Evaluating Bug-Fixing in Software Product Lines: an Industrial Case Study

Jorge Echeverría
Universidad San Jorge
Zaragoza, Spain
jecheverria@usj.es

Francisca Pérez
Universidad San Jorge
Zaragoza, Spain
mfperez@usj.es

Andrés Abellanas
BSH Electrodomésticos
Zaragoza, Spain
andres.abellanas@bshg.com

Jose Ignacio Panach
Universitat de València
Valencia, Spain
joigpana@uv.es

Carlos Cetina
Universidad San Jorge
Zaragoza, Spain
ccetina@usj.es

Óscar Pastor
Universitat Politècnica de València, Spain
opastor@pros.upv.es

ABSTRACT

[Background] Bug-fixing could be complex in industrial practice since thousands of products share features in their configuration. Despite the importance and complexity of bug-fixing, there is still a lack of empirical data about the difficulties found in industrial Software Product Lines (SPLs). [Aims] This paper aims to evaluate engineers' performance fixing errors and propagating the fixes to other configured products in the context of an industrial SPL. [Method] We designed and conducted an empirical study to collect data with regard to bug-fixing tasks within the context of a Induction Hob SPL in the BSH group, the largest manufacturer of home appliances in Europe. [Results] We found that effectiveness, efficiency and satisfaction got reached good values. Through interviews we also found difficulties related to unused features, cloning features unintentionally, detecting modified features, and propagating the fix when the source of the bug is the interaction between features. [Conclusions] The identified difficulties are relevant to know how to better apply SPLs in industry in the future.

Keywords

Software Product Line, Variability Modeling, Usability

1. INTRODUCTION

Software Product Lines (SPLs) have been applied in a wide variety of domains as a means to achieve quality improvements, extensive reuse, and development productivity [3]. In SPLs, product line engineers can create a family of products by configuring each product through the selection and customization of different features. In industrial SPLs, bug-fixing could be complex for product line engineers since thousands of products share features among them.

Most of the existing works in the field of empirical analysis in SPLs focus on analyzing testing techniques to look for bugs, but they do not deal with how to fix those bugs. For example, Steffens et al. conducted an industrial evaluation using a pairwise SPL testing technique named MoSo-PoLiTe [11]. Ma et al. propose a random testing technique for SPLs [8]. Uzuncaova et al. evaluate a testing technique that incrementally generates tests for SPLs [13]. There are other approaches that have studied the reliability on improvements in the evolution of SPL through empirical studies, such as Krishnan et al. [7]. Krishnan analyzes to what extent the common and variation components change over time. The analysis focuses on Eclipse Product Line from 2007 to 2010, considering failure trends, change trends and failure relationship as variables. Other approaches such as Ohira et al. have evaluated how the use of patterns can improve the bug management [9].

The aim of this paper is to evaluate bug-fixing of an Induction Hob SPL in the BSH group. BSH is the largest manufacturer of home appliances in Europe and one of the leading companies in the sector worldwide. Their induction division has been producing induction hobs (the brand portfolio is composed by Bosch and Siemens among others) over the last 15 years.

To evaluate bug-fixing, we planned a case study carried out by the engineers of the induction division. Engineers had to perform tasks in order to fix a bug in a configured product and to propagate the fix to other configured products. The results show difficulties for bug-fixing in SPLs with regard to unused features, cloning features unintentionally, detecting modified features, and propagating the fix when the source of the bug is the interaction among features. These difficulties are relevant for SPL approaches and addressing them can contribute to promote the adoption of SPLs in industry.

The remainder of the paper is structured as follows: Section 2 overviews SPLs and bug-fixing in SPLs. Section 3 presents the case study design. Section 4 and 5 present the results and discussion. Section 6 describes the threats to validity. Section 7 concludes the paper.

