
<td>Post-test Excel expertise a</td>
<td>3.74</td>
<td>0.72</td>
<td>3.89</td>
</tr>
<tr>
<td>Post-test design confidence b</td>
<td>3.25</td>
<td>1.09</td>
<td>3.11</td>
</tr>
<tr>
<td>Post-test coding confidence b</td>
<td>3.08</td>
<td>1.13</td>
<td>2.69</td>
</tr>
<tr>
<td>Post-test error-free confidence b</td>
<td>3.29</td>
<td>1.09</td>
<td>3.44</td>
</tr>
<tr>
<td>Mean number of linking errors/workbook</td>
<td>7.63</td>
<td>9.52</td>
<td>11.56</td>
</tr>
<tr>
<td>Linking error% a</td>
<td>11.05%</td>
<td>0.14</td>
<td>16.75%</td>
</tr>
<tr>
<td>Mean number of conceptual errors/workbook</td>
<td>0.38</td>
<td>0.59</td>
<td>0.37</td>
</tr>
<tr>
<td>Mean number of oversight errors/workbook</td>
<td>1.10</td>
<td>0.84</td>
<td>1.15</td>
</tr>
<tr>
<td>Mean number of total errors/workbook</td>
<td>9.07</td>
<td>9.66</td>
<td>12.96</td>
</tr>
<tr>
<td>Time (h)</td>
<td>19.00</td>
<td>6.62</td>
<td>18.13</td>
</tr>
<tr>
<td>Post-test DFD confidence b</td>
<td></td>
<td></td>
<td>2.98</td>
</tr>
<tr>
<td>Post-test ‘DFD helped’ c</td>
<td></td>
<td></td>
<td>2.84</td>
</tr>
<tr>
<td>Post-test ‘DFD took longer’ c</td>
<td></td>
<td></td>
<td>2.83</td>
</tr>
<tr>
<td>Post-test ‘Will use DFD again’ c</td>
<td></td>
<td></td>
<td>2.88</td>
</tr>
</tbody>
</table>

a 1 Novice, 5 Expert.
b 1 Very unconfident, 5 Very confident.
c 1 Strongly disagree; 5 Strongly agree.
### Table 5
Regression analysis results (Study #2) N = 88

<table>
<thead>
<tr>
<th>Independent variables</th>
<th>Post SS expertise</th>
<th>Post. excel expertise</th>
<th>Post. des. conf.</th>
<th>Post. coding conf.</th>
<th>Post. error free conf.</th>
<th># Missing links</th>
<th># Conc. errors</th>
<th># Oversignt errors</th>
<th># Total errors</th>
<th>Total Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender (1 M, 2 F)</td>
<td>-0.33&lt;sup&gt;a&lt;/sup&gt;</td>
<td>-0.21</td>
<td>-0.30</td>
<td>-0.17</td>
<td>-0.53&lt;sup&gt;a&lt;/sup&gt;</td>
<td>1.04</td>
<td>0.14</td>
<td>0.42&lt;sup&gt;a&lt;/sup&gt;</td>
<td>1.32</td>
<td>-0.87</td>
</tr>
<tr>
<td>Age group</td>
<td>0.21</td>
<td>0.21</td>
<td>0.42</td>
<td>0.32</td>
<td>0.04</td>
<td>0.01</td>
<td>0.36</td>
<td>-0.07</td>
<td>0.15</td>
<td>0.82</td>
</tr>
<tr>
<td>Class level (1 Jr., 2 Sr., 3 Grad.)</td>
<td>-0.13</td>
<td>-0.07</td>
<td>-0.34</td>
<td>-0.31</td>
<td>-0.49&lt;sup&gt;a&lt;/sup&gt;</td>
<td>1.60</td>
<td>-0.18</td>
<td>0.24</td>
<td>1.89</td>
<td>-2.60</td>
</tr>
<tr>
<td>Domain experience (1 -Other, 2 Acctg.)</td>
<td>0.57&lt;sup&gt;a&lt;/sup&gt;</td>
<td>0.38</td>
<td>1.02&lt;sup&gt;b&lt;/sup&gt;</td>
<td>0.24</td>
<td>0.68</td>
<td>2.72</td>
<td>0.17</td>
<td>0.08</td>
<td>2.52</td>
<td>-3.66</td>
</tr>
<tr>
<td>Treatment group (1 Control, 2 Treat)</td>
<td>-0.12</td>
<td>-0.19</td>
<td>0.29</td>
<td>0.58&lt;sup&gt;a&lt;/sup&gt;</td>
<td>-0.15</td>
<td>-5.75&lt;sup&gt;b&lt;/sup&gt;</td>
<td>0.06</td>
<td>0.00</td>
<td>-5.63&lt;sup&gt;b&lt;/sup&gt;</td>
<td>1.03</td>
</tr>
<tr>
<td>Pre. Excel expertise</td>
<td>0.30&lt;sup&gt;b&lt;/sup&gt;</td>
<td>0.29&lt;sup&gt;b&lt;/sup&gt;</td>
<td>0.30&lt;sup&gt;a&lt;/sup&gt;</td>
<td>0.43&lt;sup&gt;b&lt;/sup&gt;</td>
<td>0.06</td>
<td>-3.18</td>
<td>-0.07</td>
<td>-0.15</td>
<td>-3.22&lt;sup&gt;a&lt;/sup&gt;</td>
<td>-1.11</td>
</tr>
<tr>
<td>Pre. DFD expertise</td>
<td>0.14</td>
<td>0.16</td>
<td>0.27&lt;sup&gt;a&lt;/sup&gt;</td>
<td>0.16</td>
<td>0.04</td>
<td>-0.34</td>
<td>0.47</td>
<td>0.02</td>
<td>-0.23</td>
<td>-0.64</td>
</tr>
<tr>
<td>Pre. design confidence</td>
<td>0.08</td>
<td>0.06</td>
<td>0.00</td>
<td>-0.09</td>
<td>-0.02</td>
<td>1.35</td>
<td>0.07</td>
<td>0.03</td>
<td>1.41</td>
<td>-0.28</td>
</tr>
</tbody>
</table>

