Multiple View Interactive Environment to Analyze Software Product Line Tools

Kattiana Constantino, Eduardo Figueiredo, Glauco Carneiro and Raquel Minardi
Software Engineering Laboratory (LabSoft), Department of Computer Science
Federal University of Minas Gerais (UFMG), Belo Horizonte, MG, Brazil
Universidade Salvador (UNIFACS), Salvador, BA, Brazil
{kattiana, figueiredo, raquelcm}@dcc.ufmg.br, glauco.carneiro@unifacs.br

ABSTRACT

Software Product Line (SPL) relies on the development of a collection of information systems from a shared set of software assets. In this context, the adoption of SPL in the industry relies heavily on tool support. This paper presents Vi**SPL**atform, a visualization environment aimed at portraying data related to experiments focusing on SPL tools. We conducted a preliminary evaluation to analyze to which extent the platform is effective to support the understanding of characteristics of the SPL tools. The results of this study show that Vi**SPL**atform can somehow indicate strengths and improvement opportunities in the analyzed SPL tools. For instance, they show that *Automatic Analysis* is a strength of the SPLOT tool and that *Interface* is an improvement opportunity. (*Better if read in colors*).

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: [Miscellaneous]; D.2 [**Software Engineering**]: [Miscellaneous]

General Terms

Software Engineering and Information Visualization

Keywords

Software product line tools, information visualization.

1. INTRODUCTION

Software Product Line (SPL) is a set of software systems sharing a common, managed set of features that satisfies the specific needs of a particular market segment [14]. A feature represents an increment in functionality or a system property relevant to some stakeholders [1]. An important concept of an SPL is the Feature Model (FM) which is used to represent the common and variable features in SPL [6].

The number of empirical studies focusing on SPL tools published in the literature has increased over the last years [5] [12] [8]. The analysis of data from experiments can be complex, involving many variables and possible crossings that could lead to a high number of possible analyzes. These aforementioned factors characterize the knowledge discovery as not a trivial task. The literature has provided evidences that visualization resources and techniques are effective to meet user needs regarding the analysis of specific data sets [16]. The primary goal of visualization is to convey information in an understandable, effective, easy-to-remember way [7]. This paper proposes the use of a multiple view interactive environment called Vi**SPL**atform [18]. The interactive visual representations and analytical interaction techniques focused on empirical data analysis to help data analysts to comprehend and explore data from research methods in the SPL tool context.

The remainder of this paper is organized as follows. Section 2 presents SPL concepts. Section 3 introduces the purpose of our visual platform composed by the visualization techniques presented and explains in more detail each technique applied. Section 4 briefly gives an overview about an empirical study setting using a SPL tool which the empirical data was collected to provide the data set for this current work and outlines a preliminary evaluation. Section 5 discusses some related works. Finally, Section 6 concludes the paper and outlines future directions for this work.

2. SPL DEVELOPMENT

The SPL development involves modeling features that represent different viewpoints, sub-systems, or concerns of the software system [1]. Figure 1 depicts an example of FM. In feature models, nodes represent features and edges show relationships between parent and child features [1]. These relationships define how the features can be combined. There are common features found in all products of the product line (known as mandatory features) and variable features that allow distinguishing between products in a product line. Variable features define points of variation and their role is to permit the instantiation of different products by enabling or disabling specific SPL functionality. Generally, these variable features are represented by optional (“OR”) or alternative (“XOR”) features. As shown in Figure 1, the nodes are represented by boxes and connections between them are represented by edges.

According to [13], 41 tools were identified to support SPL development and variability management. These tools support the representation and management of reusable artifacts instead of providing means for conventional development from scratch. Considering this scenario, selecting the

SBSI 2016, May 17th-20th, 2016, Florianópolis, Santa Catarina, Brazil

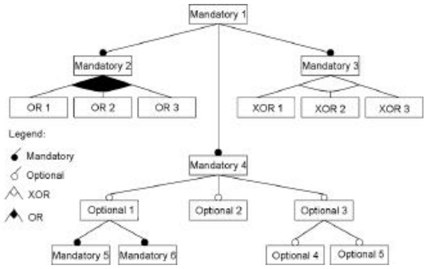

Figure 1: Example of a feature model.

tool that best fit the SPL development goals is not as trivial as it might seen.

3. VI**SPL**ATFORM

Following the reference model presented in Figure 2, the *Raw Data* to be conveyed in ViSPLatform comes from artifacts of SPL tools experiments such as study protocol, participants knowledge background, work experience, tasks to be performed and the corresponding results.

