Detecting and Visualizing Inter-Worksheet Smells in Spreadsheets

Hermans, FFJ; Pinzger, M; van Deursen, A

DOI
10.1109/ICSE.2012.6227171

Publication date
2012

Document Version
Accepted author manuscript

Published in
Proceedings of the International Conference on Software Engineering (ICSE)

Detecting and Visualizing Inter-worksheet Smells in Spreadsheets

Felienne Hermans, Martin Pinzger, Arie van Deursen
Software Engineering Research Group
Delft University of Technology
Delft, the Netherlands
{*f.f.j.hermans,m.pinzger,arie.vandeursen*}*@tudelft.nl*

***Abstract*—Spreadsheets are often used in business, for simple tasks, as well as for mission critical tasks such as finance or forecasting. Similar to software, some spreadsheets are of better quality than others, for instance with respect to usability, maintainability or reliability. In contrast with software however, spreadsheets are rarely checked, tested or certified. In this paper, we aim at developing an approach for detecting *smells* that indicate weak points in a spreadsheet's design. To that end we first study code smells and transform these code smells to their spreadsheet counterparts. We then present an approach to detect the smells, and to communicate located smells to spreadsheet users with data flow diagrams. To evaluate our apporach, we analyzed occurrences of these smells in the Euses corpus. Furthermore we conducted ten case studies in an industrial setting. The results of the evaluation indicate that smells can indeed reveal weaknesses in a spreadsheet's design, and that data flow diagrams are an appropriate way to show those weaknesses.**

***Keywords*-spreadsheets; code smells; refactoring; data flow diagrams**

I. Introduction

Spreadsheets are widely used in industry: Winston [25] estimates that 90% of all analysts in industry perform calculations in spreadsheets. Their use is diverse, ranging from inventory administration to educational applications and from scientific modeling to financial systems. Especially in the financial domain spreadsheets are prevailing. Panko [23] states that 95% of U.S. firms, and 80% in Europe, use spreadsheets in some form for financial reporting.

Business analysts using spreadsheets usually have very limited training in programming or structuring data. In spite of that, they effectively are *end-user programmers*, and as such face many of the challenges of professional developers, such as identifying faults, debugging, or understanding someone else's code [16].

This paper aims at providing support for spreadsheet users to take on these end-user programming challenges, focused on support for identifying potentially risky parts in a spreadsheet's high level *design*, i.e. the way in which worksheets are organized and depend on each other.

Our starting point is the code smell metaphor introduced by Fowler [9]. In particular we study the coupling and cohesion of classes (called Collaboration Disharmonies by Lanza and Marinescu [17]) and transform these code smells in such a way that they apply to the coupling of worksheets rather than classes. This leads to a list of *inter-worksheet smells*.

In order to detect these smells automatically, we define metrics for each of them. With each metric, we establish a threshold, to know at what point to identify a worksheet as smelly. We follow the approach of [2] who analyzed metrics on source code and set thresholds by determining the worst 70, 80 and 90% of all methods and choosing corresponding metric values for the thresholds representing medium, high and very high risk.

We then address the issue of communicating identified smells to spreadsheet users. For this we use our original worksheet visualization, since earlier experiments have shown that these diagrams are useful for spreadsheet users to get an overview of the structure of their spreadsheet. We enrich those data flow diagrams with colors and tool tips to convey the inter-worksheet smells to spreadsheet users.

We perform a quantitative and qualitative evaluation of our approach. Firstly, we analyzed the Euses corpus, and investigated the occurrence of inter-worksheet smells. Secondly, we conducted a series of case studies at a large Dutch financial institution, called Robeco. Here we conducted ten case studies. In each study we analyzed a real-life spreadsheet, and discussed the located smells with the spreadsheet owner, supported by the generated data flow diagram. The evaluations aim to answer the following research questions:

- R_1 What inter-worksheet smells are the most common, and why?
- R_2 How do inter-worksheet smells expose threats to spreadsheet quality and calculation integrity?
- R_3 To what extent are enriched data flow diagrams an appropriate way of visualizing inter-worksheet smells?

The findings of these evaluations indicate that (1) inter-worksheet smells are commonly found in real-life spreadsheets, (2) inter-worksheet smells can indicate real weaknesses in a spreadsheet's design and (3) data flow diagrams seem useful to help spreadsheet users locate and understand inter-worksheet smells.

This paper is organized as follows. Section II gives an overview of related work in the area of code smells and

978-1-4673-1067-3/12/$31.00 © 2012 IEEE

ICSE 2012, Zurich, Switzerland

spreadsheet metrics. In Section III we sketch the background of our approach and illustrate it with a motivating example. Section IV introduces the inter-worksheet smells, followed by Section V that explains how to automatically detect the smells. In Section VI the communication of the smells to spreadsheet users by means of data flow diagrams is described. Section VII describes the implementation of our prototype Breviz, while Section VIII explains the two evaluations we have performed in detail. Section IX discusses the applicability of the proposed approach, followed by the concluding remarks in Section X.

II. Related Work

Efforts related to our research include work on code smells, starting with the canonical work by Fowler [9]. His book gives an overview of code smells and corresponding refactorings. Recent efforts focused on the automatic identification of code smells by means of metrics. Marinescu [19] for instance, uses metrics to identify *suspect* classes, those classes that might have design flaws. Lanza and Marinescu [17] explain this methodology in more detail. Alves *et al.* [2] focus on a strategy to obtain thresholds for metrics from a benchmark. Olbrich *et al.* furthermore investigates the changes in smells over time, and discusses their impact [21].

Furthermore, there are papers that address common errors in spreadsheets. Ayalew *et al.* [3] for instance names the incorrect grouping of data, and incorrect coupling between those groups as sources of errors. Panko [22] names omission errors as the most dangerous of spreadsheet errors. Omission errors are those errors that arise from "misinterpretation of the situation that should be modeled" (Powell *et al.* [24]). In our own previous work [12] we have developed an approach to visualize the flow of data within spreadsheets by means of a data flow diagram. The evaluation showed that those diagrams are useful for spreadsheet professionals that need to understand the structure of a spreadsheet. In other recent work [11] we have applied code smells to individual formulas, rather than worksheets.