ESEM '16, September 08-09, 2016, Ciudad Real, Spain
© 2016 ACM. ISBN 978-1-4503-4427-2/16/09…$15.00
DOI: http://dx.doi.org/10.1145/2961111.2962635

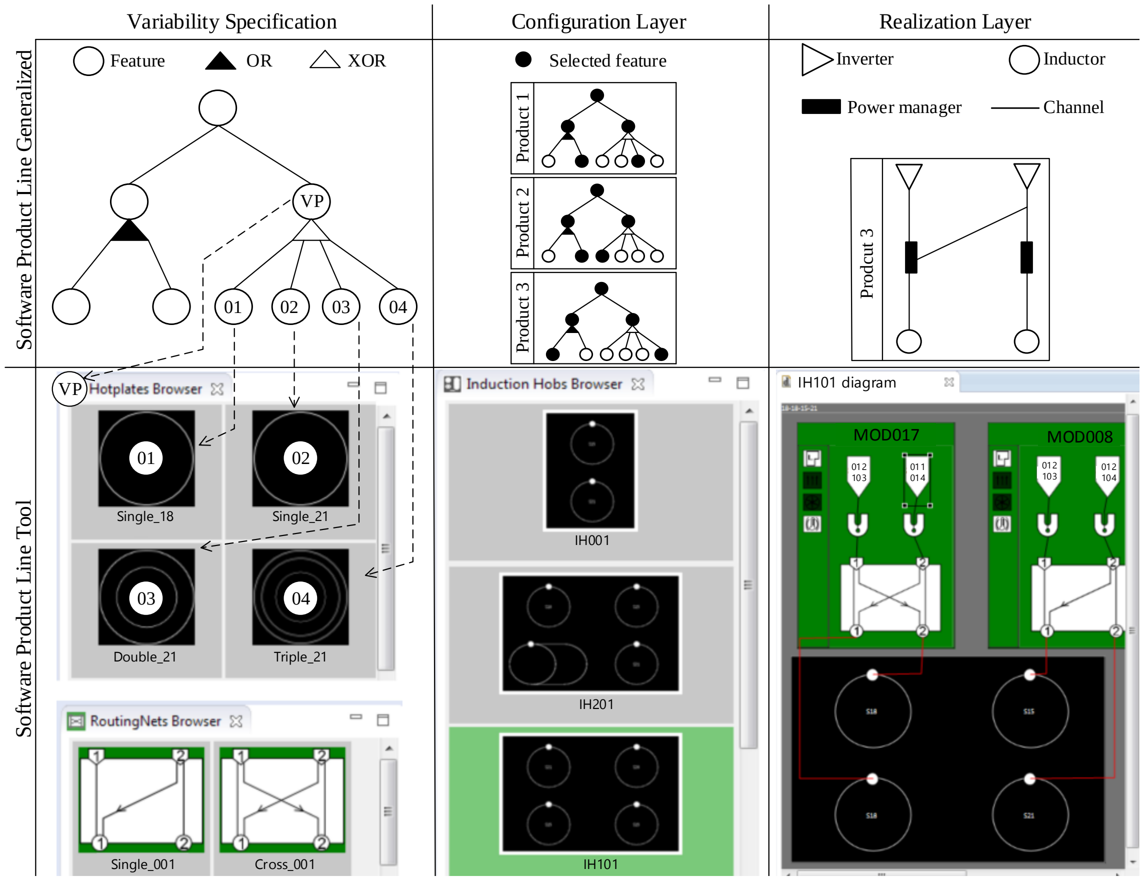


Figure 1: Software Product Line Tool

2. SPLS AND BUG-FIXING

SPLs require tools that allow users to tailor products that fit business requirements or market needs by varying the features that compose each product. Up to now, many variability tools such as pure::variants (www.pure-systems.com) and Gears (www.biglever.com) have been created to support SPLs. Although implementation details are different, these tools include the following key elements:

Variability Specification. This represents the information of all possible opportunities of variation of an SPL (variation points). A widely used representation for variation points are feature models [6], which represent all possible opportunities of variation in terms of features and relationships among them as a hierarchically arranged set. For example, a variability specification using a feature model is shown in the upper-left side of Figure 1. For each Variation Point (see VP in the figure), the product configuration tool shows a set of options that can be selected for product configuration.