3.1 Vi**SPL**atform Architecture

The *DataTables* in Figure 2 is obtained from raw data from the experiments after conversion to both JSON and Comma Separated Values (CSV) files. In the case of the JSON format, the file contains an object list with a set of empirical SPL tools data. At this point, data representing *real attributes* (attributes related to the target domain, in this case characteristics of SPL tools) is structured and organized for the next step as a *Visual Structure*. As presented in Figure 2, the *Visual Structure* provides data to feed the views with visual attributes. Real attributes are related to the domain that is the focus of the user tasks, in this case, experiments regarding SPL tools. Visual attributes are related to the visual representation provided in the views, such as position, color and shape of visual elements. The visual structure in Vi**SPL**atform is implemented with JavaScript language. The visual structure enables the use of the mapping described before to set the appearance of the views and to handle the behavior as a result of captured events performed by the user.

In Figure 2, *Views* correspond to the user-interface in HTML which receives the visualization techniques. Thereby, Vi**SPL**atform manages visual models for one or more data sets, separating visual attributes (location, size, color, etc) from the abstract data. One or more views provide a graphical display of the visualization, while control modules process user interaction which can affect change at any part of the application. To implement views in Vi**SPL**atform, we used the analytical data visualization tool called *Data-Driven Documents*[1] (D3) [2], a JavaScript library for manipulating documents based on data. This library provides resources to set and manipulate attributes of graphical elements in Scalable Vector Graphic (SVG), which is a XML-based vector image format for two-dimensional graphics with support for interactivity and animation. Moreover, we used Cascading Style Sheets (CSS) that is a style sheet language used for describing the presentation of a document written in a markup language. Finally, we built the platform using components provided by the Bootstrap Framework[2], which includes several stylesheets and jQuery plugins for establishing interactive web sites or application user interfaces. All these technologies were employed for realizing a dynamic exploration and visualization experience.

3.2 Raw Data of SPL Tools

In this section, we describe the *Raw Data* used by ViSPLatform. Firstly, we explain the functionalities of SPL tools. After that, we list the issues related to strengths and improvement opportunities of SPL tools. Finally, to provide better contextualization, we characterize the participants of an empirical study focusing on SPL tools described in Section 4 on this paper.

SPL Tool Functionalities. Based on previous work [13] the following functionalities of SPL tools were considered in this paper: (i) *Feature Model Edition.* Edit a representation of all products of the SPL in terms of features. (ii) *Automated Feature Model Analysis.* It can be defined as the computer-aided extraction of information from FMs. (iii) *Product Configuration.* A product is declaratively specified by selecting or deselecting features according to stakeholder's preferences. (iv) *Feature Model Import/Export.* It is how the FM can be imported or exported in XML, CSV or other format and how to integrate the FM in a new project.

Strengths and Improvement Opportunities of SPL Tools. We want to know what are the strengths and improvement opportunities of a SPL tool faced by participants during the execution of tasks. In particular, we ask the participants about the following items: (i) *automatic organization*, (ii) *automatic analysis*, (iii) *editor*, (iv) *examples available*, (v) *hot keys*, (vi) *integration with other tools*, (vii) *interface*, (viii) *persistence models*, (ix) *product configuration*, (x) *tutorials and users guides*. In such a way, we have knowledge about the percentage of responses for each mentioned items. In addition, we can make a comparison on what are the top rated as strengths and improvement opportunities of SPL tools.

Background of the Participants. We used a questionnaire to collect the following data about the background of each participant: id, gender, tool used, university that each participant is enrolled, and level of different skills, such as Object-Oriented Programming (OOP), Unified Modeling Language (UML), and Software Product Line (SPL). For these skills, participants could claim to have high, medium, low, or no knowledge. In addition, the questionnaire also collected data regarding their Work Experience (WE) that could be classified as: more than 3 years, 1 to 3 years, up to 1 year, and never worked in software development industry.

3.3 Visual Encoding of Data Tables

In this section, we describe the mapping of raw data presented in Section 3.2 into visual attributes to be conveyed in views.