Finally, there are papers on spreadsheet metrics, which are also aimed at locating weak points in spreadsheets. In 2004, Bregar published a paper presenting a list of spreadsheet metrics based on software metrics [4]. He however does not provide any justification of the metrics, nor did he present an evaluation. Hodnigg and Mittermeir [13] propose several spreadsheet metrics of which some are similar to Bregar's. Their metrics are divided into three categories: general metrics, such as the number of formulas and the number of distinct formulas; formula complexity metrics, such as the number of references per formula, and the length of the longest calculation chain; and finally metrics, such as the presence of scripts in, e.g., Visual Basic for Applications (VBA), user defined functions and external sources. Besides the metrics, the authors also pose the interesting suggestion to use different types of visualizations for cells with different values for the metrics. Hole *et al.*[14] propose an interesting approach to analyze spreadsheets in terms of basic spreadsheet metrics, such as the number of functions used, the presence of charts and the complexity of program code constructs to predict the level of the spreadsheet creator.

We have combined the work on the definition and detection of code smells with existing spreadsheet work, to obtain a list of inter-worksheet smells that can be automatically detected.

III. Background & Motivating Example

In previous work [12] we created an approach for generating leveled data flow diagrams from spreadsheets, to support professionals in explaining their spreadsheets to colleagues. Figure 1 depicts an example of such a data flow diagram, showing all worksheets within a spreadsheet and the formula relations between them. Rounded squares represent worksheets, and an arrow between worksheet A and worksheet B means that formulas in worksheet B refer to cells in worksheet A. The thickness of the arrow indicates how many unique formulas connect A and B. From this figure we can learn that the spreadsheet in question contains five worksheets, and that, for example formulas in 'exam' refer to cells in worksheet 'result79813'.

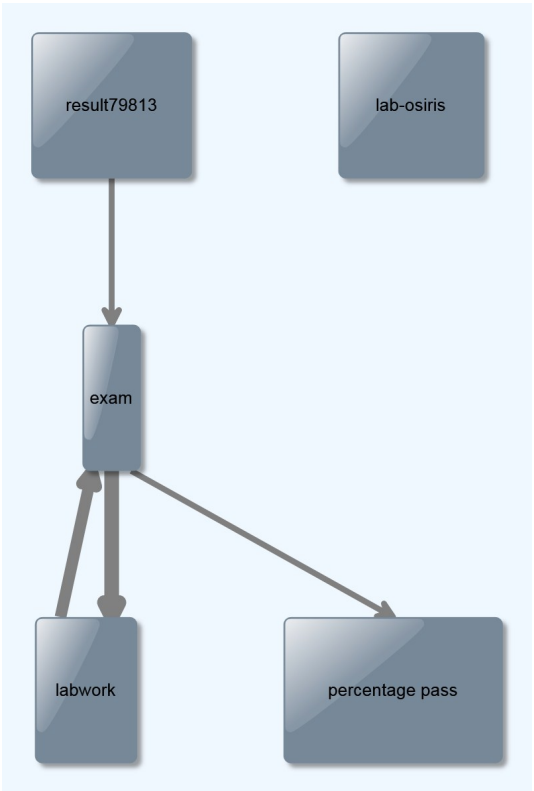

Figure 1. A leveled data flow diagram, representing data flow in a spreadsheet

We have implemented the generation of leveled data flow diagrams, and tested our approach at Robeco, a Dutch investment bank [12]. While we observed users working with the data flow diagrams, we noticed that they often used the diagrams as a means of assessing the quality of the spreadsheet it represented. A spreadsheet leading to a 'spaghetti' diagram was considered of lower quality than a diagram that looked very structured.

This observation led to the idea of using our diagrams to assess spreadsheet quality, which we will elaborate on in the remainder of this paper. We will investigate the hypothesis that the global view of our leveled data flow diagrams can also serve as a means of identifying weak points, or even flaws in a spreadsheet's design.

The following example demonstrates this. Figure 1 shows the data flow visualization of a real-life spreadsheet that is used by a university professor to calculate the grade for a course. It consists of five worksheets. From the diagram some aspects of the spreadsheet immediately catch the eye. For instance, one of the worksheets *lab-osiris* is not connected to the other sheets. This sheet contains the data from Osiris, the university's grading system. This information can be important when working with the spreadsheet, since the spreadsheet user might mistakenly think that the exam scores are automatically updated when the information in Osiris is updated. To determine this without Breviz would require the user to select all cells for all worksheets one by one and checking their dependents. Furthermore, the loop between *exam* and *labwork* stands out. The name *exam* could suggest that the worksheet only contains data about the exam, however it apparently also contains information regarding the lab work. This example illustrates the type of information that can be gathered from a data flow diagram generated from a spreadsheet.

We found, in observing users at Robeco working with Breviz that the questions raised by looking at the diagram, help spreadsheet users assess the quality of their spreadsheet.

IV. Inter-worksheet Smells

In this section we look at Fowler's code smells [9] and investigate which of the code smells can be adapted in such a way that it applies to spreadsheets. We focus on inter-class code smells, since they can be related to the inter-worksheet smells. We also leave out the code smells that involve inheritance—Parallel Inheritance Hierarchies and Refused Bequest—since that concept does not directly occur in spreadsheets.

A. Inappropriate Intimacy

This smell indicates that a class has too many dependencies on implementation details of another class. A related spreadsheet smell would be a worksheet that is overly related to a second worksheet. This is possibly unhealthy for several reasons. First, adapting one sheet likely requires inspecting the other worksheet, requiring the spreadsheet user to switch back and forth between the two worksheets, increasing the chance that errors are made [20]. Secondly, since there is a strong semantic connection between the two worksheets, the fact that they are split could influence understandability.

B. Feature Envy

Feature Envy is the phenomenon where a method M seems more interested in the fields of another class than of the class that contains M. In general, Fowler suggests to put a method in the class that contains most of the data the method needs. This code smell seems very applicable to spreadsheets: if there is a formula that is more interested in cells from another worksheet, it would be better to move the formula to that worksheet. This will likely improve understandability, since the formula is then closer to the cells it is referring to. Conway and Ragsdale [6] stated in their spreadsheets guidelines that "things which are logically related should be arranged in close physical proximity and in the same columnar or row orientation". If the result of the formula is needed in other worksheets, it could still be linked to the necessary sheets after it is moved.

