How Are Spreadsheet Templates Used in Practice: A Case Study on Enron

Liang Xu, Wensheng Dou*, Jiaxin Zhu, Chushu Gao, Jun Wei, Tao Huang
University of Chinese Academy of Sciences, China
State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, China
{xuliang12, wsdou, zhujiaxin, gaochushu, wj, tao}@otcaix.iscas.ac.cn

ABSTRACT

To reduce the effort of creating similar spreadsheets, end users may create expected spreadsheets from some predesigned templates, which contain necessary table layouts (e.g., headers and styles) and formulas, other than from scratch. When there are no explicitly predesigned spreadsheet templates, end users often take an existing spreadsheet as the instance template to create a new spreadsheet. However, improper template design and usage can introduce various issues. For example, a formula error in the template can be easily propagated to all its instances without users' noticing. Since template design and usage are rarely documented in literature and practice, practitioners and researchers lack understanding of them to achieve effective improvement. In this paper, we conduct the first empirical study on the design and the usage of spreadsheet templates based on 47 predesigned templates (490 instances in total), and 21 instance template groups (168 template and instance pairs in total), extracted from the Enron corpus. Our study reveals a number of spreadsheet template design and usage issues in practice, and also sheds lights on several interesting research directions.

CCS CONCEPTS

• **Applied computing** → Spreadsheets • **Software and its engineering** → Software reverse engineering

KEYWORDS

Spreadsheet, template, empirical study

ACM Reference format:

Liang Xu, Wensheng Dou, Jiaxin Zhu, Chushu Gao, Jun Wei and Tao Huang. 2018. How Are Spreadsheet Templates Used in Practice: A Case Study on Enron. In *Proceedings of the 26th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engi-neering (ESEC/FSE'18), November 4–9, 2018, Lake Buena Vista, FL, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3236024.3264834

*Corresponding author

ESEC/FSE '18, November 4–9, 2018, Lake Buena Vista, FL, USA

ACM ISBN 978-1-4503-5573-5/18/11…$15.00
https://doi.org/10.1145/3236024.3264834

	B	C	D	Z	AA
45	Hours	HE1	HE2	HE24	**TOTALS**
46	Generation Deals	0	0	0	**=SUM(C46:Z46)**
47	EES Buys	0	0	0	**=SUM(C47:Z47)**
48	EES Sells	0	0	0	**=SUM(C48:Z48)**
49	Transmission to SP-15	0	0	0	**=SUM(C49:Z49)**
50	SP-15 Load	0	0	0	**=SUM(C50:Z50)**
51	Enpower Deals	0	0	0	**=SUM(C51:Z51)**
52	**Variance**	**=SUM(C46:C51)**	**=SUM(D46:D51)**	**=SUM(Z46:Z51)**	**=SUM(AA46:AA50)**

(a) Blank (Template)

	B	C	D	Z	AA
45	Hours	HE1	HE2	HE24	**TOTALS**
46	Generation Deals	0	0	0	**=SUM(C46:Z46)**
47	EES Buys	0	0	0	**=SUM(C47:Z47)**
48	EES Sells	0	0	0	**=SUM(C48:Z48)**
49	Transmission to SP-15	20.22	19.74	20.93	**=SUM(C49:Z49)**
50	SP-15 Load	-20.22	-19.74	-20.93	
51	Enpower Deals	0	0	0	
52	**Variance**	**=SUM(C46:C51)**	**=SUM(D46:D51)**	**=SUM(Z46:Z51)**	**=SUM(AA46:AA50)**

(b) 10-3 (Instance)

Figure 1. A real-world template and one of its instances.

1 INTRODUCTION

Spreadsheets, as one of the most successful end-user programming platforms, are widely used to perform various tasks, such as financial reporting, data storage and data analyses [20].