The bottom-left side of Figure 1 shows the tool that BSH uses to configure Induction Hobs (IHs) in which a catalog of available *hotplates* is shown since *hotplates* are captured as variation point in the feature model. The catalog contains many elements as features are specified for that variation point in the feature model. As the figure shows, the catalog of hotplates is made up of four different types of hotplates due to the specification of four features (see the features from 01 to 04 in the figure) for the *hotplate* variation point.

Configuration layer. This is a portfolio with different configured products. The differences that exist across different configured products are the features that are selected in the variability specification. For example, the upper-middle side of Figure 1 shows that the portfolio is made up of 3 different products, which have different selected features in the variability specification (see products from 1 to 3 in the figure).

In the BSH SPL, a graphical representation depicts each IH that make up the portfolio of IHs as the snapshot of the BSH tool shows in the bottom-middle side of Figure 1.

Realization layer. This enables the customization of the features that have been selected for a specific product in the variability specification. Features are merely symbols that are materialized in this layer to fragments of code of a programming language (e.g., Java), or to model fragments that are expressed using general purpose modeling languages (e.g., UML) or Domain Specific Modeling Languages (e.g., the Induction Hob Domain Specific Language). For example, the upper-right side of Figure 1 shows the model of the *Product 3* that is obtained by materializing each selected feature to its model fragment. Specifically, this model is expressed using the Induction Hob Domain Specific Language, which its main concepts are the following: *Inverter, Inductor, Power Manager,* and *Channel.*

The bottom-right side of Figure 1 shows a snapshot of the BSH tool that shows a model with the product realization of an IH product identified as *IH101.*

ID	Name	Description
CL	Bug-fixing in the Configuration Layer	A bug has been found in the induction hob IH011 with the module MOD006. This module must be replaced with the module MOD014. The module MOD006 must not be replaced in other induction hobs.
CL-P	Bug-fixing in the Configuration Layer with Propagation to other products	A bug has been found in the induction hob IH017 since the inverter INV014958 of the module MOD016 does not run correctly. The inverter must be replaced by the inverter INV015231. This fix must be propagated to every induction hob that contains the replaced inverter in its configuration.
RL	Bug-fixing in the Realization Layer	A bug has been found in the value of the parameter VMAX that is set in the module MOD020 of the induction hob IH002. The value of the parameter VMAX has to be changed to 39. This fix must not be propagated to other induction hobs.
RL-P	Bug-fixing in the Realization Layer with Propagation to other products	A bug has been found in the value of the parameter VMAX that is set in the module MOD019 of the induction hob IH051. The value of the parameter VMAX has to be changed to 44. This fix must affect every induction hob that contains in its configuration the aforementioned module.

Table 1: Bug-fixing tasks

Over time, it could be necessary bug-fixing in industrial practice (e.g., a bug has been reported in an IH product due to performance issues in a *hotplate*). In the context of a SPL, bug-fixing could encompass the following tasks:

- **Bug-fixing in the Configuration Layer.** This entails the replacement of a selected feature with a different one. For example, replacing the selected *single18 hotplate* feature with the *triple21 hotplate* feature.
- **Bug-fixing in the Realization Layer.** This entails the modification of code or model elements of a configured product. For example, changing the *VMAX* parameter of the *inverter* to 42.
- **Bug-fixing propagation.** This could be done to spread the fix to other configured products. For example, the *IH101* product has been fixed but it may be also necessary to propagate the fix to other IH products that share features with the *IH101* product.

Therefore, four tasks (see Table 1) have been chosen for importance and frequency of use. According to the engineers, these tasks are necessary for many industrial processes and are done daily.

