Anais da 1º Escola Regional Engenharia de Software (ERES 2017)
em Alegrete, RS, Brasil, 20 de outubro de 2017

Towards a Generic Process for SPL Re-engineering

Luciano Marchezan[1], Elder Rodrigues[1], Maicon Bernardino[1], Marcelo Laser[2], Fernando Lima[1]

[1]Federal University of Pampa (UNIPAMPA)
Av. Tiarajú, 810, Ibirapuitã – Alegrete, RS – Brasil

[2]Pontifical Catholic University of Rio Grande do Sul (PUCRS)
Av. Ipiranga, 6681 – Partenon, 90619-900 – Porto Alegre – RS – Brazil

{lucianomarchezan94, marcelo.laser, fernando.fortlima}@gmail.com

elderrodrigues@unipampa.edu.br, bernardino@acm.org

Abstract. *Software Product Lines (SPL) are a well known solution to create reusable software products. Usually a set of product variants is analyzed and commonalities and variabilities are extracted. In this paper, we present our first step towards a generic process to extract those characteristics from product variants. Our process aims to have flexibility to be customizable for different scenarios. We applied a survey and obtained an early feedback which enables us to measure the impact and the contribution of our proposal.*

Resumo. *Linha de Produto de Software (LPS) é uma solução bem conhecida para se criar uma família de produtos de software. Geralmente, um conjunto de variantes de produto é analisado e suas variabilidades são extraídas. Neste trabalho, apresentamos um primeiro passo na direção de um processo genérico para extração dessas características de variantes de produto. Nosso processo busca uma flexibilidade que possibilita sua customização para diferentes cenários. Nós aplicamos um questionário e obtivemos um feedback inicial que nos permite medir o impacto e a contribuição da nossa proposta.*

1. Introduction

While many systems are being developed proactively as a Software Product Line (SPL) [Pohl et al. 2005], there are still those that emerge when a company has one or more software products and wants to create a software product line out of them [Van der Linden et al. 2007]. In this case, should they start from scratch or can they use those software products and transform then into a product line? If they choose to transform their products into a product line, how can they do it? We may find an answer to the latter question in the software product line re-engineering process. The goal of this process is to take a number of product variants and transform them into a SPL with the use of techniques, methods and tools [Assunção et al. 2017]. To the best of our knowledge, a solid and well-known generic process for performing the SPL re-engineering process has not been proposed yet. We find that this kind of process would be a relevant contribution for the Software Product Line Engineering (SPLE) [Van der Linden et al. 2007] field because it may reduce the complexity and effort of SPL re-engineering.

In accordance with the data we found regarding the need for a generic SPL re-engineering process, we have created a generic process for SPL re-engineering. The main

goal of our process is to be customizable and applicable in different scenarios to detect and extract features from a set of product variants. With our process, we intend to provide enough flexibility regarding artifacts, strategies and techniques used for feature retrieval. This flexibility is given by allowing the choice and combination of different techniques for feature retrieval based on information gathered during the process' execution. This information includes: team experiences and skills, domain engineering artifacts, requirements artifacts, product artifact types and extensions, and technologies used to develop the products. Before concluding and executing our process and conducting its evaluation, we sought to obtain feedback regarding the proposal. To this purpose, we have created a survey to assess the impact, contribution and originality of our proposal. We have sent this survey to researchers and professionals in the SPL and re-engineering field and analyzed the results to help us refine and improve our proposal.

The main objective of this paper is propose a generic process for SPL re-engineering. We also justify, explain and analyze the results of the survey we created to assess our proposal. With the survey, we have obtain a good degree of confidence about the impact and relevance of our process. The rest of this work is structured as follows: a background for SPLs and the related work are described in Section 2. Our process is showed in Section 3 and the results of the survey are discussed in Section 4. Lastly, the conclusions and future work will be presented in Section 5.

