The Effect of Delocalized Plans on Spreadsheet Comprehension - A Controlled Experiment

Bas Jansen and Felienne Hermans

Report TUD-SERG-2017-009
The Effect of Delocalized Plans on Spreadsheet Comprehension

A Controlled Experiment

Bas Jansen
Delft University of Technology
Email:b.jansen@tudelft.nl

Felienne Hermans
Delft University of Technology
Email:f.j.hermans@tudelft.nl

Abstract—Spreadsheets are widely used in industry. Spreadsheets also suffer from typical software engineering issues. Previous research shows that they contain code smells, lack documentation and tests, and have a long live span during which they are transferred multiple times among users. These transfers highlight the importance of spreadsheet comprehension. Therefore, in this paper, we analyze the effect of the organization of formulas on spreadsheet comprehension.

To that end, we conduct a controlled experiment with 107 spreadsheet users, divided into two groups. One group receives a model where the formulas are organized such that all related components are grouped closely together, while the other group receives a model where the components are spread far and wide across the spreadsheet. All subjects perform the same set of comprehension tasks on their spreadsheet.

The results indicate that the way formulas are located relative to each other in a spreadsheet, influences the performance of the subjects in their ability to comprehend and adapt the spreadsheet. Especially for the comprehension tasks, the subjects perform better on the model where the formulas were grouped closely together. For the adaptation tasks, we found that the length of the calculation chain influences the performance of the subjects more than the location of the formulas itself.

I. INTRODUCTION

The use of spreadsheets in industry is widespread. For millions of employees, spreadsheets form the day-to-day tool to solve business questions, create reports, and deliver support for planning and scheduling activities.

Spreadsheets can be considered a successful end-user programming language. It could also be argued that spreadsheets are more than an end-user programming language and that they are code [1], as there are many similarities with code. Like code, spreadsheets implement concepts like composition, selection and repetition. In spreadsheets, simple objects can be combined into more complex ones by including references to other cells within a cell’s formula, they implement selection with an if-then-else structure and they mimic iterations by the replication of the same formula across many rows or columns.

However, like code, spreadsheets suffer from software maintenance issues. They lack documentation [2], contain code smells [3] and clones [4], have an average lifespan of five years and are used or maintained by an average of twelve different users [2].

There is a growing body of research in which methods of software engineering are transferred to spreadsheets. This research include testing [5], reverse engineering [6], [7], code smells, [3], [8], and refactoring [9], [10].

Because of their average lifetime of 5 years, spreadsheets are transferred multiple times from one user to another. These transfer scenarios stress the importance of spreadsheet comprehension. With respect to source code we know that the concept of delocalized plans or locality is a factor that influences program comprehension. There are several studies that indicate that delocalization negatively correlates with program comprehension [11], [12], [13], [14]. For spreadsheets this concept of locality is also relevant. The advice we can extract from several spreadsheet guidelines with respect to locality is conflicting. Conway and Ragsdale [15] stated in their spreadsheet guidelines that “things which are logically related should be arranged in close physical proximity and in the same columnar or row orientation”. However, the FAST standard [16] prescribes to separate inputs from calculations and to put them on different worksheets. The first approach adheres to the concept of locality, the second approach will lead to delocalized plans. What is the effect of this design choice on the overall comprehensibility of the spreadsheet? It is this question that leads us to the core question of this research paper: What is the effect of delocalized plans on spreadsheet comprehension?

To address this goal, we set up a controlled experiment with 107 spreadsheet users. We create two different versions of a spreadsheet model, that differ in the organization of the formulas within the spreadsheet. In one model the formulas are closely grouped together while in the other model they are spread over multiple worksheets. The subjects are divided over the two models and asked to perform the same set of comprehension tasks. During the experiment, we measure 1) correctness, 2) perceived difficulty, 3) the time to completion, and 4) number of clicks needed for completion of the task.

The results reveal that the existence of delocalized plans in spreadsheets influences the user’s ability to comprehend and adjust the spreadsheet. When users have to execute comprehension tasks on formulas with longer calculation chains they perform significantly better on the model that contained less delocalized plans.