3. DESIGN OF THE CASE STUDY

3.1 Objective

The goal of this study is to evaluate engineers' performance fixing errors and propagating the fix to other configured products in the context of SPL of the BSH induction division. Since companies use to manage thousands of features and products (e.g., the BSH product configuration tool manages about 2^{100} product configurations), bug-fixing can be difficult to handle by product line engineers.

Following Wohlin et al.'s guidelines [15], the goal of our study was to:

Analyze the engineers' performance in a SPL when they fix errors;

For the purpose of filling in the gap of in empirical evaluation of this topic;

With respect to the layer where the error is detected and the different fix propagation;

From the viewpoint of software engineers;

In the context of a SPL in the BSH induction hob division.

In relation to the above goal, we seek to answer the following research questions:

RQ1 What is the effectiveness to fix errors and to propagate the fix in a family of products in a SPL?

RQ2 What is the efficiency to fix errors and to propagate the fix in a family of products in a SPL?

RQ3 What is the satisfaction to fix errors and to propagate the fix to the family of products in a SPL?

To answer the above research questions, we use a case study. The response variables in our research design are effectiveness, efficiency and satisfaction. The factor is the kind of bug-fixing with four levels. These four levels are: bug-fixing in the Configuration Layer (CL), bug-fixing in the Configuration Layer with Propagation to other products (CL-P), bug-fixing in the Realization Layer (RL) and bug-fixing in the Realization Layer with Propagation to other products (RL-P).

3.2 Metrics

All response variables are measured based on the performance of the engineers with study tasks. The **effectiveness** is defined as the percentage of study tasks performed correctly by the engineer without assistance. The task is decomposed into a set of subtasks following the Keystroke-Level Model method to calculate the task percentage. For instance, if half of these subtasks within a task are performed without assistance, the effectiveness is 50%. The **efficiency** is the ratio between the effectiveness and the spent time (in minutes) to perform the task according to Common Industry Format (CIF) for Usability Test Reports [1]. Finally, the **satisfaction** is measured using a satisfaction questionnaire filled out by the engineers after finishing the study tasks. The questionnaire was composed by ten questions with a Likert scale.

3.3 Instruments

The following methods and sources are used to collect data:

Demographic Questionnaire. This includes questions to identify the profile of each subject. The information asked through the questionnaire is: their educational level, length of time working in the actual department (years), age, gender, time spent working with the software of IHs every day, and knowledge about development environments and modeling tools.

Performance Measurement. This evaluates how subjects perform bug-fixing of IHs in a SPL through performance times, completed tasks and time used when the subjects were assisted to perform the tasks. This data enables the calculation of efficiency and effectiveness.

Satisfaction questionnaire. This is System Usability Scale (SUS), which measures participant's subjective satisfaction with the tool. The questionnaire was composed by ten questions with a Likert scale. In the SUS original the word "system" was replaced by "SPL tool". SUS, with only ten questions, yields reliable results [12]. The SUS questions address different aspects of the user's reaction to the bug-fixing facet of the SPL tool as a whole (e.g., "I found the SPL tool unnecessarily complex", "I felt very confident using the SPL tool") as opposed to asking the user to assess specific features of the system (e.g., visual appearance, organization of information, etc.).

Interview. This determines the participants' understanding of the bug-fixing facet of the SPL, and to detect the parts of the bug-fixing facet of the SPL that are more problematic from a human computer interaction point of view along with the real causes of the problems [5]. For instance, a question is "Which step of the task has been the most difficult for you?".

3.4 Participants

The subjects were five electronic engineers that worked in the BSH induction division. These engineers are experts in developing induction hub software. They spent from 1.5 to 12 years working in the induction department (a mean of 6.7 years). They spent from 1 to 8 hours per day working with software of IHs (a mean of 5 hours per day). In addition, 60% of the subjects stated that they were proficient in integrated development environments (e.g. Eclipse).