2. Background and Related Work

Software Product Line: To the best of our knowledge, the product line concept and its characteristics came from the automobile industry, introducing an important characteristics called variability [Pohl et al. 2005]. Variability refers to the flexibility, in other words, the ability to create artifacts to be reused among different products of the same family [Kang et al. 1990]. Some of the main benefits for implementing software product lines are: long-term cost reduction, significant quality improvement of the products and reduction of the time-to-market [Van der Linden et al. 2007]. The field dedicated to the study of this development of SPLs is Software Product Line Engineering.

Software Product Line Engineering (SPLE): Plenty of technologies are used in SPLE as facilitators, making the SPL creation process easier. From these technologies, we can highlight: component-based architecture, software project patterns and the object-oriented paradigm [Kang et al. 1990]. A model for Software Product Line Engineering was proposed by Van der Linden et al. (2007), it is divided between domain engineering and application engineering. Each activity of domain engineering, except product management, generates a high-level artifact that will be used in application engineering. These high-level artifacts are: reference requirements, reference architecture, domain realization mechanisms, and domain tests. They are later specialized in application artifacts: software requirements specifications, software architecture, source code, and test cases. A common practice in the SPL development is to get a set of product variants and extract their variabilities to create the SPL. This process is called Software Product Line Re-engineering [Assunção et al. 2017].

Software Product Line Re-engineering: The term re-engineering can be described as "the examination and alteration of a subject system to reconstitute it in a new form and the subsequent implementation of the new form" [Chikofsky and Cross 1990].

In the SPL context, re-engineering is used to transform a system, system family, or system variants into a software product line as illustrated in Figure 1. According to [Assunção et al. 2017], the SPL re-engineering process is composed of three main phases: detection, analysis and transformation. During the detection phase, the variability points and commonalities of the products are identified and extracted through the use of feature retrieval techniques. The second phase is analysis, where the discovered features are organized as a feature model. The last phase, transformation, is when artifacts linked with these features are managed and modified in order to create the SPL.

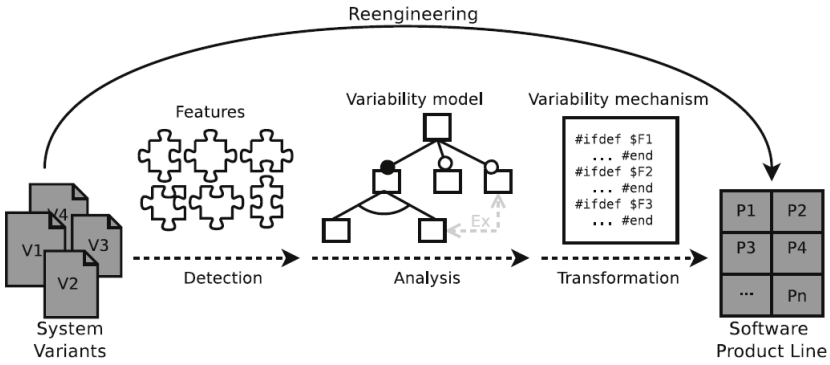


Figure 1. SPL re-engineering process [Assunção et al. 2017]

Related Work: We sought the state of the art of the SPL Re-engineering field considering three aspects of our work: re-engineering phases covered, retrieval techniques proposed or recommended, and the flexibility of the proposal. For the first topic, most of the works had their proposal focusing on the first two phases: detection and analysis [Stoermer and O'Brien 2001] [Rubin and Chechik 2012] [Eyal-Salman et al. 2012] [Damaševičius et al. 2012]. These works, however, are not concerned in offering different options for retrieval techniques. What is showed in those works is a static process for feature extraction and analysis. Regarding the retrieval techniques, we found works that proposed a combination of different techniques in [Beuche 2012] [Martinez et al. 2014] [Rubin et al. 2013]. Still, their focus in a specific strategy, such as information retrieval, different from our work that intends to give different strategies to provide a better adaptability for different scenarios. For the last topic, we found fully flexible proposals in [Krueger and Clements 2013] [Beuche 2012] [Martinez et al. 2014]. These proposals proved that can be used for different situations with the support of tools, different techniques and artifacts. We think, however, that our proposal gives more attention to the planning phase, something we have not found in those works. Nonetheless, those works present a better validation than ours, giving them more reliability. Additional related works can be found in [Assunção et al. 2017].