The contributions of this paper are:

• A definition of the concept of delocalized plans in spread-
sheets (Section II)

- Design of a controlled experiment with 107 spreadsheet users to analyze the ability of subjects to comprehend and adjust a spreadsheet (Section III)
- Translation of the software comprehension tasks as defined by Pacione et al. [17] to the spreadsheet domain (Section III)
- An empirical evaluation of the effect of delocalized plans in spreadsheets on spreadsheet comprehension (Section IV)

We organize the remainder of this paper in the following way. In the next section, we provide background information about the concept of delocalized plans in the context of spreadsheets. In Section III we describe the set-up of the experiment followed by a presentation of the results in Section IV and a discussion of the results in Section V. The paper is concluded with an overview of the related work (Section VI) and the concluding remarks (Section VII).

II. DELOCALIZED PLANS IN SPREADSHEETS

A. Delocalized Plans in Source Code

The concept of locality in source code is a well-researched area. Weinberg [11] defines the concepts of locality and linearity and their effect on program comprehension. Locality means that all relevant parts of the program are located closely together. Linearity means that decisions in the program are arranged in a strictly linear sequence. The two concepts are related: arranging decisions in a non-linear sequence often causes non-locality. Locality and linearity can support a programmer with the comprehension and adaption of code.

Letovsky and Soloway [12] reached a similar conclusion. They explore the relation between program comprehension and delocalized plans. A plan is defined as the technique that is used to realize the intention behind the code. A delocalized plan means that the code for the plan is spread far and wide in the source code. In their study, they concluded that delocalized plans are more liable to misinterpretations.

B. Translating Delocalized Plans to Spreadsheets

The concept of delocalized plans can easily be translated to spreadsheets. In the left part (a) of Figure 1 we have highlighted the formula in cell C32. It calculates the total funding a school receives for its entry level students. The formula in cell C32 is:

\[ = C30 \times C31 \]

Cells C30 and C31 are the direct precedents of this formula and are located close to the formula itself (in the same column in the two rows directly above it). However, the calculation chain does not stop there. Both cells C30 and C31, in turn, also contain a formula and they again refer to other cells. To completely understand the calculation you need to trace back not only the direct but also the indirect precedents. They are illustrated in Figure 1 with the blue arrows. The indirect precedents are located somewhat further away. Nevertheless, they still can be presented in a readable manner on a single screen. The complete model spans 14 columns by 32 rows.

In the right part (b) of Figure 1 we illustrate the same calculation, with the same input and output, but in a spreadsheet where the formulas are organized in a different way. The calculation in this model is located in cell C21 and its formula is:

\[ = 019 \times C43 + C50 \]

If we limit our attention to the direct precedents, we already can observe that they are located much farther apart than in the first example. They are located in different columns (O and C) and also the vertical distance is larger. The precedents are located two rows above, 22 rows below, and 29 rows below the formula. If we also include the indirect precedents, the situation deteriorates. It is going to be very difficult to present them all in a readable manner on a single screen. This version of the model spans 16 columns by 104 rows.

However, in a spreadsheet, there is a third dimension. In the previous two examples, the precedents of a formula were all located on the same worksheet. Yet, it is also possible that precedents are located on a different worksheet. Previous research [18] defines the Feature Envy code smell in the context of spreadsheets. A formula suffers from Feature Envy if it makes references to cells on different worksheets. The authors argue that refactoring such a formula by moving it to that different worksheet “will likely improve understandability, since the formula is then closer to the cells it is referring to.”

Letovsky and Soloway use the term delocalized plan if the code for a plan is “spread far and wide in the text of the program”. If we apply this to spreadsheets, a delocalized plan would be a formula that has its precedents spread widely across the spreadsheet. This could mean that the precedents are located far apart on the same worksheet or that they are located on different worksheets. We regard a formula as delocalized if it is impossible to get an overview of all its precedents in a single glance.

Both Letovsky and Soloway and Weinberg argue that source code that tries to avoid delocalized plans is easier to comprehend and adapt. Is this also true for spreadsheets? To answer this question we have designed a controlled experiment. The set-up of this experiment is discussed in the next section.

III. EXPERIMENTAL SETUP

The goal of this paper is to answer the question: Are spreadsheets that contain delocalized plans harder to understand? To address this goal we have formulated the following research questions. How does the existence of delocalized plans in spreadsheets influence the user’s ability to:

- RQ1 understand a component of a spreadsheet?
- RQ2 understand the complete calculation model of a spreadsheet?
- RQ3 adapt a spreadsheet?