C. Middle Man

Fowler defines a middle man as a class that delegates most of its operations to other classes, and does not contain enough logic to justify being a separate class. When this occurs, it might be time to refactor out the middle man.

This smell could also occur in spreadsheets, where 'middle man' formulas occur, that contains only a reference to another cell and calculations, like the formula '=Sheet1!A2.' If a worksheet contains many of those middle man formulas, it might be better to remove this worksheet, and move its functionality to other worksheets.

A worksheet suffering from the Middle Man smell could complicate the structure of a spreadsheet, and therefore reduces spreadsheet quality. Many papers on spreadsheets stress the importance of structure in a spreadsheet's design. Janvrin and Morrison [15] for instance, state that using a structured design approach reduces risks in end-user spreadsheet development. Markus and Keil agree and further note that structured design methodologies increase the accuracy and reliability of information systems [18]. Cheney *et al.* [5] found that the more structured an end-user works, the more likely the success of the application.

D. Shotgun Surgery

Source code smells of 'shotgun surgery' when one change results in the need to make a lot of little changes in several classes. One of its common forms is a method A that is referred to by several other methods in several different classes. When A is changed, it is likely that also the callers of A will have to be changed, resulting in a lot of changes at different places.

The spreadsheet translation of this code smell is a formula F that is referred to by many different formulas in different worksheets. By the same logic, the chances are high that many of the formulas that refer to F will have to be changed if F is changed.

Shotgun Surgery could have an impact on the maintainability of a spreadsheet, since it requires the user to make a number of changes when F is changed.

V. Detecting Inter-worksheet Smells

In this section, we present our approach to automatically detect worksheet smells in spreadsheets. We base our approach on existing work done in the domain of object-oriented software engineering, such as [2], [17], [21]. We mainly follow the approach by Marinescu described in [19]: for each of the smells we define one or more metrics that indicate the presence of that particular smell. We subsequently analyze a large body of spreadsheets and set thresholds by determining the worst 70, 80 and 90% of all worksheets and choosing corresponding metric values for the thresholds representing medium, high and very high risk.

Definitions: To be able to reason about spreadsheets and smells, we define the following types, sets and functions.

Cell The type C represents a cell in a spreadsheet.

Worksheet W represents a worksheet in a spreadsheet, and is defined as a set that contains all cells that are located in the worksheet.

Spreadsheet S represents a spreadsheet, and is defined as a set that contains all worksheets contained in the spreadsheet.

Precedents Precedents P is a function of type $C \rightarrow \{C\}$ representing all precedents of a given cell. Precedents are the cells that a formula refers to.

Connection A connection K is a tuple (A,B) of two cells. Two cells are called connected if $A \in P(B)$.

Connection Set The set KS is the set containing all connections of a spreadsheet.

A. Inappropriate Intimacy

To detect Inappropriate Intimacy we investigate the amount of data coupling between two different worksheets. We count the number of connections between two worksheets, which we call the *Intimacy* of these worksheets that is of type $W \times W \rightarrow$ int and is defined as follows

$$\text{Intimacy}(w_0,w_1) \equiv |\{(c_0,c_1 \in KS : c_0 \in w_0 \wedge c_1 \in w_1 \wedge w_0 \neq w_1\}|$$

We count the number of pairs in KS where the cell c_0 is contained by worksheet w_0 and the cell c_1 is contained by worksheet w_1 and the two worksheets are not the same. Note that this definition implies that if there are two formulas on worksheet y referring to the same cell in x, we count this as two connections rather than one.

To get the intimacy for one worksheet, we take the maximum intimacy the worksheet has with any of its connected worksheets

$$II(w_0) \equiv \max\{\text{Intimacy}(w_0, w_1) : w_0,w_1 \in S\}$$

B. Feature Envy

The Feature Envy smell is actually a smell that applies to a formula rather than to a worksheet. We adhere to the approach of Olbrich *et al.* [21] who state that when a method has a certain smell, by extension the class that contains it is also smelly.

To detect the Feature Envy smell on worksheets, we inspect all its formulas and check the cells that they are 'interested in'; the cells that they refer to. As a metric for enviousness, we count all references a formula has to cells contained by the other worksheets of a spreadsheet. Hence the definition of *Enviousness(FE)*, a function of type $C \rightarrow$ int is

$$FE(c_0) \equiv |\{(c_0,c_1) \in KS : \exists w: c_0 \in w \wedge c_1 \notin w\}|$$

We count the number of pairs from the connection set where cell c_0 is contained by w but not the cell c_1.

C. Middle Man

The Middle Man smell is detected when a worksheet is mainly used to pass values to another worksheet. To detect this smell we use the definition of a special type of formulas, the *middle man* formula. A middle man formula does not contain any operations besides the $=$ operation that gets a value from another cell. The function MMF: $C \rightarrow$ bool indicates whether a formula is a middle man formula.

When there is a calculation chain that contains two consecutive passing formulas, there is risk of the Middle Man smell. We therefore count the number of middle man formulas in a worksheet that are referred to by another middle man. This makes *Middleman (MM)* a function of type $W \rightarrow$ int, defined as

$$MM(w) \equiv |\{(c_0,c_1) \in KS: c_1 \in w \wedge MMF(c_0) \wedge MMF(c_1)\}|$$

D. Shotgun Surgery

We count the number of connected formulas that are both middle man formulas. To detect shotgun surgery in worksheets, we take our inspiration from the method described by Lanza and Marinescu [17]. This method entails counting *changing methods*; the number of distinct methods that call a method A of the class and *changing classes*; the number of classes in which the methods that