3. A Generic Process for Software Product Line Re-engineering

Based on the information collected when analyzing related work, we have created our generic re-engineering process for SPL. The process takes a set of product variants or a single product as input, analyzes its artifacts and extracts its features. The process is detailed in the following sections. The process structure can be divided in: 1) Sub-processes: activities performed during the process which contain other activities inside them;

2) Activities: actions performed that use and generate different artifacts; 3) Actors/Roles: which define the actors that will perform an activity; 4) Artifacts: which define the artifacts used in each activity; 5) Techniques: which are the possible techniques used to extract features from the product artifacts.

Concerning the roles, the possible are: 1) Domain Engineer: possesses valid knowledge related to an area of human endeavour, an autonomous computer activity, or other specialized discipline [Kang et al. 1990]; 2) Architect: has valid knowledge related to software architecture, design and architectural patterns; 3) Analyst: requirements specialist; 4) Developer: possesses knowledge of programming languages and technologies; 5) Feature Tester: must know domain glossary, domain constraints, and requirements information; 6) Feature Retriever: must have knowledge about feature retrieval strategies and techniques.

Regarding the artifacts, they can be of two types: input or output. Input artifacts can be categorized as mandatory, optional or alternative. Mandatory are artifacts that must be used in an activity, optional are those that do not have to be used, and an alternative group is one in which at least one artifact must be used. This allows the process to be instantiated in different forms according to the needs of those performing it. Subprocesses and activities are described below.

Process Overview: The process is composed of three main sub-processes: `Planning`, `Detection` and `Documentation Analysis`. `Detection` and `Documentation Analysis` are parallel sub-processes, shown in Figure 2. During `Planning`, team information is gathered to be used during `Documentation Analysis` to make choices about artifact collection. During `Detection`, the features are extracted, categorized and grouped. The extraction is made based on the techniques which are also chosen during `Detection`. During the `Documentation Analysis`, the re-engineering documentation is collected and compiled. All the activities of this subprocess are optional and they can be performed at the same time of any activity from `Detection`.

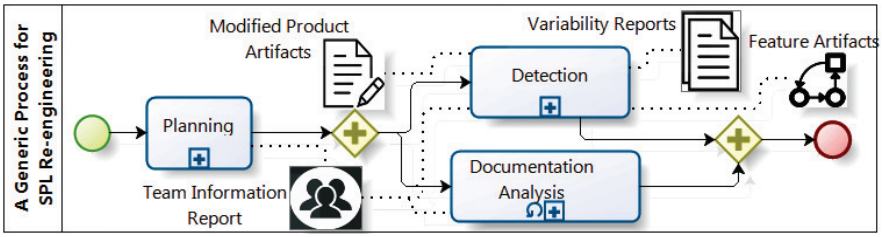


Figure 2. Process Overview

Planning: The first activity of this sub-process is to `Collect Team Information`, as shown in Figure 3. During this activity, information about the team that will execute the process is collected. This information includes experience, skills, knowledge and preferences of each member. The information is registered in a document, and stored in a database. This activity can be performed by any actor and has two optional inputs: `Project Plan` and `Business Organization Chart`. Its output is the team information report. The second activity is to `Define Roles and Tasks` based on the information collected on the previous activity. Here, all actors participate

and the team information report is updated.

Detection: The first activity performed here is `Feature Search`, which is a sub-process shown in Figure 4. During `Feature Search`, strategies and techniques are chosen and applied in product artifacts to find and extract features. To decide which strategies and techniques must be used, the team information report and the documentation generated during the `Documentation Analysis` are analyzed.