The distinction between RQ1 and RQ2 is the level of abstraction. If we would explain the difference between RQ1 and RQ2 in the context of source code, then RQ1 is about code explanation and RQ2 about system understanding [19].

To answer these questions we design a controlled experiment. In the remainder of this section, we explain its set-up.
Fig. 1. Two examples of the same formula. In the left example (a) all precedents are grouped closely together, the size of the model is 14 columns by 32 rows. In the example on the right (b), the precedents are spread far and wide in the model. The size of the model is 16 columns by 104 rows.

A. Subjects

To recruit subjects for the experiment we invite the participants of one of our MOOCs (Massive Open On-line Course), post it on social media and announce it via the mailing lists of EUSPRIG and our own research website. People who are interested are randomly assigned to one of the two models. In total, we received 107 valid responses.

We ask the participants to assess their own Excel skills and how frequently they use Excel on a five-point Likert scale. Table I gives an overview of how the participants are distributed over the two models and the average score for their Excel skills and the frequency in which they use Excel.

<table>
<thead>
<tr>
<th>Model C</th>
<th>Model F</th>
</tr>
</thead>
<tbody>
<tr>
<td># Subjects</td>
<td>56</td>
</tr>
<tr>
<td>Frequency</td>
<td>3.3</td>
</tr>
<tr>
<td>Skill Level</td>
<td>3.6</td>
</tr>
</tbody>
</table>

B. Tasks

We ask each subject to answer nine questions about the spreadsheet. The questions are the same in both models. In order to answer these questions the participants have to analyze the formulas in the spreadsheet, make small changes and correct errors. To obtain a proper set of comprehension tasks we use the framework that was defined by Pacione et al. [17], commonly used for empirical evaluation of code comprehension [20],[19]. They analyzed sets of software comprehension tasks that were suggested in software visualization and comprehension evaluation literature and classified them in nine distinct software comprehension activities. Table II gives an overview of these nine categories.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>Investigating the functionality of (a part of) the system</td>
</tr>
<tr>
<td>A2</td>
<td>Adding to or changing the system’s functionality</td>
</tr>
<tr>
<td>A3</td>
<td>Investigating the internal structure of an artefact</td>
</tr>
<tr>
<td>A4</td>
<td>Investigating dependencies between artefacts</td>
</tr>
<tr>
<td>A5</td>
<td>Investigating runtime interactions in the system</td>
</tr>
<tr>
<td>A6</td>
<td>Investigating how much an artefact is used</td>
</tr>
<tr>
<td>A7</td>
<td>Investigating patterns in the system’s execution</td>
</tr>
<tr>
<td>A8</td>
<td>Assessing the quality of the system’s design</td>
</tr>
<tr>
<td>A9</td>
<td>Understanding the domain of the system</td>
</tr>
</tbody>
</table>

Our questions\(^1\) are designed in such a way that each of them addresses, at least, one of the categories and that all, for spreadsheet relevant categories, are covered. Table III shows for each question in the experiment one or more of Pacione’s activities that are related to the question in the experiment.

\(^1\)Both the questions and the spreadsheet models used in this experiment can be found at: https://doi.org/10.6084/m9.figshare.3167983
### C. The Spreadsheet model

To ensure that all the different comprehension tasks are covered in the experiment we need a spreadsheet of a reasonable size with non-trivial calculations. Furthermore, the experiment should simulate a realistic scenario. For these reasons we choose a spreadsheet that is used in practice by schools to calculate the total funding they receive from the Dutch government, based on the number of students and the number of certificates they issue.

We created two versions of the model, one that adheres to the concept of locality (Model C, precedents are located close to the formula) and one that contains numerous of delocalized plans (Model F, precedents are located far from the formula). The delocalized plans were created by grouping related data on separate worksheets, a practice that is often used by spreadsheet users to structure data in a spreadsheet. We tried to keep the formulas in both models as much the same as possible. However, the layout of the data in a spreadsheet and the structure of the formulas are highly intertwined. Because of this we had to change some of the formulas. Figure 2 is a screen shot of Model C. The blue arrows visualize the precedents that are used in the formula of the active cell. They are all located on the same worksheet in the rows above the formula.

In Figure 3 the same formula is displayed, but now in Model F. The precedents are not located on the same worksheet but spread over four different worksheets.