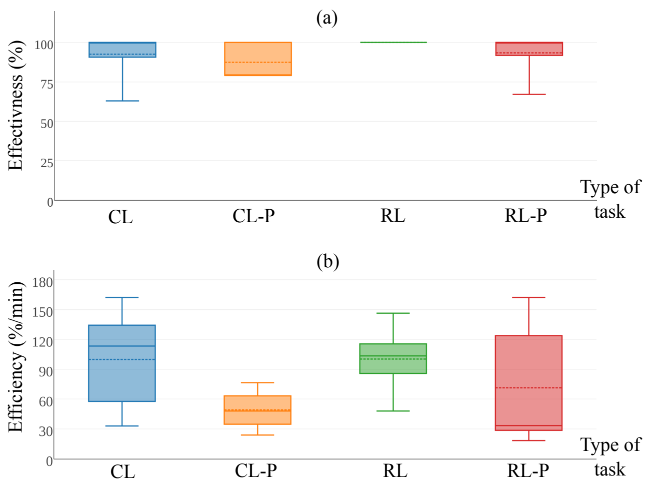
Apart from the subjects, an instructor and an observer were also involved. The instructor provided the information about the tool, gave a tutorial of the tool, clarified doubts, and interviewed the subjects. The observer took notes and recorded the interviews for further analysis.

3.5 Procedure

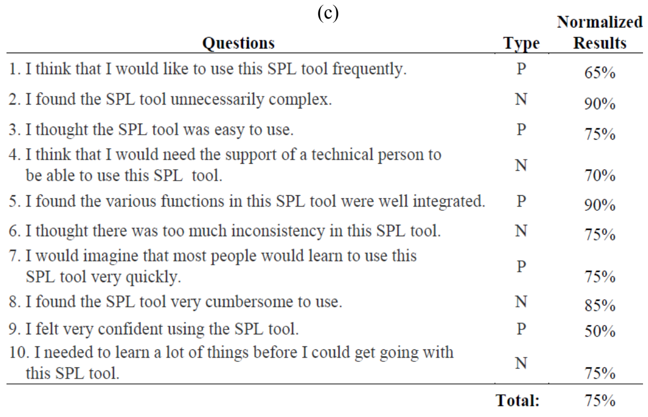
This section describes the procedure used to run the study.

1. Subjects started by receiving information about the goals and the procedure of the study. Also, they were informed that their interaction would be recorded.
2. The participants attended a small tutorial about the BSH tool. This tutorial was taught by the instructor.
3. Subjects were asked to fill in the demographic questionnaire.
4. Subjects received instructions for the Performance Measurement. They were advised to try to accomplish the tasks without any assistance, and that they should only ask for help if they felt unable to complete the task on their own.
5. Subjects were asked to complete four tasks related to the bug-fixing facet of the SPL tool that are obtained as a result of combining the bug-fixing tasks identified at the end of Section 2. Table 1 shows the list of tasks, their identification, name and description. These tasks were used to calculate the effectiveness and efficiency. To avoid a possible ceiling effect, time was not limited to complete the tasks.
6. The observer writes down subjects' performance when carrying out bug-fixing tasks. This information is complemented with a video recording of the session.
7. Subjects were then asked to complete a System Usability Scale questionnaire. This questionnaire was used to calculate the satisfaction of engineers about the tool under study.
8. Subjects were interviewed by the instructor about the tasks that they performed.
9. The observer reviewed the recordings of the engineers performing the tasks in order to calculate the efficiency, effectiveness and satisfaction. Finally, the interview about the tasks was transcripted.