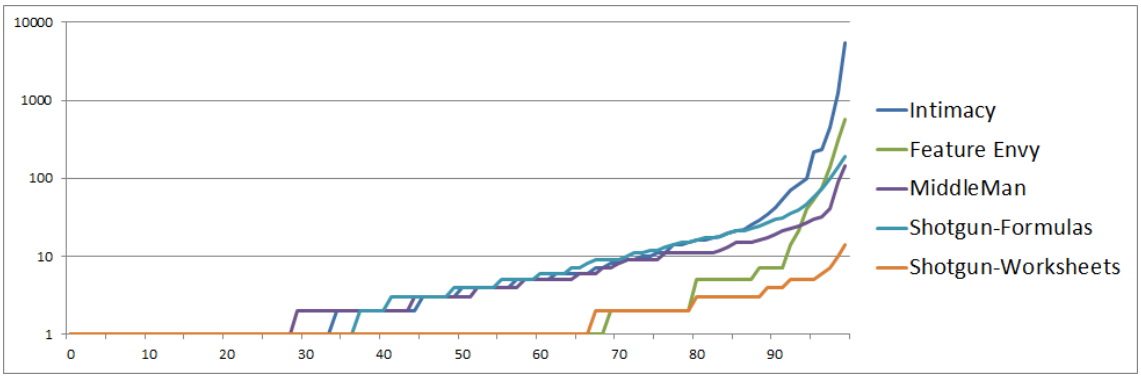

Figure 2. The distribution of the various metrics on a quartile graph with logarithmic scale on the y-axis

call the measured method are defined. These methods and classes are called *changing*, because they are the methods and classes likely to change if method A changes. We adapt this method to make it applicable to spreadsheets, and introduce the following definitions, both of type $W \rightarrow int$

Changing Formulas The number of formulas that refer to a formula in worksheet w, defined as

$$\text{ChangingFormulas(w)} \equiv |\{(c_0,c_1)\in KS : c_0 \notin w \wedge c_1 \in w\}|$$

Changing Worksheets The number of worksheets in which the changing formulas lie, defines as

$$\text{ChangingWorksheets}(w_0) \equiv |\{w_1 \in S: (\exists (c_0,c_1)\in KS : c_0\ w_1 \wedge c_1 \in w_0)\}|$$

E. Determining the Thresholds

Now we have determined the metrics for each of the smells, we establish the thresholds for each of the metrics. We establish the thresholds by analyzing the distribution of the metric values over a large, representative body of spreadsheets. We follow the approach of Alves *et al.* [2]. They inspect the percentage of metric values below a certain given percentage, and set the thresholds accordingly.

The body of spreadsheets we use is the Euses Spreadsheet Corpus [8]. This corpus contains real-life spreadsheets from all sorts of domains, ranging from educational to financial, and from inventory to biology. It was created in 2005 and has since then been used by several researchers to evaluate spreadsheet algorithms, among which [1] and [7].

The corpus comprises of 4,223 spreadsheets, which together contain 15,015 worksheets. Of those spreadsheets, there are 1,711 spreadsheets that contain formulas, divided over 4,250 worksheets. Figure 2 shows the distribution of all metrics over the worksheets in a quartile graph.

As can been seen in this figure, the metrics all follow a power law like distribution, having most of their variability on the tail. Feature Envy and Shotgun Surgery-Worksheets only have values higher than one above 65% of the worksheets, and the other three metrics pass the value of 10 at around 70%.

Because the metrics follow a distribution similar to the metrics in [2], we use the same method for setting the thresholds for our metrics. We therefore choose 70%, 80% and 90% as the thresholds for medium, high and very high risk that a worksheet suffers from an inter-worksheet smells as introduced above. Table I shows the thresholds of the five metrics for the four smells.

Table I
THRESHOLDS FOR THE METRICS DETERMINING THE SMELLS

Smell	70%	80%	90%
Inappropriate Intimacy	8	16	42
Feature Envy	3	5	7
Middle Man	7	11	19
Shotgun - Changing Formulas	9	16	30
Shotgun - Changing Worksheets	2	3	4

VI. VISUALIZING INTER-WORKSHEET SMELLS

Once we have established the preliminary code smells, there is the question how to communicate the smells to spreadsheet users. Since we have found in earlier work that data flow diagrams are very suitable to explain the structure of a spreadsheet, we will use them to convey the inter-worksheet smells as well. For this paper we adapted the diagrams by enriching them with information about the inter-worksheet smells. When a smell is located, the worksheet box in the dataflow diagram is colored yellow, orange or red, for smells at the 70%, 80% or 90%. A tool tip explains the spreadsheet user what smell is located, and what cells contribute to this smell.

Colors and tool tips are simple and well-known user interface aspects, hence this seems an effective way of indicating code smells. Figure 3 depicts the user interface of our tool Breviz, with the spreadsheet smell indicators,

for a spreadsheet from the Euses corpus. As this figure shows, smells are located in two worksheets, 'Capital Exp' and 'Cost-Quarterly'. The tool tips indicate what smells are located, Feature Envy in 'Capital Exp' and Inappropriate Intimacy in 'Cost-Quarterly'. The tool tips furthermore name the cells that contribute to these smells.

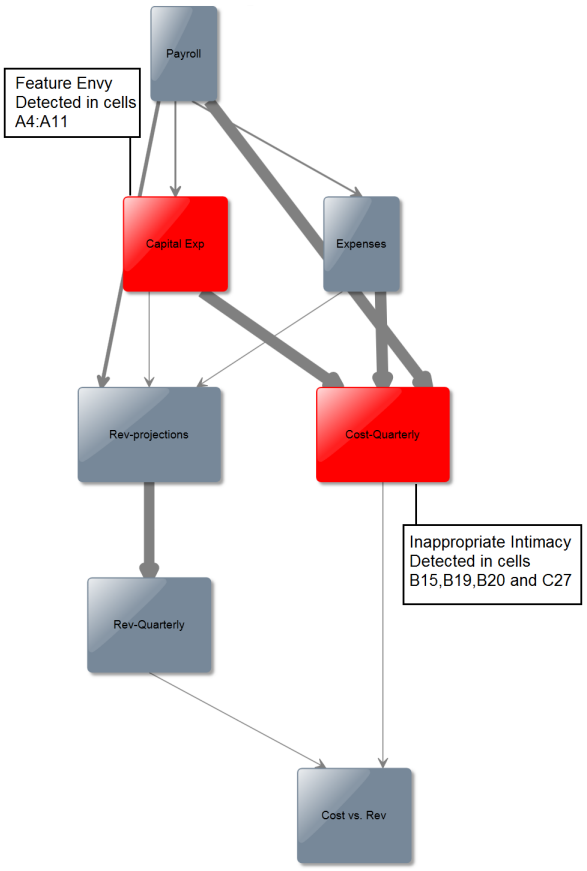

Figure 3. A screenshot of a spreadsheet data flow visualization enriched with smell information

VII. Implementation

The enriched data flow visualizations were incorporated into our existing spreadsheet analysis system Breviz [12]. Breviz is implemented in C# 4.0 using Visual Studio 2010. It utilizes the Gembox component to read Excel files[1] and visualizes the data flow diagrams using YFiles for .NET/WPF[2]. Breviz, which is currently implemented as stand-alone program, reads an Excel file, writes the metrics described in Section IV to a SQL Server 2008 database and subsequently generates an annotated data flow diagram.