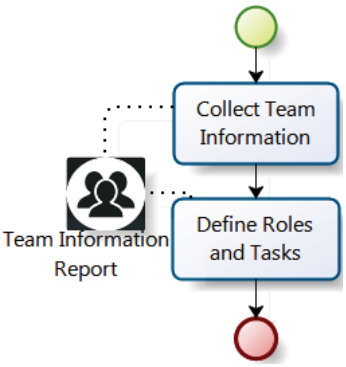


Figure 3. Planning Sub-process

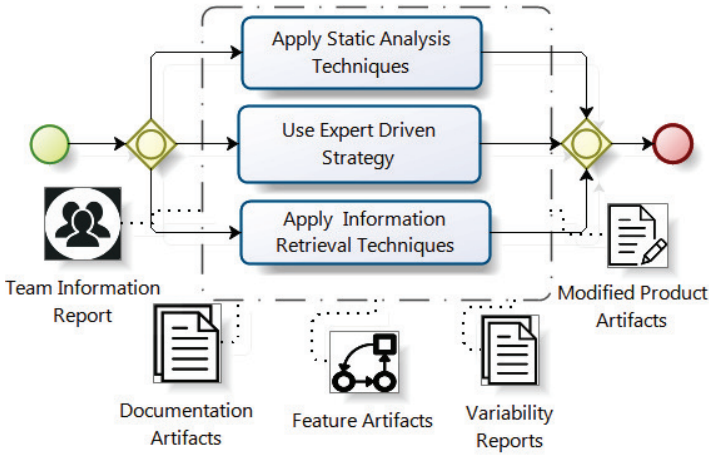


Figure 4. Feature Search Sub-process

`Detection` generates three outputs: 1) Variability reports: written in natural language, contain information about the feature extraction results; 2) Feature artifacts: outputs generated from the feature retrieval techniques, such as dependency trees and feature descriptions. 3) Modified product artifacts: product artifacts modified to include feature entry points (*e.g.* comments, markings, and tags.). These modifications do not have impact in the products execution. `Detection`, however, may not be singly performed; it is executed along side `Documentation Analysis`.

Documentation Analysis: The activities of this sub-process are optional, and are performed (or not) based on retrieval techniques, strategies, and artifacts available. All activities of `Documentation Analysis` can be performed at the same time of any activity from `Detection`. During `Documentation Analysis`, domain information is collected, a domain glossary is created, requirements information of the products are gathered, and information about architecture, technologies and artifacts is registered. The main contribution of this sub-process is to serve as a support process for `Detection`. This is done by using the documentation collected and registered in `Documentation Analysis` to make decisions, extract data, and guide the feature retrieval activity.

4. Assessing Our Process

We have applied a survey[1] to researchers and professionals of the SPLE and Re-engineering field to evaluate the impact, relevance and originality of our proposal in the SPL re-engineering field. We believe that this early feedback may give us information

[1]The details of the questionnaire instrument of the survey can be found at `http://tiny.cc/eres2017`

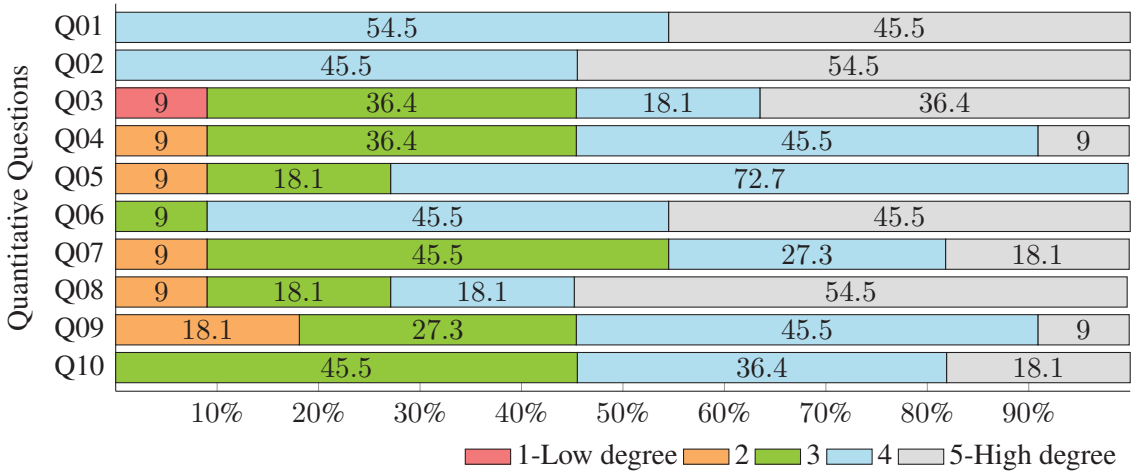
needed to improve and expand our process. In the following sections we explain the survey and discuss the results.