End users usually develop a group of similar spreadsheets to perform similar business tasks. For example, the monthly financial reports in a company usually share the same table structures and computations. To speed up the preparation of such similar spreadsheets, end users may prepare semi-finished spreadsheets with some predefined formulas and table layouts, and then create new spreadsheets based on these semi-finished spreadsheets. In this paper, we use *predesigned templates* to denote this kind of semi-finished spreadsheets, and instances to denote the spreadsheets created based on the predesigned templates. Fig. 1 shows a typical predesigned spreadsheet template and one of its instances, which is extracted from the Enron corpus [13]. Note that, predesigned templates usually do not contain actual data. For example, in the predesigned template of Fig. 1a, cells C46:Z51 are filled with the default value (e.g., 0). In some cases, end users may not predesign spreadsheet templates, and they choose an existing spreadsheet as the template, delete the original data, and create new spreadsheets based on the spreadsheet. In this paper, we use *instance templates* to denote the spreadsheet instances which are used to create new spreadsheets.

There are a number of benefits of using spreadsheet templates. First, the predefined table layouts and formulas can greatly reduce users' effort in creating new spreadsheets. Second, all the instances created through the same template usually share the unified layout, which makes it easy for end users to process their spreadsheets with other external programs.

However, using spreadsheet templates also results in many issues in practice. 1) The impact of design flaws in the predesigned and instance templates could be large and wide. For example, users may need to make some common changes when creating new instances based on an incomplete predesigned template. 2) Worse, if a template contains errors, then these errors can be propagated to its instances without users' noticing. For example, in the template of Fig. 1a, the formula in cell AA52 is "=SUM(AA46:AA50)", but it should be "=SUM(AA46:AA51)". This error was propagated to the instance shown in Fig. 1b. 3) Incorrect usage of templates may introduce errors. In the instance of Fig. 1b, the user deleted the formulas in cells AA50 and AA51 unintentionally, and this makes the spreadsheet error-prone.

Existing research on spreadsheets mainly focuses on bug detection [2][6][8][10][14][15], structrue analysis [7][9][16], debugging [3][5], testing [1][18], and evolution analysis [11][17], while the problems of how spreadsheet errors are introduced, in particular when reusing spreadsheet templates, are unexplored. To the best of our knowledge, there is no study on spreadsheet template usage.

In this paper, we present the first empirical study on the usage of spreadsheet templates in practice to obtain insights for spreadsheet usage and research. Our study is conducted on the Enron corpus [13], an industrial-scale spreadsheet corpus, which was collected from the email archive of the Enron Corporation [22]. We collect 47 predesigned templates (490 instances in total) from the Enron corpus by a keyword-based approach. We further collect 21 instance template groups (168 template and instance pairs in total) from the Enron corpus by a random sampling. Then, we analyze the changes made during the creation of instances, find out what kinds of errors occur in templates and their corresponding instances, and understand how they were introduced. This seminal study provides various evidences and insights, and uncovers several opportunities for users and researchers to achieve more effective spreadsheet practice from the perspective of spreadsheet templates.

2 RESEARCH QUESTIONS

The quality of predesigned templates has a wide impact on its instances. If an error occurs in a predesigned template, it may be propagated to all its instances without users' noticing. To address the quality concerns, we raise the first research question: ***RQ1: Are the predesigned templates well-designed?***

Although many techniques have been proposed to detect errors in spreadsheets, why and how spreadsheet errors are introduced are still unexplored. Such knowledge can be leveraged to invent more effective techniques of error detection, prevention and fixing. Therefore, we raise the second research question: ***RQ2: What and how errors are introduced during predesigned template usage?***

If the predesigned templates are not available, users tend to reuse the instances instead of templates. To find out the prevalence of such practice and reveal its problems, we raise the third research question: ***RQ3: What is difference between the predesigned template reuse and instance template reuse?***

3 METHODOLOGY

To study the usage of spreadsheet templates in practice, we focus on the templates contained in the Enron corpus [13], whose spreadsheets are extracted from industry and can reflect how users use the templates in practice.

3.1 Template and Instance Identification

Spreadsheet templates have two levels of granularity:

- ***Worksheet-level template***: A worksheet in a spreadsheet is designed or used as a template, and its instances are worksheets.
- ***Spreadsheet-level template***: A spreadsheet is designed or used as the template. A spreadsheet-level template may contain more than one worksheet.