To inspect the formula the user has to switch back and forth between four different worksheets. This is in contrast to the first example (Figure 2, Model C), in which all precedents are on the same worksheet and because they are located close to the formula itself it is not even necessary to scroll when inspecting the formulas.

In a spreadsheet with delocalized plans formulas make reference to cells that are located far from the formula, often on a different worksheet. The number of external references (to another worksheet) is also used to determine if a spreadsheet is suffering from the Feature Envy smell. To be sure that the two spreadsheets differ sufficiently from each other we analyzed both models on the occurrence of the Feature Envy smell.

As a metric for enviousness we counted all references a formula has to cells contained by the other worksheet which is the same definition that was used in [18]. For both spreadsheets we divided the number of these external references by the total number of references for all unique formulas. Model C scored 0%, Model F 73%.

To determine if one of the models is suffering from Feature Envy we have to establish a threshold for the described metric. We established this threshold by analyzing the distribution of the metric over the Enron Spreadsheet Corpus [21] and used the approach of Alves et al. [22]. In total we analyzed 9,080 spreadsheets with a total of 770,644 unique formulas. 90% of the analyzed spreadsheets had a score for Feature Envy of 16% or less. In accordance with [22] we defined this score of 16% for the metric as a very high risk that the spreadsheet is suffering from the Feature Envy smell. Based on this threshold

<table>
<thead>
<tr>
<th>Question</th>
<th>Task</th>
<th>RQ</th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
<th>A6</th>
<th>A7</th>
<th>A8</th>
<th>A9</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 Explain a calculation</td>
<td>RQ2</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Q2 Find and correct an error</td>
<td>RQ3</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Q3 Adapt a calculation</td>
<td>RQ2</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>Q4 Explain a key concept of the model</td>
<td>RQ2</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Q5 Determine relationship between two cells</td>
<td>RQ1</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Q6 Find dependents of a cell</td>
<td>RQ1</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Q7 Explain how the spreadsheet can be improved</td>
<td>RQ1</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Q8 Assess adaptability of the spreadsheet</td>
<td>RQ3</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Q9 Assess transferability of the spreadsheet</td>
<td>RQ2</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>-</td>
<td>2</td>
<td>-</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
Model F is suffering from Feature Envy and Model C not.

D. Experimental procedure

To participate in the experiment the subjects have to register themselves on the web page of the experiment and after registration, they are randomly redirected to a download link for either Model C or Model F. In the instructions we ask them to answer all questions in one sitting. We added a VBA script to the spreadsheet that logs all activities of the subjects. The script is activated when a participant clicks on a random cell in the spreadsheet. From that moment on, each click, activation of a worksheet, or change to a cell is logged with a timestamp. As soon as a subject closes the workbook the log file is automatically sent to us. In this set-up, we cannot control the subject’s behavior. It is plausible that the subject will be interrupted during the experiment and is not able to complete all the tasks in one sitting. However, using the timestamps we could deduce if there was a continuous time line. From the results, we learned that in 99% of the cases the time between two activities (a click or a change in a cell) is not more than one and a half minute. We, therefore, ignored all gaps between activities that lasted more than one and half minute, to ensure a proper timing of the tasks.

E. Variables and Analysis Procedure

The independent variable in this experiment is the existence of delocalized plans in a spreadsheet. There are several dependent variables. The first is the correctness of the answer. We measure this by checking all answers against our answer model and grading it on a scale of one to ten (where ten is a perfect score).

The second dependent variable is the perceived difficulty of the question. After each question, we ask each subject to indicate on a five-point Likert scale how difficult they thought the question was.

The other dependent variables in the experiment are the time, the number of clicks, and the number of worksheet changes a subject needs to answer a question. We choose to measure the number of clicks because the most common way to analyze a formula in a spreadsheet is to select the cell with the formula, which then can be inspected in the formula bar. Every cell selection leads to a click and gives us information about the formulas the user analyzed during a specific task. The VBA script measures the aforementioned variables. When a subject closes the workbook the associated log file is automatically sent to us by e-mail.

In the next section, we present the results of our experiment and answer our research questions.

IV. Results

As described in the previous section, we analyzed five different variables: correctness, perceived difficulty, time, clicks, and worksheet changes. During the experiment the subjects had to answer nine different questions. Table IV gives an overview per question of the results for these variables.