[1]http://www.gemboxsoftware.com/spreadsheet/overview
[2]http://www.yworks.com/en/products_yfilesdotnet_about.html

VIII. Evaluation

In this section we will explain how our inter-worksheet smells and their visualization were evaluated. With the evaluation aim to answer the research questions. To do so, we performed two separate evaluations, both a quantitative and a qualitative evaluation. In the quantitative evaluation we analyzed the occurrence of inter-worksheet smells in the Euses Spreadsheet corpus, given the thresholds we have selected. With this evaluation, we focused on research question R_1.

For the qualitative evaluation, aimed at more understanding of R_1, plus answers to R_2 and R_3, we asked 10 professional spreadsheet developers for access to their real-life spreadsheets. We let our tool Breviz identify possible code smells and showed the user the enriched data flow visualization. We subsequently asked the spreadsheet users to reflect on the identified inter-worksheet smells.

The following subsections describe the two evaluations in more detail.

A. Inter-worksheet Smells in the Euses Corpus

1) Goal: During the first evaluation we want to learn more about the occurrence of the four inter-worksheet smells, and hence focus on the question what smells are most common(R_1).

2) Setup: In this evaluation we used the Euses Spreadsheet Corpus. As stated above, this corpus consists of 4,223 real-life spreadsheets, from all sorts of domains, ranging from educational to financial, and from inventory to biology.

For each of the four inter-worksheet smells, we checked how many spreadsheets contain worksheets with metric values above the 70%, 80% and 90% thresholds. This gives an overview of the distribution of the smells over the spreadsheets.

3) Results: Table II shows the results of the first evaluation. As can be seen in this table, Feature Envy is the most common smell, with 12.4% of the spreadsheets containing worksheets with metric values above the 70% threshold.

Feature Envy is followed by Inappropriate Intimacy, with 9.6% above the 70% threshold. Feature Envy and Inappropriate Intimacy are related, since Feature Envy can be the cause of Inappropriate Intimacy.

The observation that those two smells are common is consistent with previous work, in which we have seen that it is difficult for spreadsheet creators—that are usually not trained as programmers—to structure their spreadsheets in a logical way [12].

We believe these smells pose a real threat to spreadsheet understandability, since in previous user studies it has been shown that it is difficult for spreadsheet users to work with the connection between multiple worksheets. Nardi and Miller [20] for instance found that "it is difficult to get a global sense of the structure of a spreadsheet. That requires tracing the dependencies among the cells." We

will investigate these smells in more detail in the second evaluation.

Third most common is the Middle Man smell, with which 5.9% of the spreadsheets are diagnosed. Finally, with 4.1% of the spreadsheets suffering from Shotgun Surgery at the 70% level, this is the least common smell in the Euses corpus.

Table II
PERCENTAGE OF SPREADSHEETS IN THE EUSES CORPUS THAT CONTAINS AT LEAST ONE WORKSHEET THAT SUFFER FROM AT LEAST ONE SMELL ABOVE THE 70, 80 AND 90% THRESHOLDS.

Smell	**>70%**	**>80%**	**>90%**
Feature Envy	12.4 %	8.7%	5.8%
Inappropriate Intimacy	9.6%	6.8%	4.2%
Middle Man	5.9 %	5.0%	4.0%
Shotgun Surgery	4.1 %	3.3%	1.5%
Any of the above smells	23.3 %	17.6%	12.8%

B. Inter-worksheet Smells in Ten Real-Life Case Studies

1) Goal: The objective of the second evaluation is to answer the "why" of research question R_1, as well as to provide the answers to R2 and R3.

2) Setup: For the second evaluation we gathered 10 professional spreadsheets from the financial domain. We performed our evaluation research at Robeco, a Dutch asset management company with approximately 1600 employees worldwide, and over €130 billion worth of assets under management. In a survey we conducted among 27 of their analysts, we found that they use Excel for an average of 3 hours a day, underlining the importance of spreadsheets in their daily work. Furthermore, we found that spreadsheets have an average lifetime of more than five years, and individual spreadsheets are used by 13 different analysts on average [12].

We invited participants of this previous survey to participate in this evaluation. We asked them to select one of their large and complex spreadsheet, which they worked with often. Ten subjects responded positively and participated in this evaluation. The ten subjects were familiar with our data flow diagram visualization, since all of them participated in our previous study [12].

Before the case studies started we provided subjects with information about the setup. We explained participants that we were going to identify inter-worksheet smells that could possibly indicate weak points in their spreadsheets. We furthermore told them we wanted to discuss the smells with them in a short interview. Finally, we provided subjects with a list of the four inter-worksheet smells and a short explanation of the smells, so they could study this before the experiment.

For each of the ten subjects and their spreadsheet, the procedure was as follows:

First, we asked the subjects to explain the purpose and context of the spreadsheet. We then generated the annotated data flow diagram. Subjects studied the data flow diagram and the corresponding spreadsheet, for a maximum of 10 minutes, after which the interview part of the study started. In this part of the study, we asked the subject for each of the located inter-worksheet smells:

- Do you understand why this part of your spreadsheet is marked as potentially risky?
- Do you agree that there is indeed a problem or an error here?
- Can you explain why you structured the spreadsheet like this?
- Does the data flow visualization help you find the smells?

With these questions we analyzed whether the smells and their impact are recognized by spreadsheet users, and investigate their causes. We furthermore learned about the reception of the data flow diagrams.

Table III shows an overview of the characteristics of the spreadsheets used in the case study. As can be seen in this table, the spreadsheets are of considerable size, with as many as 25 worksheets. Seven of the ten spreadsheets suffered from at least one of the inter-worksheet smells, and Inappropriate Intimacy is the most common smell among the spreadsheets.