The relationship among templates and their instances is rarely explicitly documented. Thus, a spreadsheet-level template and its instances are usually stored as independent spreadsheets by spreadsheet systems, e.g. Microsoft Excel. Although VEnron [11] clusters multiple versions of the same spreadsheet into an evolution group, it is still challenging to distinguish the instances of a spreadsheet-level template from the revised versions of another instance. Different from the spreadsheet-level templates, a worksheet-level template and its instances are usually placed in the same spreadsheet, and do not suffer from this issue. To ensure the accuracy of template and instance identification, we only focus on the worksheet-level templates in this paper.

Identifying predesigned templates and their instances. We use keywords (i.e., "template" and "blank") to search the possible predesigned worksheet-level templates in the Enron corpus. The worksheets in the search results are identified as predesigned templates if their most data cells contain default values, e.g., 0.

We identify corresponding instances by manually checking whether existing worksheets have the similar layout with the template in the spreadsheets. If a worksheet shares the similar layout with a predesigned template, it is identified as an instance.

Identifying instance templates and their instances. To inspect instance templates in practice, we randomly sample 100 spreadsheets that contain at least three non-empty worksheets from the Enron corpus. Then, we carefully inspect these 100 spreadsheets and find that 48 out of them contain at least two similar worksheets. Among them, only 3 spreadsheets contain predesigned worksheet templates. This means that it is common for users to reuse instance templates instead of predesigned templates (45% vs. 3%).

Then, we cluster similar worksheets in the same spreadsheets into groups and try to recover the creation order of these similar worksheets in each group according to the chronological order (e.g., "Jan 2001" and "Feb 2001") or sequence information (e.g., "Sheet1" and "Sheet1(2)") in them. Finally, we successfully recover the creation order in 30 worksheet groups. For each worksheet group, we assume that for each two adjacent worksheets, the first one can serve as the instance template to create the second one. This assumption is reasonable in practice: users usually reuse the latest worksheet to create a new one, other than reusing earlier ones.

Note that some of the identified templates and instances are duplicated. To avoid their impact on our result, we design a feature-based duplication removal algorithm which compares some features (e.g., cell type and font family) of two worksheets cell by cell. Finally, we in total collect 47 predesigned templates and their 490 instances, and 21 instance template groups, including 168 template and instance pairs.

3.2 Analysis of Template Usage

To answer the above three research questions, we manually align an instance and its corresponding template by applying change operations (e.g., inserting or deleting rows or columns), then automatically identify the content changes cell by cell, including the changes on formulas, headers, font family and size, and styles of table borders. We implement a plugin for Microsoft Excel to record all changes for further analyses. For example, our manual inspection goes into the changes on formulas and table layouts to detect errors during template usage.

3.3 Threats to Validity

The representativeness of our studied spreadsheets is the main threat to external validity. Different from other existing spreadsheet corpora EUSES [12] and Fuse [4], the Enron corpus was extracted from industry rather than the Internet, whose spreadsheets can reflect how users use the templates in practice. The main threats to internal validity come from the accuracy of template and instance identification. To identify the instances for each predesigned template, we simply assume that the worksheets with the same layout in the same spreadsheets are created by reusing the predesigned templates. To identify the instances for each instance template, we assume that users use the latest instance as the template to create new instances. These assumptions are not always true. But it is quite common in practice according to our experience.

4 EMPIRICAL STUDY RESULTS

4.1 Quality of Predesigned Templates

Intuitively, a well-designed predesigned template should meet the following four requirements.

- **Predefine a complete table layout.** The table layout means the spatial relationship between the header cells and data cells, including the headers and some settings related to the appearance of the tables, such as the border styles and background colors of cells.
- **Predefine all necessary formulas.** Formulas are the business logics implemented in the spreadsheets, and determine what the inputs are and how to process data.
- **Eliminate common operations made in creating different instances.** Common operations are operations that are performed in the creation of most instances, which hinder users' effort minimization. Thus, they can reflect the completeness of the predesigned templates.
- **Contain no errors.** Like bugs can be propagated through code reusing, e.g., code cloning [19], errors in a predesigned template can be propagated to its instances without noticing and may cause serious damages to the quality of instances.