The results are displayed separately for both spreadsheet models. Model C was designed in adherence to the concept of locality, trying to minimize the number of delocalized plans.
This is in contrast to Model F where the precedents of formulas were spread far and wide in the spreadsheet, leading to a high number of delocalized plans (see for examples Figure 2 and 3 in Section III).

A Shapiro-Wilk test showed that the data did not follow a normal distribution for almost all combinations of questions and models. Therefore, we used the Wilcoxon-Mann-Whitney test to determine per question if there was a significant difference between the two models. Table V shows that for some questions there is a significant difference between the two models.

For each variable with a significant difference we calculated the effect with the Cliff’s Delta (Table VI). For most variables the measured effect was medium ($0.33 \leq d < 0.47$) [23]. Exceptions are the number of clicks for Question 6 where the effect is large ($d \geq 0.47$) and the correctness, where the effect for Q3, Q4, and Q6 is small ($0.15 \leq d < 0.33$).

In the remainder of this section, we will interpret the results and answer our research questions.

### A. RQ1: How does the existence of delocalized plans in spreadsheets influence the user’s ability to understand a component of the spreadsheet?

Pacione considers several levels of abstraction with respect to code comprehension. We see this reflected in the set of comprehension activities (Table II). In our experiment, the questions Q5 and Q6 are related to code comprehension on the component level (see Table III).

In Question 5 we ask the subjects to determine if one cell is a precedent of another cell. An example of such a question is:

‘Answer the following statement with yes or no. The student value for entrance education (result of table IV, cell N24) is used in the calculation of the School student value (cell C28).’

To answer this question the subjects have to inspect the formula in cell C28 to see if it makes a reference to cell N24. For this question, there is no significant difference in performance between the two groups. This in contrast to Question 6 where there is a significant difference. In this case the subjects have to answer the following question (see also Figure 4):

‘Which cells are affected if you change the ‘budget factor student value’ (C74).’

This question is different from Question 5. In Question 5 we provide two components and ask the subjects if they are related. In this question, we provide one component and ask the subjects to identify all components that are directly or indirectly related to this one. To be able to answer this question they have to trace all dependents of this cell. The participants in Model C answered this question significantly faster and needed fewer clicks. Notable is the high number of clicks that participants need to answer this question in Model F. In addition of these clicks they also needed to change frequently between worksheets.

Based on these results we observe that the existence of delocalized plans in spreadsheets does influence the user’s ability to comprehend the relations between components of
the spreadsheet. If there are more delocalized plans, the user needs more time and is less accurate. However, we only see this effect when users are inspecting longer calculation chains.

B. RQ2: How does the existence of delocalized plans in spreadsheets influence the user’s ability to understand the complete calculation model of the spreadsheet?

Question 1 and Question 4 test to which degree the subject understands the workings of the model. As stated in the previous section, the model calculates the total funding a school receives given the number of students. The main component of the funding is the student value. It is depending on the number of students but has a different value. In Question 1 we ask the subject to explain why the student value is higher than the total number of students. To answer this question they have to analyze a set of formulas and the relations between them.

For both models, the correctness of the answer did not show any difference. However, on average it took the participants in Model F more time and they needed more clicks than the participants in Model C. Also the Model F participants thought that the question was more difficult. In Model C all cells used to calculate the student value could be inspected on a single worksheet without the need for scrolling. In Model F the participants had to switch between two worksheets. It is plausible that the switching between sheets and the fact that all precedents of the formula could not be inspected on a single screen is responsible for the more time that was needed, the higher number of clicks, and a higher level of perceived difficulty.

In Question 4 we also asked the participants to explain the working of the model. An important concept in the model is the cascade factor. It is a weighting factor that ensures that the school gets less funding for students that need more time to finish their education. We ask the participants to explain the cascade factor in their own words. Although there is no significant difference in the time and number of clicks needed to answer this question, the quality of the answers in model C is significantly higher (See Table V).

From these results, we conclude that the existence of delocalized plans in a spreadsheet influences the user’s ability to comprehend the complete calculation model of the spreadsheet. However, we also observe that although we see a difference for both questions, the results differ per question. In Question 4 there is a difference in the correctness of the answers and in Question 1 it is the timing, perceived difficulty and the number of clicks. In future research, we will design a follow-up experiment to gain better insight into the causes for these differences.