Survey Structure: We split the survey in six sections, intending to better organize the information and get better understanding from the participants. The first section is an introduction to the survey, briefly showing the goals of our research as well as the goals of the survey itself. The second section of the survey has questions regarding the participants' data such as academic degree, knowledge and experience in the SPLE area. The third section has our process documentation and is responsible for giving the information the participants needed to answer the later questions. The fourth and fifth sections contained quantitative questions to assess our process. We have used the Likert Scale to give participants options to rate the answers based on their opinion where the number 1 represents a low degree of agreement and 5 represents a high degree of agreement with the question. The last section was created to ask for feedback regarding the survey, such as clarity of the questions, comments about the survey, and comments about our process.

Participants' Profile: To answer our survey, we considered researchers and professionals that are working in either the SPLE or the re-engineering field. Within the survey, we included questions to obtain information about the participants, such as knowledge, skills and experience. The total number of answers was eleven, which we consider a small amount of participants; however, it do not reduces the importance of the survey. Regarding the experience with SPL, we have asked how many years the participants have been working or researching in the SPL engineering field. The answers proved to be satisfactory for us because 64% of the participants have been working or researching in this field for 1 to 5 years. We have also asked about the participants technical knowledge related with topics of interest for this work. The great majority of the participants answered to have technical knowledge regarding SPL, Re-engineering and Process Documentation. Less than a half of the participants also express to have knowledge about Feature Retrieval Techniques and SPL Re-engineering.

Survey Questions: Our survey had a total of 10 questions, the quantitative results are showed in Figure 5. In the first question, we asked if a generic process for performing SPL re-engineering would be a relevant contribution to the field (**Q01**). For the second question, we asked about the relevance of customization and applicability of the process in different scenarios (**Q02**). For both question, we got good results as shown in Figure 5. In the third question, we asked if our process is a novel contribution to the field (**Q03**), where we obtained mixed answers. The fourth question was related with the impact of our proposed process (**Q04**). With the following question been related with the flexibility of our process to collect information about the team, assign roles and define tasks (**Q05**). For these three question, we obtained a high level of acceptance. For the next two questions, we have asked exclusively about the artifacts documented within our process. The first question was regarding the relevance of a process that provides flexibility for the artifacts used (**Q06**). The second question was if the participants thought that our process is a novel contribution in regards only to its flexibility related with artifacts (**Q07**). Similar to the two questions above, we have asked if, considering the flexibility related with strategies and techniques for feature retrieval, a SPL re-engineering process is a relevant contribution (**Q08**). We have also asked if, regarding the flexibility related with strategies and techniques for feature retrieval, our process is a novel contribution to the SPL re-

engineering field (**Q09**). For these four question, we have obtained another mixed set of answers. The last quantitative question was related with the applicability of our process (**Q10**). For this question, we obtained once more a mixed set of answers.



PS: Some percentages generate a recurring decimal which do not result in a 100% total.