3) Results:

General Observations: In all the ten case studies, we noticed that the subjects were surprised when confronted with the data flow diagrams. They often expressed statements such as "are those worksheets really that connected?" or even "are you sure that arrow is correct?". These experiences corroborate what we found in a previous study [12]. Since the arrows in the data flow diagram are thicker when more formulas are connecting two worksheets, subjects could easily see that two worksheets were strongly connected. We noticed that subjects found the data flow visualization to be helpful in understanding the smells. One of the subjects stated "that arrow is so fat. That can't be good".

The popups then helped to explain what smell exactly was found. We found the addition of the smelly cells in the popup to be of great value as well. This way users could locate the smells within the worksheet, and determine how the spreadsheet could be improved.

Inappropriate Intimacy: When interviewing the subjects about the Inappropriate Intimacy smell, we were again struck by how difficult it is for spreadsheet users to understand dependencies between the worksheets. In all seven cases where Inappropriate Intimacy was found, it took users time to understand what cells were connecting the worksheets and why, stating e.g. "what did I do here again" and "I don't remember why I needed to connect those sheets". Even when supported by the data flow diagrams,

Table III
CHARACTERISTICS AND NUMBER OF SMELLS ABOVE THE 70% THRESHOLDS OF SPREADSHEETS USED IN THE TEN CASE STUDIES.

ID	Spreadsheet Description	#Worksh.	#Cells	#Form.	#Un. Form.	Size(Kb)	II	FE	MM	SS
1	Calculate dividend	5	13,697	6,012	53	183	2	-	-	-
2	Overview of investment strategies	5	21,600	3,031	98	605	3	2	1	3
3	Model companies in the energy sector	14	82,000	14,742	531	826	2	7	2	6
4	Valuation of swaps	8	31,485	5,034	67	1,690	1	3	-	-
5	Overview profit and loss of all traders	10	17,263	9,152	142	4,261	-	-	-	-
6	Overview of risk for different sectors	9	9,190	148	12	332	-	-	-	-
7	Comparing different calculation models	14	24,580	3,388	39	348	5	3	5	-
8	Planning of trades	1	2,329	1,683	64	76	-	-	-	-
9	Report interest rate and liquidity risk data	25	59,116	17,930	117	1,693	8	6	-	4
10	General ledger data for analysis	11	11,906	3,049	56	1,732	3	3	-	-

they were searching for the reason that a strong dependency was detected, and why they constructed the spreadsheet in that way. This was caused mainly because the spreadsheets did not have the right documentation to support users in these type of questions. While some spreadsheet contained documentation that explained how to work with the spreadsheet, none of the spreadsheets contained information on the *design decisions* of the spreadsheet.

Asking the subjects whether they understood why the Inappropriate Intimacy smell was found, all responded positively. However not all of them agreed that the smell was indeed harmful. We recognized two patterns causing for Inappropriate Intimacy. One is the use of an 'auxiliary' worksheet, in which data is stored, in combination with another worksheet in which this data is referred to. This construction is similar to a join on two tables. This is often implemented using a VLOOKUP, MATCH or IF function.

A second case, deemed more dangerous by subjects, involves two worksheets referring to each other without a clear distinction between the two worksheets. In this case, we witnessed subjects making statements along the lines of "it would in fact be better to merge these two worksheets". In future work, we plan to investigate these two types of intimacy in more detail.

The data flow diagrams were helpful when identifying the smells. They helped the subjects to see what worksheets where connected. One of the subjects stated: "a red block draws my attention, and the thick arrow indicates where there is too much connection".

However, especially in the second case, where two worksheets were overly intertwined, users looked in the spreadsheet, to investigate the formulas in more detail.

Feature Envy: A formula suffers from Feature Envy at the 70% level when at least 3 of its references are located in another worksheet than the formula itself is in.

When a formula refers to a number of cells that lie in a different worksheets, understanding the formula becomes harder. This is due to the fact that modern spreadsheet programs such as Excel highlight the cells that a formula refers to when a formula is selected, but only those cells that are located on the same worksheet as the selected formula. The highlighting feature is used extensively by spreadsheet users, but it does not provide any help when analyzing references across different worksheets. The above contributes to the fact that in all cases in which we discovered Feature Envy, subjects agreed that a refactoring would help, since they recognized the difficulty of analyzing a formula with many references lying in a different worksheet. One subject stated about a formula in his spreadsheet, which referred to no less than 8 cells in a different worksheet: "this formula annoys me, I have to go back to the other sheet so many times to look up the references, it makes me dizzy".

Performing the calculation on the worksheet where the referents are, and than referring to the result from the worksheet where the result is needed, seemed more healthy to all subjects.

In the case of Feature Envy, the data flow diagrams also supported the identification of the smell. However, in this case subjects needed to dig deeper into the spreadsheet formulas to understand why they were marked as envious. While the data flow diagrams list the cells that are smelly, subjects often felt the need to inspect the worksheet, and select the listed cells, to inspect and judge their smelliness.

Middle Man: While performing the interviews with spreadsheet owners whose spreadsheets suffered from the Middle Man smell, we were surprised to see that there was one case in which middle man formulas were passing on information *within* the worksheet. We had not anticipated this use of middle man formulas, and we also did not find this kind of Middle Man smell in the spreadsheets of the Euses corpus.

When we asked the subject why these values were passed within the worksheet, he answered that "it is easy to always have the value in sight when working on the spreadsheet, because then I can see the values I am calculating with". While explaining this he realized that maybe the worksheet had grown too big to still be easy to work with. The subject

then came up with a 'refactoring', such as splitting the functionality of the worksheet in two, or 'locking' the values needed in a row or column. As such, the Middle Man smell within a worksheet can be a smell indicating the worksheet has grown too big.

A second, related category of Middle Man occurrences happens when a value is passed along worksheets to be in sight everywhere. This value passing can be implemented in two different ways. Either it is passed from the source sheet to each worksheet individually, for instance from worksheet A to worksheet B, and from worksheet A to worksheet C. This does not cause the Middle Man smell.

However, the value passing can also be done in a chain. This happens when values are passed from worksheet A to worksheet B, and the same values are then passed from worksheet B to worksheet C. This second situation does cause the Middle Man smell, and is risky for several reasons. Firstly, it is not imminent what the source of the data is, since from worksheet C, it looks like the data is coming from worksheet B, and inspection of worksheet B is needed, to see that the data stems from worksheet A. Secondly, it is possible that the 'chain' gets broken somewhere. In this case, Middle Man smell can be a reason to change the structure of the spreadsheet such that each worksheet gets its needed input directly from the source worksheet.