In both Question 7 and 9, the subjects are asked to assess the quality of the spreadsheet. In Question 7 they have to suggest improvements to the model, whereas in Question 9 they are asked to assess how easy it is to explain the working of the spreadsheet to somebody else. Both questions are related to the user’s ability to understand the spreadsheet.

It is notable that there is no significant difference, for both questions, in the subject’s assessment of the quality of the spreadsheet, whereas we do observe a difference in the performance when subjects have to execute a comprehension task in Q1 and Q4.

We also received contradicting advice about how to improve the spreadsheet. A subject working with Model C (that consisted of a single worksheet) suggested to: “split the long sheet to several sheets and name them by what they calculate.”, while a subject working with Model F (consisting of multiple
worksheets) wrote that: “having all the inputs on one sheet would be helpful (to prevent having to scroll trough tabs all the time)

In both groups, there were several subjects that suggested that the spreadsheet would be easier to comprehend if named ranges were used (replacing cell addresses by meaningful names). In future work, we will research the effect of using named ranges on formula comprehension.

C. RQ3 How does the existence of delocalized plans in spreadsheets influence the user’s ability to adapt the spreadsheet?

In both Question 2 and 3, the subjects are asked to make adaptations to the model. In Question 2 they have to find and correct an error. For this question, we do not see any significant difference between the two groups. The subjects knew beforehand which part of the spreadsheet contained the error. To find the error it was not necessary to trace all the precedents, inspecting the formula was sufficient. This explains why the organization of the formulas in the spreadsheet did not influence the subjects ability to correct the error.

In Question 3 we asked the participants to change the model. Each school receives additional funding for disabled students. In our model, it takes into account both the number of students at entry level as well as the other students. In the question, we asked to exclude the entry level students from the calculation. In this case, the results indicate that it was easier to make this change in Model F than in Model C. For model F, the participants needed fewer clicks and submitted a higher number of correct answers. In Model C all components of the formula were located on a single sheet. In Model F the components were split between two worksheets. In order to make the change, the subjects needed to change sheets several times. Nevertheless, the performance in Model F was better.

There are two possible explanations. Although in Model F, the components of the formula were spread over two different sheets, on both sheets it was possible for the subjects to see all information in a single glance. Whereas in Model C all information was located on one sheet, but not visible in a single glance. We suspect that subjects had to scroll to see the complete picture. So one explanation could be that the ability to see all information in a single glance is responsible for the better performance.

Another explanation could be the formula itself. In order to make the change in Model C, the subjects needed to make changes in more than one cell, whereas in Model F it was sufficient to make a change in only one cell.

We designed a small second experiment to determine which of these two explanations is more plausible. We provided fourteen employees of the financial staff of Delft University of Technology with an advanced Excel training. During this training we let them do an exercise that was designed to determine which of the aforementioned explanations is the most plausible.

In this experiment, for both models, it was not possible to see all information in one glance and scrolling was necessary. The only remaining difference was the way the formula was constructed. The fourteen employees were randomly assigned to either Model C or Model F. The results of this experiment are summarized in Table VII.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Model C</th>
<th>Model F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correctness</td>
<td>8.1</td>
<td>10</td>
</tr>
<tr>
<td>Difficulty</td>
<td>3.2</td>
<td>3.4</td>
</tr>
<tr>
<td>Time</td>
<td>8:15</td>
<td>4:40</td>
</tr>
<tr>
<td>Clicks</td>
<td>40</td>
<td>9</td>
</tr>
</tbody>
</table>

Again the performance was better in Model F. All eight subjects that were assigned to Model F completed the exercise flawlessly. In this model it was sufficient to change just a single cell, making the way the formula was constructed the most plausible explanation for the better performance.

This finding also sheds light on a previously unresolved issue regarding the trade-off between the long method and the long calculation chain smell in spreadsheet formulas as discussed in [3] and [24]. Defining a short formula to avoid the long method smell inevitably leads to a longer calculation chain. Trying to reduce the long calculation chain smell will lead to longer formulas. Based on the findings in this paper it appears that reducing the length of the calculation chain has more impact on maintainability of a formula than the length of the formula itself. This contradicts the FAST [16] guideline to write short formulas. A short formula is maybe easier to understand, but it leads to longer calculation chains and the experiment shows that users make more errors when they have to make adjustments to the logic in the calculation chain. In future research, we will conduct an experiment with a larger test group to further validate these conclusions.