Figure 5. Frequency Diagram of the Quantitative Questions

Results Analysis: By analyzing the results of the quantitative questions we have concluded that our process is on the right direction towards our main goal. We could notice that there is still some lack of flexibility in some points, which we will work to reduce. Despite these problems, we could obtain a high degree of confidence related with the impact, relevance and originality of our proposal. For the first topic, we could see with the results of the questions that a generic process that provides flexibility for SPL re-engineering would make impact in the field. This indicates that the motivation for creating our process makes sense and we have been doing a significant research in the area. Regarding the relevance of our proposal, we could also obtain satisfactory results showing that our proposal is being developed in the right direction. In the last section of the survey, we have placed two open questions to give participants the opportunity to leave comments and suggestions for our process. Most of the suggestions were related to expanding the flexibility of our proposal. The participants think that we should improve some aspects of the documentation to obtain more clarity about how flexible the process really is. Another suggestion was related with the process being specific in some points such as artifacts used by some retrieval techniques. Some participants suggested that we define these techniques in a generic way and provide supporting tools and methods as a recommendation for executing each technique.

5. Conclusion

In this work, we have presented a generic process for software product line re-engineering. Our proposal aims to give enough flexibility for those that may execute it to reduce cost and effort when performing the re-engineering process. This flexibility is made possible through the use of optional and alternative input artifacts, different strategies and techniques for feature extraction, and by making some of the activities optional. With the application of the survey, with quantitative questions, to evaluate our proposed process,

we obtained an early feedback from researchers and professionals of the SPLE and Reengineering field. The results of the survey showed that our proposal may be a relevant contribution in the area. This was the most important information derived from the survey's feedback because we want to have enough confidence before moving forward with our research. We could also see that some improvements have to be made regarding the flexibility of our process, which we plan to improve in future works. We also intend to validate the proposal through a case study in a real development environment.

References

Assunção, W., Lopez-Herrejon, R., Linsbauer, L., Vergilio, S., and Egyed, A. (2017). Reengineering legacy applications into software product lines: a systematic mapping. *Empirical Soft. Engineering*, pages 1–45.

Beuche, D. (2012). Modeling and building software product lines with pure:: variants. In *Proc. of the 16th Inter. SPL Conf. - Volume 2*, pages 255–255. ACM.

Chikofsky, E. and Cross, J. (1990). Reverse engineering and design recovery: A taxonomy. *IEEE software*, 7(1):13–17.

Damaševičius, R., Paškevičius, P., Karčiauskas, E., and Marcinkevičius, R. (2012). Automatic extraction of features and generation of feature models from java programs. *Information Technology And Control*, 41(4):376–384.

Eyal-Salman, H., Seriai, D., Dony, C., and Al-msie'deen, R. (2012). Recovering traceability links between feature models and source code of product variants. In *VARiability for You Workshop: Variability Modeling Made Useful for Everyone*, pages 21–25. ACM.

Kang, K., Cohen, S., Hess, J. A., Novak, W., and Peterson, A. (1990). Feature-oriented domain analysis (FODA) feasibility study. Technical report, DTIC Document.

Krueger, C. and Clements, P. (2013). Systems and software product line engineering with biglever software gears. In *Proc. of the 17th Inter. SPL Conference co-located workshops*, pages 136–140. ACM.

Martinez, J., Ziadi, T., Klein, J., and Le Traon, Y. (2014). Identifying and visualising commonality and variability in model variants. In *European Conference on Modelling Foundations and Applications*, pages 117–131. Springer.

Pohl, K., Böckle, G., and van Der Linden, F. (2005). *Software product line engineering: foundations, principles and techniques*. Springer Science & Business Media.

Rubin, J. and Chechik, M. (2012). Combining related products into product lines. In *Inter. Conf. on Fundamental Approaches to Software Engineering*, pages 285–300. Springer.

Rubin, J., Czarnecki, K., and Chechik, M. (2013). Managing cloned variants: a framework and experience. In *17th Inter. SPL Conference*, pages 101–110. ACM.

Stoermer, C. and O'Brien, L. (2001). Map-mining architectures for product line evaluations. In *Working IEEE/IFIP Conf. on Software Architecture*, pages 35–44. IEEE.

Van der Linden, F., Schmid, K., and Rommes, E. (2007). *Software product lines in action: the best industrial practice in product line engineering*. Springer Science & Business Media.