In most cases where we saw a formula value being passed, it was to have the value in sight. It is interesting that the Middle Man smell actually reveals a weakness in the implementation of current spreadsheet systems, since apparently there is a large need among spreadsheet users to have some of their values from other parts of the spreadsheet in sight when working on it. Since there is no feature to do this, the problem is solved with formulas, which might make the spreadsheet bigger and more difficult to understand.

In the case of Middle Man smells, the three subjects found the data flow diagrams a good way to indicate inter-worksheet smells. All three quickly understood what data was being passed, and why a worksheet was marked as Middle Man. As one of them stated "The arrows in the diagram exactly show where data is being passed, so I can judge the structure of the spreadsheet".

Shotgun Surgery: In the cases where Shotgun Surgery was observed, all three subjects agreed with the fact that there was a shortcoming in the design of their spreadsheet.

In the most extreme case (spreadsheet 9) there was a worksheet on which 220 formulas depended, spread over 10 worksheets. When faced with this, the subject answered that he understood it would be extremely difficult for someone to modify this worksheet, since it involved checking those ten worksheets, and possibly also modifying them. In other words, he foresaw the Shotgun Surgery that would be necessary in order to adapt this worksheet. Another reason that Shotgun Surgery exposes a threat to the spreadsheet quality occurred in spreadsheet 3, where 7 worksheets depended on one single worksheet with 97 formulas. It turned out that the spreadsheet had been revised recently, and that some of the references were no longer actually in use. In this case too the subjects felt the need to change the spreadsheet when seeing the annotated data flow diagram, to make it up-to-date and more easy to understand for others. One of the subjects stated "I should really take some time to improve these formulas, since this is already confusing for me, so it must be terrible for others". In the third case there were 3 worksheets depending on the smelly worksheet, far less than in the other two cases. However even in this case the spreadsheet owner agreed that the current structure could be improved to increase understandability.

In the cases with Shotgun Surgery we observed the same reception to the data flow diagrams as with the Feature Envy smell: initially it is useful to identify the smells. However in order to know exactly what causes the smell, spreadsheet owners wanted to inspect the worksheet themselves.

C. Conclusions

With the results of the Euses analysis and the case studies, we revisit the research questions.

R_1 **What inter-worksheet smells are the most common, and why?** In the first evaluation, Feature Envy is the most frequent smell, and Inappropriate Intimacy comes second. The second evaluation with 10 industrial spreadsheets confirmed these results.

As we have stated before, we believe these two smells are both due to the fact that it is difficult for end-user programmers to create the right abstraction for their worksheets. This calls for an explicit visualization of the design, as provided by our dataflow diagrams.

R_2 **How do inter-worksheet smells expose threats to spreadsheet quality and calculation integrity?**

In the second evaluation, we have learned that the inter-worksheet smells can indeed reveal weaknesses in spreadsheets. Most notable was the fact that upon occurrence of the Shotgun Surgery smell, subjects came up with adaptations of the spreadsheet structure themselves, because they recognized the dangers that the smell was posing.

R_3 **To what extent are enriched data flow diagrams an appropriate way of visualizing inter-worksheet smells?** In general, the ten participants found the dataflow diagrams a good way to indicate inter-worksheet smells. Since the data flow diagram shows the relation of all worksheets, it was easy for the subjects to understand what worksheet was marked as smelly and why. However, in some cases, especially when investigating Feature Envy and Shotgun Surgery, users wanted to see details, and felt the need to open the spreadsheet and inspect the marked formulas.

In future work we plan to address these limitations, such as to link the data flow diagram visualization with the spreadsheet. This allows the user to navigate from the

diagram to the interesting cells in the worksheets and vice versa.

IX. Discussion

Our current approach to finding inter-worksheet smells helps spreadsheet users to understand the weaknesses in their spreadsheets design. In this section, we discuss a variety of issues that affect the applicability and suitability of the proposed approach.

A. VBA Code, Pivot Tables and Charts

In this stage of the research, we have only taken formulas into account when calculating the spreadsheet metrics to detect the smells. Spreadsheets can contain other constructions as well, such as pivot tables, charts and VBA code. Those too might be *smelly*, lowering the quality of the spreadsheet as a whole. In future work we will turn our attention to detection of smells in other constructs of spreadsheets.

B. Mental Model of a Spreadsheet

Both in this paper and in previous work we have found that spreadsheet users often do not have the correct mental model of the structure of a spreadsheet. When they were shown the visualization of the data flow diagram, they were often wondering about certain arrows. We plan to deepen this research in a future study, for instance, by having users draw the mental model of a spreadsheet, and comparing that to the as-implemented model. In that way we can learn why their is often a difference in a user's perception of the worksheet relationships.

C. Understanding Spreadsheet Design Decisions

While some spreadsheets in practice contain a manual or documentation on how to use the spreadsheet, we so far have not found spreadsheets containing the design decisions that were taken while building the spreadsheet. In the qualitative evaluation we have seen that it is hard for spreadsheet users to recover this design information from memory. While our data flow diagrams support in learning the current state of the spreadsheet, it does not help in understanding why the spreadsheet was constructed the way it is. In future work we will investigate this fact, by storing the history of a spreadsheet, and supporting spreadsheet users in documenting changes they make.

D. Threats to Validity

A threat to the external validity of our evaluation concerns the representativeness of the Euses Corpus spreadsheet set. This set, however, is large (containing over 4000 spreadsheets), and is collected from practice. A second threat to the external validity of our evaluation concerns the representativeness of the selected set of the spreadsheets in the employees at Robeco and their spreadsheets. However other papers [10], [23] report on industrial spreadsheet stories similar to the ones we found at Robeco, so their practice seems to be representable.

Furthermore, there is a risk of aptitude treatment interaction since all of the 10 participants where involved in a previous evaluation of data flow diagrams, and it could be the case that only the most positive ones responded to our request to participate in this study.

With respect to internal validity, one of the threats is the fact that we did not pick a random sample. This effect can be decreased by using a larger test group in future experiments. We however believe the current test group serves as a good reference group, as the persons varied in their age, function and daily tasks with spreadsheets.