4. RESULTS



(c)



Questions	Type	Normalized Results
1. I think that I would like to use this SPL tool frequently.	P	65%
2. I found the SPL tool unnecessarily complex.	N	90%
3. I thought the SPL tool was easy to use.	P	75%
4. I think that I would need the support of a technical person to be able to use this SPL tool.	N	70%
5. I found the various functions in this SPL tool were well integrated.	P	90%
6. I thought there was too much inconsistency in this SPL tool.	N	75%
7. I would imagine that most people would learn to use this SPL tool very quickly.	P	75%
8. I found the SPL tool very cumbersome to use.	N	85%
9. I felt very confident using the SPL tool.	P	50%
10. I needed to learn a lot of things before I could get going with this SPL tool.	N	75%
	Total:	75%

SUS results. *P = Positively worded item, N= Negatively worded item.*

Figure 2: Results

Next we discuss the results for response variables.

Effectiveness. Figure 2 (a) shows that 15 out of 20 tasks (four tasks per user) were completed and performed correctly. In terms of frequency of assistance, one user (User 3) required assistance in the task CL, and other user (User 2) required assistance in the task RL-P. The remainder of tasks did not require assistance. In addition, the task CL-P was correctly finished by two users, the remainder of the users did not spread correctly the required changes. On the other hand, the average effectiveness of the task CL-P was 87.4%, meanwhile the remainder of the tasks scored a value of average effectiveness higher than 90%. Finally, the task RL was correctly performed by all users. According to these results, we can answer RQ1 stating that effectiveness to fix errors and to propagate the fix in a family of products in a SPL obtains a good value (close to 100%).

Efficiency. Figure 2 (b) shows that the tasks which involved only one product (CL and RL) were performed with higher efficiency than the tasks which involved propagation (CL-P and RL-P). Furthermore, when the propagation is not involved, the average for efficiency is very similar in both tasks configuration (99.83%/min) and realization (100.28%/min) layers. When propagation is required to fix the bug, the average of efficiency is better in the realization layer task (71.39%/min) than in the configuration layer task (49.1%/min). On the other hand, the standard deviation is higher than 35%/min in all tasks except in the task CL-P (19.96%/min). Finally, considering the tasks with effectiveness equal to 100%, the average of efficiency in task CL-P was 48.52%/min, while the average of efficiency in the other tasks was higher than 84.62%/min. According to these results, we can answer RQ2 stating that efficiency to fix errors and to propagate the fix in a family of products in a SPL is close to 100%/min when effectiveness is low.

Satisfaction. Figure 2 (c) shows the perceived satisfaction (75%). This score, according to Bangor et al. to improve the interpretation of SUS scores, qualifies the tool as "good" [2]. In addition, items 3 and 5 related to complexity and functionality respectively reached the highest value (90%). On the other hand, item 9 related to confidence scored the lowest value (50%). According to these results, we can answer RQ3 stating that satisfaction to fix errors and to propagate the fix in a family of products in a SPL obtains a good value.

5. DISCUSSION

This section analyzes in detail the results through the interviews the instructor did at the end of the tasks. Subjects were asked about their required assistance and difficulties during the performance of the bug-fixing tasks, and about their answers in the satisfaction questionnaire. This information enabled us to increase the knowledge about users' performance in a SPL when they fix errors, enhancing the answers to the research questions as follows:

RQ1 What is the effectiveness to fix errors and to propagate the fix in a family of products in a SPL?

In the configuration layer, subjects had to fix a bug in an IH product by replacing one of its features with a different one. Whereas four subjects successfully fixed the bug, one subject required the Instructor's assistance because he did not remember the necessary steps in the SPL tool to replace one feature in a configured product.

In the realization layer, subjects had to fix a bug in an IH product by changing the value of a parameter that is set in a selected feature of its configuration. All subjects successfully fixed the bug without assistance. This may be due to the fact that subjects perform their main daily work in the realization layer.

In the CL-P propagation task, three subjects did not successfully complete the task because they propagated the bug-fix from a product to other products that did not require the bug-fix. By reviewing the videos, we detected that the difficulty was related to bugs that depended on the interaction among features. Specifically, the bug occurred in CL-P due to the simultaneous configuration of a feature (e.g., feature A) with another one (e.g., feature B). The strategy applied by all subjects to fix the bug was to modify one of the two features and then propagate the fix. Two of the five subjects only propagated the fix to the products that had the simultaneous configuration (feature A and feature B).