The last question that is related to the user’s ability to adapt the model is Question 8. In this question subjects are asked to assess how easy it would be to adapt the spreadsheet. The results for this question do not show a significant difference between the two models.

The results do not provide evidence that the existence of delocalized plans influences the user’s ability to adapt the spreadsheet. However, they do indicate that regardless the existence of delocalized plans, the length of a calculation chain does influence the user while adapting the spreadsheet. If there is a longer calculation chain, the user needs more clicks to make the change in the model and is more likely to make errors.

V. DISCUSSION

A. Threats to validity

There are several threats to the internal validity of this study. The first threat is the subject’s skill level with respect to Excel. To mitigate this risk we asked the subjects to rate how frequently they use Excel and how they assess their skill level on a five-point Likert scale. Table I shows that for both
variables there is no significant difference between the two groups.

The second threat is that the subjects were aware of the goals of the Study. To mitigate this we referred to the goals of the study in very general terms in both the invitation to participate and the instruction (for example: ‘We are very much interested in how people are interacting with spreadsheets’). Furthermore, the log file we used to analyze the subject’s interaction with the spreadsheet was sent to us without showing it to the subject.

A third threat to the internal validity is the threat of self-selection. We invited participants of a MOOC about spreadsheets, approached mailing list members of a spreadsheet interest group (EUSPRIG), and members of our own mailing list. Because of this, it is plausible that all subjects have a more than average interest in the subject of spreadsheets and are not representative of the total population of spreadsheet users. Nevertheless, we decided to approach these possible subjects to maximize the set of participants. And although these subjects have possibly a more than average interest in spreadsheets it is also more likely that they use spreadsheets in their daily activities.

A final threat to the internal validity could be a learning effect. We mitigated this by allowing subjects to participate only once in the experiment. If we received multiple log files from the same e-mail address, only the first submission was accepted for the experiment. Nevertheless, a learning effect also occurs because the subject acquires more knowledge of the model while answering the questions. We decided not to randomize the sequence of questions to mitigate this effect. Pacione [17] considers several levels of abstraction with respect to code comprehension and the sequence of our questions has been structured accordingly.

A threat to the external validity of the experiment could be the representativeness of the spreadsheet model that we used in the study. We mitigated this risk by using a spreadsheet model that is based on a model that is used in practice by professionals.

Also, the representativeness of the subjects is an external threat. This risk is reduced by the relatively large sample size of 107 subjects. They varied in their age, cultural background, and Excel skill level and were randomly assigned to one of the two models.

B. Effect of long calculation chains

When we started this study, we expected to find a clear relation between the existence of delocalized plans in spreadsheets and the user’s ability to comprehend the spreadsheet. We did find such a relation, but it is more nuanced than we thought. It appears that also the length of a calculation chain is an important factor that influences spreadsheet comprehension. Furthermore, the type of comprehension task seems to be relevant. Subjects perform significantly better in spreadsheets without delocalized plans when they have to explain the working of the model. However, when we ask them to adapt the model, it is the length of the calculation chain that determines their performance and not the existence of delocalized plans.

C. Long Calculation Chain smell versus Multiple Operations and Multiple References smells

In previous research on code smells in spreadsheet formulas it is stated that there is a trade-off between the Long Calculation Chain smell and the Multiple Operations and Multiple References smells [3], [24]. Our results indicate that the Long Calculation Chain smell has a higher impact on the user’s ability to comprehend and maintain a formula than the Multiple Operations and Multiple References smells. It is easier for a user to comprehend and adapt a long formula with a short calculation chain than a short formula with a long calculation chain. This finding contradicts the popular belief that short formulas are easier to comprehend. For example, the FAST Standard Organization advises in their popular FAST standard to write short formulas: “A formula longer than your thumb likely means that it should be broken into more than one step.” [16].

VI. Related Work

A. Software Engineering Methods and Spreadsheets

There are numerous studies that focus on applying software engineering methods on spreadsheets. Rothermel et al. brought the concept of testing to spreadsheets with the What You See is What You Test paradigm [5]. Other researchers focused on the domain of reverse engineering and came up with methods to extract class diagrams from spreadsheets [6] [7] or to visualize the data flow within spreadsheets [25]. Fowler introduced the concept of smells in code [26], but smells also exist in spreadsheets. They are described in detail in the work of Hermans [27] [18], Cunha [8] and Barowy et al. [28]. From smells, it is a small and logical step to refactoring. Hermans defined refactorings for formula smells in spreadsheets [9]. Inspired by this work Badame and Dig developed RefBook, a tool that supports a number of refactorings for spreadsheet formulas [10].