R<sup>2</sup> 0.32<sup>a</sup> 0.27<sup>b</sup> 0.31<sup>c</sup> 0.22<sup>a</sup> 0.20<sup>a</sup> 0.17<sup>a</sup> 0.21<sup>a</sup> 0.11 0.17<sup>a</sup> 0.09

<sup>a</sup> p ≤ 0.05.  
<sup>b</sup> p ≤ 0.1.  
<sup>c</sup> p ≤ 0.001.
the highest correlation with the dependent variables, it was included in the regression analysis, and the pretest spreadsheet expertise measure was omitted. Table 5 shows the initial regression analysis for all variables. Significant results were attained for gender, pre-test Excel expertise, DFD expertise, and design confidence. A reduced model analysis performed by omitting the non-significant independent variables yielded essentially the same results. Table 6 shows the regression analysis results for the design methodology attitudinal measures obtained from the treatment group subjects.

4.6. Summary and discussion

Apparently our independent variables accounted for significant differences in all of the dependent measures except number of conceptual errors and overall time spent on design and coding. An interesting gender difference surfaced: females reported significantly \((p \leq 0.05)\) less perceived post-test spreadsheet expertise and confidence that their spreadsheets were error-free. This concurs with Harrison and Rainer’s finding that males tend to have more experience with computers and higher confidence levels. Interestingly, the females also had significantly \((p \leq 0.05)\) fewer oversight errors. We postulated that this may have been a result of their lower confidence levels: perhaps the females reviewed their spreadsheets more and caught their oversight errors.

Subjects in higher class levels (i.e., seniors and graduate students) also displayed less confidence that their spreadsheets were error free. There were no notable correlations between class level and any of the other independent variables; in fact, subjects in the higher levels seemed, in general, to have less pre-test application expertise. We attributed this result to the inexperience and overconfidence of youth.

Domain experience was a positive predictor of post-test spreadsheet expertise \((p \leq 0.05)\) and post-test design confidence \((p \leq 0.01)\). Neither of these findings were particularly surprising. Domain experience was somewhat correlated to pre-test spreadsheet expertise \((r = 0.19)\) as well, so apparently the accounting majors had more prior expertise with spreadsheets than the non-accounting majors. The problem was fairly complex, with ten different worksheets and a total of 66 links, so it is reasonable to expect that domain experience would contribute to design confidence.