Table 1. Issues in Predesigned Templates

Issues	Predesigned Template
Common changes	9 (19%)
Missing formulas	14 (30%)
Formula errors	3(6%)
Total	19(40%)

Based on the above four requirements for well-designed templates, we evaluate the design of predesigned templates by detecting common changes, missing formulas and formula errors. Specifically, common changes reflect the completeness of the predesigned templates. For example, if all or most instances of a template include the same header, the template has an incomplete layout and can be refined by introducing the header. In this paper, the changes and formulas that occur in more than half of a template's instances are considered as common changes. We detect missing formulas in a predeisigned template using the following rule: the cells in instances contain formulas, while the corresponding cells in the predesigned template do not. It is challenging to discover and confirm all formula errors in the predesigned templates. Thus, we only focus on the range errors [21], of which the identification is feasible and easily confirmed.

As shown in Table 1, we find that common changes exist in the usage of 9 (19%) predesigned templates, 14 (30%) predesigned templates have missing formulas, and 3 (6%) predesigned templates contain range errors.

Answer to RQ1: *40% of predesigned templates have design issues, which may counteract the expected benefits, and substantial improvements are required.*

4.2 Errors in Predesigned Template Usage

We observe four kinds of errors introduced in the instances when using predesigned templates, including missing formulas, range errors, unnecessary formulas and inconsistent formulas. Table 2 shows the numbers of detected errors caused by different reasons. We in total detected 19,965 errors, introduced in the usage of 79% (37 out of 47) predesigned templates.

We detected 18,741 missing formulas in 317 instances of 34 (72%) templates. The main reason is that users overwrote a formula with a constant value. While, 560 missing formulas are caused by the missing formulas in predesigned templates. That indicates the errors in the predesigned template can be easily propagated to its instances without users' noticing.

Range errors [21]occur when the ranges of cells referenced by formulas include unrelated cells or miss some related cells. We detected 605 range errors in 63 instances of 16 (34%) templates. The main cause is that users forgot to append the new inserted cells to the range. Take Microsoft Excel as an example, if a new cell is inserted somewhere in the middle of a range, the range will be automatically expanded, such that the new value and all old values are still referenced by formulas. However, users may append the range by inserting cells at the boundary, and the

Table 2. Distribution of the Four Types of Errors Caused by Different Reasons

Missing Formula		Range Error		Unnecessary Formula		Inconsistent Formula	
Reason description	*Detected*	*Reason description*	*Detected*	*Reason description*	*Detected*	*Reason description*	*Detected*
Overwrite	17,552	Range change	475	Cell type change	83	Wrong repair	217
Error in template	560	Cell type change	86	Wrong location	53	Wrong formula	198
Cell insertion	291	Error in template	38	Formula deletion	30	Reference	36
Formula deletion	290	Row insertion	6	Location change	2		
Row (Cell) insertion	45(3)						

values in the new inserted cells will not be included in the range and referenced by formulas automatically.

Unnecessary formulas denote the formulas added in the instances with no valid inputs. We detected 168 unnecessary formulas in 64 instances of 12 (26%) templates. The main cause is that users changed the types of all the cells referenced by the formulas. Such operations may make the formula receive no valid input, (e.g., date, string or sequence numbers) and become unnecessary.

Inconsistent formulas denote the formulas used in the instances are different from what they are in the corresponding templates. We in total detected 451 inconsistent formula errors in 50 instances of 11 (23%) templates. Most of them are caused by wrong error detection and repair suggestions provided by spreadsheet systems, e.g., Excel. When creating new instances, users accepted those repair suggestions and modified formulas by provided operations in the smart tag.

Answer to RQ2: *The usage of predesigned templates faces severe issues. In the usage of 79% predesigned templates, at least one error is introduced, and missing formulas are the most common errors.*

4.3 Errors in Instance Template Usage

Similar to predesigned template study, we first inspect the quality of instance templates by checking the errors contained in them. In total, we detected 391 errors in 30% (51 out of 168) instance templates, including 357 missing formula errors and 34 unnecessary formula errors.