In the RL-P propagation task, one subject did not successfully complete the task. The subject fixed the bug in one induction hob, he abandoned the task performance because he failed to propagate the fix in a product family. In the interview, the subject claimed that it was very important a better feedback when a new feature is created.

RQ2 What is the efficiency to fix errors and to propagate the fix in a family of products in a SPL?

In every task, the subjects spent time trying to test whether the changes to fix the bug were correctly performed. For instance, the User 4 claimed:*"I am not sure whether the propagation has been correctly performed and I need to check the changes"*. On the other hand, User 5 achieved the better values of efficiency. This was due to the detailed knowledge about the features in the SPL tool of User 5.

In the CL task, User 3 emphasized the importance of how the different features are named. He claimed in the interview:*"I spent a lot of time because the names of the inverters are different between the SPL tool and my daily work"*.

In the RL task, all subjects successfully fixed the bug. We detected that four of the five subjects created clones of features unintentionally. This was because subjects created model fragments in the realization layer to fix the bug that correspond to features that already exist. The bug-fixing strategy applied by all subjects in RL tasks was to copy and modify the feature and then they selected this fixed feature for the product that produced the bug. In the interview, three of the subjects stated that copying a feature was a cumbersome and error-prone task. In fact, the time obtained with regard to efficiency confirms that manually recreating a feature is the step which takes more time. To avoid this difficulty, engineers suggested that the SPL tool should provide automation to copy and modify features.

RQ3 What is the satisfaction to fix errors and to propagate the fix to the family of products in a SPL?

All users declared that the SPL tool was *"a useful tool"*. Nevertheless, four subjects agreed that SPL tool should report when a feature is not used by any product after bug-fixing. Since the appearance of an unused feature in the SPL is important, engineers need to decide whether the unused feature is removed, kept, or whether its status changes to denote that situation.

Item 9 in the satisfaction questionnaire shows that users are not very confident with the SPL tool. Two of the five subjects claimed that if the name of the modified feature was not changed in the configuration layer after bug-fixing, it could be a source of difficulties to other users of the SPL since they are not going to detect that the feature has been modified at realization layer. Furthermore, User 4 expressed his fear about a correct propagation of the changes and he doubted about the code generated by the SPL tool.

6. THREATS TO VALIDITY

We use the classification of threats to validity in [10]:

Construct validity: This type of validity reflects the extent to which the operational measures represent what the researchers have in mind. This threat was addressed using well established protocols [4].

The instruments used in our research are widely accepted in the human computer interaction research community. Moreover, our study is not focused on specific tool features, our tasks are generic for bug-fixing in a SPL.

Internal validity: This type of validity concerns when the factor being investigated may be affected by other neglected factors. To mitigate this threat, we had no influence on the subjects as they were nominated by our industrial partner. Even though the number of subjects may seem relatively small, the human computer interaction research advises to use five subjects in the usability test to detect 80% of the usability problems [14].

External validity: This type of validity refers to what extent it is possible to generalize the findings. Statistical generalization is not possible from our case study. The bug-fixing tasks need to be evaluated in other modeling tools and in different contexts. Study tasks are typical of the SPL development. For this reason, these results could be relevant for other modelers and other developers of SPL tools. Since the tool used in this study is a concrete tool of our industrial partner, the generalization of findings should be undertaken with caution.

Reliability: This type takes into account to what extent the data and the analysis are dependent on the specific researchers. To minimize this threat, the values of studied measures and subjects' answers have been analyzed using the recordings. For the satisfaction measurement, we used SUS questionnaire, which is a reliable questionnaire according to [12].