B. Delocalized Plans and Program Comprehension

Weinberg defines several principles for programming language design [11]. According to the author, a properly designed language aids the programmer by keeping relevant information close at hand. In this context, he defines the concepts of locality and linearity. Locality means that all relevant parts of a program are found in the same place, for example on the same page or on a single screen. Linearity means that the decisions in the program are arranged in a strictly linear sequence.

Letovsky and Soloway [12] studied the relation between locality and program comprehension. They state that the goal of program understanding is to recover the intentions behind the code. They define a plan as the technique that is used to realize an intention. In their paper they focus on so-called delocalized plans, meaning that the code for the plan is not closely grouped, but spread far and wide in the source
code. They found that in order to understand a program, programmers make reasonable but sometimes incorrect assumptions. They tend to leave these assumptions unverified if the effort required to verify the assumption is great. Typically in delocalized plans, it will take more effort to verify the assumptions. According to the authors, these plans are more liable to misinterpretation than plans whose code is closely grouped. Delocalized plans could also be defined as plans with data flow links spanning widely separate parts of the code. Providing the developer with a data flow analyzer that makes these links explicit could reduce the risk of misinterpretations.

Constantine [14] took a different approach. He focused on the concept of localization in relation to user interface design. He defines the metric Visual Coherence that measures how the arrangement of visual components matches with their semantic relationships. He found that professional developers preferred more visually coherent designs and thought they were easier to use.

C. Controlled Experiments in Software Engineering

Other research discusses the set-up of experiments to analyze software comprehension. Pacione et al. [17] introduce a software visualization model for supporting object-oriented software comprehension. They evaluate the performance of visualization tools by assessing their performance in typical software comprehension tasks. Based on previous work and evaluation tasks found in software comprehension and software visualization literature, they introduced a framework of nine principal comprehension activities (see also Table II in Section III). In the current paper, we use this framework to make sure that all relevant program comprehension activities were covered in our experimental setup.

A similar set-up of a controlled experiment in software engineering was used by Cornelissen et al [20]. The authors use a controlled experiment to evaluate the effectiveness of a tool for the visualization of large traces and also apply Pacione’s comprehension framework to select their comprehension tasks for the experiment.

VII. Concluding Remarks

The goal of this paper is to answer the question: Are spreadsheets that contain delocalized plans harder to understand?

To address this goal, we set up a controlled experiment using two spreadsheets that differ in the organization of the formulas within the spreadsheet. In the experiment, the subjects are divided over the two models and perform a number of comprehension tasks, of which we measure correctness, perceived difficulty, completion time and the total number of clicks.

The results reveal that the existence of delocalized plans in spreadsheets influences the user’s ability to comprehend and adjust the spreadsheet: subjects perform significantly better on the model that contained less delocalized plans when they have to perform comprehension tasks on formulas with longer calculation chains.

The contributions of this paper are:

- A definition of the concept of delocalized plans in spreadsheets (Section II)
- Design of a controlled experiment to analyze the ability of subjects to comprehend and adjust a spreadsheet (Section III)
- Translation of the software comprehension tasks as defined by Pacione et al. [17] to the spreadsheet domain (Section III)
- An empirical evaluation of the effect of delocalized plans in spreadsheets on spreadsheet comprehension (Section IV)

A. Future work

This paper triggers several ideas for follow-up research. Firstly, a more extensive experiment is needed to understand the impact of delocalized plans in spreadsheets in more depth. For example, with a larger controlled experiment using multiple spreadsheets, followed by a longitudinal study. Furthermore, we envision a tool that measures the occurrence of delocalized plans in spreadsheets. Such a tool could support users in estimating the effort needed for maintaining a spreadsheet.

In this experiment, references in formulas were presented in ‘A1’ style. However, there are alternatives like ‘R1C1’ notation or the use of named ranges for references. In a future study, we will investigate the effect of these different forms of notation on formula comprehension.

REFERENCES