We also study the issues in using instance templates like the predesigned template study. We in total detect 644 formula errors in the instances of 32% (54 out of 168) instance templates, including 365 missing formula errors, 277 inconsistent formula errors and 2 range errors. Missing formulas are also the most common errors in instance reuses and 229 missing formulas are caused by missing formulas that host in instance templates.

Unlike the usage of predesigned templates, when creating new spreadsheets based on instance templates, users need to clean the data in the instance templates since they are not useful for the new instances. While, there is a higher risk of introducing errors during data cleaning. We investigate how many data in the original instance templates should be cleaned. We find that in 52% (87 out of 168) of cases, the proportion of data cells that were cleaned or overwritten is over 50%. We find 101 missing formulas were caused by data cleaning, which is a new but very popular cause in instance template usage. Data cleaning becomes a new challenge in the usage of instance templates.

Answer to RQ3: *The usage of instance templates suffers from the same types of errors.*

5 RESEARCH OPPORTUNITIES

Our preliminary study highlights several research opportunities:

Template extraction. Our study shows that 40% of predesigned templates have design issues. This finding implies that it is difficult to build a better abstraction which can cover common requirements from potential users. An approach that can extract or refine templates from a set of instances would be very helpful for reducing efforts and preventing errors.

Template-based error detection. Errors in spreadsheet templates and their instances are common. For example, we find that missing formulas in instances usually exist in their corresponding templates. A template-based approach, which detects and fixes missing formulas by comparing an instance with its template, is promising.

Propagated errors tracing. Errors in spreadsheet templates may be propagated to their instances without users' noticing. Table 2 shows that there are 598 propagated errors in templates. The relationships between templates and instances are usually missing due to lack of management systems for spreadsheets. Once an error is detected in a template, how to trace the propagated errors in its instances is important and challenging.

6 CONCLUSIONS

Spreadsheet templates are proposed to help users create spreadsheets efficiently. Improper design and usage of templates can introduce various issues. Based on 47 predesigned templates (490 instances) and 21 instance template groups (168 template and instance pairs) collected from the Enron corpus, we perform the first empirical study on the design and usage of these templates used in practice. Our findings highlight the issues during spreadsheet template usage and several interesting research directions. We have made our study results available online (*http://www.tcse.cn/~wsdou/project/TmplEnron*) for further studies.

It is generally inappropriate to generalize our findings from a single case, but our findings provide important insights for further studies. Our study framework proposed in this paper can also be reused on more datasets to obtain more generalizable results. We plan to perform further studies on other datasets in the future.

ACKNOWLEDGMENTS

This work was partially supported by National Key Research and Development Program of China (2017YFA0700603), National Natural Science Foundation of China (61702490), Frontier Science Project of Chinese Academy of Sciences (QYZDJ-SSW-JSC036), and Youth Innovation Promotion Association at CAS.

REFERENCES

[1] R. Abraham, and M. Erwig. 2006. AutoTest: A Tool for Automatic Test Case Generation in Spreadsheets. In *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 43–50.

[2] Robin Abraham, and Martin Erwig. 2007. UCheck: A Spreadsheet Type Checker for End Users. *Journal of Visual Languages & Computing* 18: 71–95.

[3] Robin Abraham, and Martin Erwig. 2007. GoalDebug: A Spreadsheet Debugger for End Users. In *Proceedings of International Conference on Software Engineering (ICSE)*, 251–260.

[4] Titus Barik, Kevin Lubick, Justin Smith, John Slankas, and Emerson Murphy-Hill. 2015. FUSE: A Reproducible, Extendable, Internet-Scale Corpus of Spreadsheets. In *Proceedings of the 12th Working Conference on Mining Software Repositories (MSR)*, 486–489.

[5] Daniel W. Barowy, Dimitar Gochev, and Emery D. Berger. 2014. CheckCell: Data Debugging for Spreadsheets. In *Proceedings of ACM International Conference on Object Oriented Programming Systems Languages & Applications (OOPSLA)*, 507–523.