X. Concluding Remarks

The goal of this paper is to underline the importance of inter-worksheet smells as a means to assess and improve spreadsheet quality. To that end we have revisited literature on code smells and spreadsheet design. This has resulted in a list of inter-worksheet smells, which we have subsequently evaluated in both a quantitative study on the Euses corpus and a qualitative evaluation with ten professional spreadsheet users and real-life spreadsheets.

The key contributions of this paper are as follows:

- The definition of four inter-worksheet smells, based on known code smells (Section IV).
- An approach for the automatic detection of the inter-worksheet smells (Section V).
- An implementation of that approach into our spreadsheet analysis toolkit Breviz (Section VII).
- A twofold evaluation of the proposed inter-worksheet smells, first on the Euses corpus, and secondly with 10 professional spreadsheet users in an industrial context (Section VIII).

The current research gives rise to several directions for future work. Firstly there is the expansion of the smells analysis to other spreadsheet concepts, such as pivot tables, graphs and VBA code. Secondly it would be interesting to examine the relation between actual errors in spreadsheets and inter-worksheet smells. Could smell detection have prevented those errors?

Finally, code smells are inseparably connected to refactoring. Hence it would be exiting to try to create refactoring strategies and refactoring tools for spreadsheet systems, and test their applicability with professional spreadsheet users.

REFERENCES

[1] R. Abraham and M. Erwig. Inferring templates from spreadsheets. In *Proceedings of the 28th International Conference on Software Engineering (ICSE 2006)*, pages 182–191. ACM, 2006.

[2] T. L. Alves, C. Ypma, and J. Visser. Deriving metric thresholds from benchmark data. In *26th IEEE International Conference on Software Maintenance (ICSM 2010)*, pages 1–10. IEEE Computer Society, 2010.

[3] Y. Ayalew, M. Clermont, and R. T. Mittermeir. Detecting errors in spreadsheets. In *Proceedings of EuSpRIG 2000 Conference*, pages 51–62, 2000.

[4] A. Bregar. Complexity metrics for spreadsheet models. In *Proceedings of EuSpRIG 2004 Conference*, page 9, February 2004.

[5] P. Cheney, R. I. Mann, and D. L. Amoroso. Organizational factors affecting the success of end-user computing. *Journal of Management Information Systems*, 3:65–80, July 1986.

[6] D.G. Conway and C.T. Ragsdale. Modeling optimization problems in the unstructured world of spreadsheets. *Omega*, 25(3):313 – 322, 1997.

[7] J. Cunha, J. Saraiva, and J. Visser. Discovery-based edit assistance for spreadsheets. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 2009)*, pages 233–237. IEEE, 2009.

[8] M. Fisher and G. Rothermel. The EUSES spreadsheet corpus: A shared resource for supporting experimentation with spreadsheet dependability mechanisms. In *Proceedings of the Workshop on End-User Software Engineering*, pages 47–51. ACM, 2005.

[9] M. Fowler. *Refactoring: improving the design of existing code*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[10] D. G. Hendry and T. R. G. Green. Creating, comprehending and explaining spreadsheets: a cognitive interpretation of what discretionary users think of the spreadsheet model. *International Journal of Human-Computer Studies*, 40(6):1033–1065, 1994.

[11] F. Hermans, M. Pinzger, and A. van Deursen. Helping spreadsheet users find and improve complex formulas. Technical Report TUD-SERG-2011-030, Delft University of Technology, http://www.se.ewi.tudelft.nl/, 2011.

[12] F. Hermans, M. Pinzger, and A. van Deursen. Supporting professional spreadsheet users by generating leveled dataflow diagrams. In *Proceeding of the 33rd international conference on Software engineering (ICSE 2011)*, pages 451–460. ACM Press, 2011.

[13] K. Hodnigg and R.T. Mittermeir. Metrics-based spreadsheet visualization: Support for focused maintenance. In *Proceedings of EuSpRIG 2008 Conference*, page 16, 2008.

[14] S. Hole, D. McPhee, and A. Lohfink. Mining spreadsheet complexity data to classify end user developers. In *Proceedings of The 2009 International Conference on Data Mining*, pages 573–579. CSREA Press, 2009.

[15] D. Janvrin and J. Morrison. Using a structured design approach to reduce risks in end user spreadsheet development. *Information & Management*, 37(1):1–12, 2000.

[16] A.J. Ko, R. Abraham, L. Beckwith, A. Blackwell, M.M. Burnett, M. Erwig, C. Scaffidi, J. Lawrence, H. Lieberman, B.A. Myers, M.B. Rosson, G. Rothermel, M. Shaw, and S. Wiedenbeck. The state of the art in end-user software engineering. *ACM Computing Surveys*, 43(3), 2010.

[17] M. Lanza and R. Marinescu. *Object-Oriented Metrics in Practice*. 2006.

[18] M. Lynne and M. Keil. If We Build It, They Will Come: Designing Information Systems That People Want to Use. *Sloan Management Review*, 35(4), 1994.

[19] R. Marinescu. Detecting design flaws via metrics in object-oriented systems. In *Proceedings of TOOLS*, pages 173–182. IEEE Computer Society, 2001.

[20] B. Nardi and J. Miller. The spreadsheet interface: A basis for end user programming. In *Proceeding of Tthe IFIP Conference on Human-Computer Interaction (INTERACT)*, pages 977–983. North-Holland, 1990.

[21] S. Olbrich, D. S. Cruzes, V. Basili, and N. Zazworka. The evolution and impact of code smells: A case study of two open source systems. In *Proceedings of International Symposium on Empirical Software Engineering and Measurement*, pages 390–400, Los Alamitos, CA, USA, 2009. IEEE Computer Society.

[22] R. Panko. What we know about spreadsheet errors. *Journal of End User Computing*, 10(2):15–21, 1998.

[23] R. Panko. Facing the problem of spreadsheet errors. *Decision Line*, 37(5), 2006.

[24] S.G. Powell, K.R. Baker, and B. Lawson. Errors in operational spreadsheets: A review of the state of the art. In *Proceedings of the 42nd Hawaii International Conference on System Sciences (HICCS 2009)*, pages 1–8. IEEE Computer Society, 2009.

[25] W.L. Winston. Executive education opportunities. *OR/MS Today*, 28(4):8–10, 2001.