Using the structured design methodology, the number of link errors reduced significantly \((p \leq 0.01)\), but had no impact on the number of oversight or conceptual errors. Interestingly, link errors were by far the most common. Each spreadsheet (for the combined control and treatment groups) had an average of 9.07 total errors; of these, 84% were linking errors. It is

### Table 6

Regression analysis results for treatment attitudinal measures (Study #2) \(N = 58\)

<table>
<thead>
<tr>
<th>Independent variables</th>
<th>Standardized regression coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DFD conf.</td>
</tr>
<tr>
<td>Gender (1 M, 2 F)</td>
<td>-0.58(^b)</td>
</tr>
<tr>
<td>Age group</td>
<td>-0.04</td>
</tr>
<tr>
<td>Class level(^e)</td>
<td>-0.16</td>
</tr>
<tr>
<td>Domain experience(^f)</td>
<td>0.64</td>
</tr>
<tr>
<td>Pre. SS/Excel expertise</td>
<td>-0.19</td>
</tr>
<tr>
<td>Pre. DFD expertise</td>
<td>0.41(^b)</td>
</tr>
<tr>
<td>Pre. design confidence</td>
<td>0.26</td>
</tr>
<tr>
<td>R(^2)</td>
<td>0.41(^c)</td>
</tr>
</tbody>
</table>

\(^a\) \(p < 0.05\).

\(^b\) \(p < 0.01\).

\(^c\) \(p < 0.001\).

\(^d\) 1 Strongly disagree; 5 Strongly agree.

\(^e\) 1 Jr., 2 Sr., 3 Grad.

\(^f\) 1 Other, 2 Acctg.
important to note that using the design methodology had no significant impact on total time spent on the project (aside from the time taken to learn the methodology).

Pre-test Excel expertise was the most significant predictor, positively impacting post-test spreadsheet expertise \((p \leq .01)\), post-test Excel expertise \((p < 0.01)\), post-test design confidence \((p \leq 0.05)\), and post-test coding confidence \((p \leq 0.01)\). Subjects with higher pre-test Excel expertise also made fewer conceptual errors \((p \leq 0.01)\) and fewer overall errors \((p < 0.01)\). These findings concur with the results of many studies emphasizing the importance of application experience and resulting expertise e.g., [1,27].

The attitudinal responses indicate that females were significantly less confident of their post-test DFD development ability. Pre-test design confidence was positively correlated to the belief that creating the DFD was helpful and would be used in the future. Subjects with a higher degree of pre-test spreadsheet expertise did not think that developing the DFD caused the development effort to take longer. Overall, from an attitudinal standpoint, the subjects were ambivalent of the costs and benefits of the structured design approach.

5. Conclusions, limitations, and future research

Our experimental studies confirm that developer attributes, specifically gender and domain and application expertise, significantly impacted error rates. Additionally, we show that using the structured design methodology significantly improved reliability for two different linked spreadsheet applications. Study #1 also inadvertently suggests that developer configuration is an important factor and has potential interactions with the development approach.

Our initial study was confounded, because the pairs were self-selected. Our second study removed this possible bias. Additionally, our research uses absolute error rates to proxy for the theoretical construct of reliability. It is not clear how error rates in spreadsheets are correlated with poor decisions resulting from spreadsheet models. Thus, a construct validity concern exists. Also, the potential interaction of problem complexity on the structured development approach usage needs to be explored further.

Our theoretical framework identifies several outcomes associated with the use of end-user developed spreadsheets. This research concentrates on outcome reliability. The positive impact of development approach cannot be generalized to usability, maintainability, and/or audibility outcomes without more research. Finally, the structured design approach used explicitly identified links during the design stage. Other methodologies that explicitly specify potential error factors prior to implementation may also improve end-user application reliability.

As end-user development increases in organizations, the identification of strategies to minimize risks and improve outcomes is becoming increasingly important. This research has made a contribution both by identifying end-user development risk factors and outcomes, and by developing and empirically evaluating the impacts of a structured design approach for spreadsheet development.

References


[23] I. Lorge, H. Solomon, Two models of group behavior in the solution of Eureka-type problems, Psychometrika 20(2) (1955) (cited in [29]).