7. CONCLUSIONS

Until now, empirical studies in SPLs have been focused on analyzing testing techniques to look for bugs and reliability but there is no empirical data about bug-fixing in SPL. We have conducted a case study for evaluating bug-fixing in the context of an industrial SPL of BSH group (BSH is the largest manufacturer of home appliances in Europe). The number of subjects in this case study is small, so the generalization of findings should be undertaken with caution.

We measured effectiveness, efficiency and satisfaction in order to determine whether there are difficulties in bug-fixing in the context of the SPL of the BSH induction division. Results show that these three variables obtain good values to fix errors and to propagate the fix in a family of products in a SPL. Effectiveness gets values close to 100%, efficiency gets values close to 100% when effectiveness is low and satisfaction is close to the maximum possible value.

The case study reveals difficulties for performing bug-fixing in the configuration layer, in the realization layer and in propagation. These difficulties are relevant for general SPL approaches and addressing them would contribute to promote the adoption of SPLs in industry.

Acknowledgments

This work has been partially supported by the Ministry of Economy and Competitiveness and ERDF funds (Project: TIN2015-64397-R). We thank SMART ADAPT (TIN2013-42981-P) from MINECO and IDEO (PROMETEOII/2014/039) from Generalitat Valenciana.

8. REFERENCES

[1] ANSI/NCITS. Ansi/ncits-354 common industry format (cif) for usability test reports. Technical report, NIST Industry Usability Reporting, 2001.

[2] A. Bangor, P. Kortum, and J. Miller. Determining what individual sus scores mean: Adding an adjective rating scale. *Journal of usability studies*, 4(3):114–123, 2009.

[3] P. Clements and J. McGregor. Better, faster, cheaper: Pick any three. *Business Horizons*, 55(2):201–208, 2012.

[4] N. Condori-Fernández, J. I. Panach, A. I. Baars, T. E. J. Vos, and O. Pastor. An empirical approach for evaluating the usability of model-driven tools. *Sci. Comput. Program.*, 78(11):2245–2258, 2013.

[5] N. Juristo and X. Ferre. How to integrate usability into the software development process. In *Proceedings of the 28th International Conference on Software Engineering*, ICSE '06, pages 1079–1080, 2006.

[6] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson. Feature-oriented domain analysis (FODA) feasibility study. Technical report, Carnegie-Mellon University, November 1990.

[7] S. Krishnan, R. R. Lutz, and K. Goševa-Popstojanova. Empirical evaluation of reliability improvement in an evolving software product line. In *Proceedings of the 8th Working Conference on Mining Software Repositories*, MSR '11, pages 103–112, New York, USA, 2011. ACM.

[8] L. Ma, C. Artho, C. Zhang, and H. Sato. Efficient testing of software product lines via centralization. *SIGPLAN Not.*, 50(3):49–52, Sept. 2014.

[9] M. Ohira, A. E. Hassan, N. Osawa, and K. i. Matsumoto. The impact of bug management patterns on bug fixing: A case study of eclipse projects. In *28th IEEE International Conference on Software Maintenance*, pages 264–273, Sept 2012.

[10] P. Runeson and M. Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, 2009.

[11] M. Steffens, S. Oster, M. Lochau, and T. Fogdal. Industrial evaluation of pairwise spl testing with moso-polite. In *Proceedings of the Sixth International Workshop on VaMoS*, VaMoS '12, pages 55–62, New York, NY, USA, 2012. ACM.

[12] T. S. Tullis and J. N. Stetson. A comparison of questionnaires for assessing website usability. In *Usability Professional Association Conference*, pages 1–12, 2004.

[13] E. Uzuncaova, S. Khurshid, and D. Batory. Incremental test generation for software product lines. *IEEE Transactions on Software Engineering*, 36(3):309–322, May 2010.

[14] R. A. Virzi. Refining the test phase of usability evaluation: how many subjects is enough? *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 34(4):457–468, 1992.

[15] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction*. Kluwer Academic Publishers, Norwell, MA, USA, 2000.