[6] Shing-Chi Cheung, Wanjun Chen, Yepang Liu, and Chang Xu. 2016. CUSTODES: Automatic Spreadsheet Cell Clustering and Smell Detection using Strong and Weak Features. In *Proceedings of International Conference on Software Engineering (ICSE)*, 464–475.

[7] Wensheng Dou, Shing-Chi Cheung, Chushu Gao, Chang Xu, Liang Xu, and Jun Wei. 2016. Detecting Table Clones and Smells in Spreadsheets. In *Proceedings of ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE)*, 787–798.

[8] Wensheng Dou, Shing-Chi Cheung, and Jun Wei. 2014. Is Spreadsheet Ambiguity Harmful? Detecting and Repairing Spreadsheet Smells due to Ambiguous Computation. In *Proceedings of International Conference on Software Engineering (ICSE)*, 848–858.

[9] Wensheng Dou, Shi Han, Liang Xu, Dongmei Zhang, and Jun Wei. 2018. Expandable Group Identification in Spreadsheets. In *Proceedings of the ACM/IEEE International Conference on Automated Software Engineering (ASE)*, 498–508.

[10] Wensheng Dou, Chang Xu, S.C. Cheung, and Jun Wei. 2017. CACheck: Detecting and Repairing Cell Arrays in Spreadsheets. *Transactions on Software Engineering (TSE)* 43: 226–251.

[11] Wensheng Dou, Liang Xu, Shing-Chi Cheung, Chushu Gao, Jun Wei, and Tao Huang. 2016. VEnron: A Versioned Spreadsheet Corpus and Related Evolution Analysis. In *Proceedings of International Conference on Software Engineering Companion (ICSE)*, 162–171.

[12] Marc Fisher, and Gregg Rothermel. 2005. The EUSES Spreadsheet Corpus: A Shared Resource for Supporting Experimentation with Spreadsheet Dependability Mechanisms. *ACM SIGSOFT Software Engineering Notes*: 1–5.

[13] Felienne Hermans, and Emerson Murphy-Hill. 2015. Enron's Spreadsheets and Related Emails: A Dataset and Analysis. In *Proceedings of the 37th IEEE International Conference on Software Engineering (ICSE)*, 7–16.

[14] Felienne Hermans, Martin Pinzger, and Arie van Deursen. 2012. Detecting and Visualizing Inter-Worksheet Smells in Spreadsheets. In *Proceedings of International Conference on Software Engineering (ICSE)*, 441–451.

[15] Felienne Hermans, Martin Pinzger, and Arie van Deursen. 2015. Detecting and Refactoring Code Smells in Spreadsheet Formulas. *Empirical Software Engineering* 20: 549–575.

[16] Felienne Hermans, Ben Sedee, Martin Pinzger, and Arie van Deursen. 2013. Data Clone Detection and Visualization in Spreadsheets. In *Proceedings of International Conference on Software Engineering (ICSE)*, 292–301.

[17] Bas Jansen, and Felienne Hermans. 2015. Code Smells in Spreadsheet Formulas Revisited on an Industrial Dataset. In *Proceedings of IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 372–380.

[18] G. Rothermel, L. Li, C. DuPuis, and M. Burnett. 1998. What You See Is What You Test: A Methodology for Testing Form-Based Visual Programs. In *Proceedings of International Conference on Software Engineering (ICSE)*, 198–207.

[19] Chanchal K Roy, and Kevin A Schneider. 2017. A Study on Bug Propagation through Code Cloning. In *Proceedings of the 33rd IEEE Internatioal Conference on Software Maintenance and Evolution (ICSME)*, 229–237.

[20] Christopher Scaffidi, Mary Shaw, and Brad Myers. 2005. Estimating the Numbers of End Users and End User Programmers. In *2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)*, 207–214.

[21] Thomas Schmitz, and Dietmar Jannach. 2016. Finding Errors in the Enron Spreadsheet Corpus. In *Proceeding of IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 157–161.

[22] Enron Corporation. Retrieved from https://en.wikipedia.org/wiki/Enron