Background of Participants. The background of participants was represented through the area and color visual attributes of the bubbles chart view. In this way, the color can be mapped to a real attribute while the area to another. The bubble chart view was selected among other views such

[1] http://d3js.org/

[2] http://getbootstrap.com/

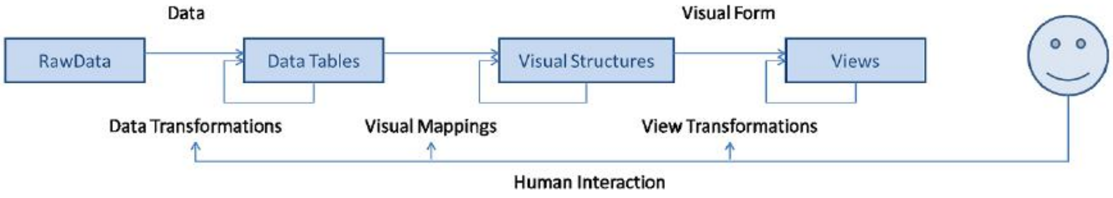

Figure 2: The Card Visualization Reference Model [4]

as circle packing or hierarchical edge bundling [2]. The reason behind the choice of the bubbles view is the possibility to provide both a panoramic and detail visual scenario of the participants that took part in the experiment. For example, we can on one hand identify using the filter to group the participants enrolled by university. And in the other hand, we can spot the knowledge of participant and its level with mouse over a bubble.

SPL Tool Functionalities/Tasks. The goal is to know the percentage of results for each task performed by participants. In other words, we want to know how the percentage of participants that complete the task with success, minor trouble, major trouble, or participants that were unable to complete the task. To represent these percentages we use donuts and divergent stacked bar chart in different views. Each slice represent one of these results. We use color to differentiate each slice. As a result, we can have an overview about each task and its performance.

Strengths and Improvement Opportunities of SPL Tools. Strengths and improvement opportunities are divergent data. Thus, we can use position and color to represent these two sides. We choose to use the Diverging Stacked Bar Chart for presenting results of rating scales. However, there are other ways to plot data as Likert or rating scale. These graphical forms include tables, bar charts of means, grouped bar charts, divided bar charts, ribbon charts, multiple pie charts, waffle plots, and radar plots [15].

The following sections present some questions we would like to answer with the tool and present the proposed visual representations as well as the analytical interaction techniques that can be used with them. We explain the rational behind the project of the techniques and discuss some analysis that can be done through them.

3.4 The Bubbles Chart View

Knowing the profile of the participants is a very important information in empirical studies. Before any analysis and conclusions, it is essential to have an overview of the characteristics of the group of people involved. In our empirical study, we mapped the "*background of participants*". We collected data about this background through a questionnaire.

We have represented several possible background information of the different groups of people (genders, universities, etc.) through a Bubble chart. Each bubble represent a participant and the areas of the circles encode levels of experience. Bubbles can then be grouped using different possible categorical attributes that segment the data. It is based on the Gestalt principle of proximity and shows if the background of the participants is similar in each group. For instance, Figure 3 shows the participants sorted by groups.

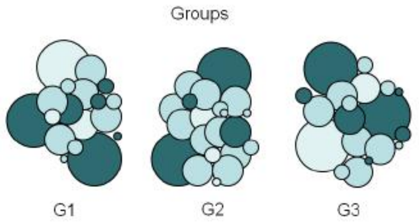

Figure 3: Participants sorted by group.

The visualization is interactive and data analysts can choose different attributes to group data. For each new selection, the bubble diagram performs a transition. Data analysts have two ways to interact with the visualization. Figure 4 depicts one of the interactions, when a data analyst run mouse over a bubble. In this case, a legend appears with extra information about the specific participant. Second, data analysts can rearrange the bubbles grouping by some options available in a selection list. When one of the options is chosen, the bubbles are separated and grouped into different regions of the screen according to the selection.

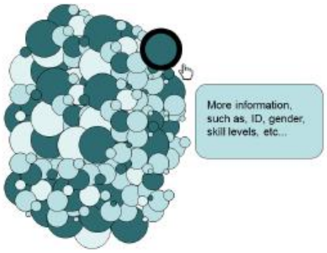

Figure 4: Background of participants

3.5 The Heatmap and Donut Chart Views

A remarkable aspect of the empirical data is the possible correlations that can be established between the background of participants and the quality of the tasks done, the level of difficulty faced or the number of mistakes. This aspect is one of the most interesting and, for this reason, we propose two different visualizations interactively linked. Figure 5 shows an example of the heatmap which is a matrix-based display that simultaneously reveals correlations between the values in a first attribute (encoded as rows) and the second one (depicted as columns). The quantitative values of the attributes are encoded as colors in different intensities. In other words, it consists of a set of cells, each colored based on a quantitative scale corresponding to attributes of the matrix.

In a usual heatmap, cells are colored based on a real number scale or can be discretized if necessary. Instead, in our heatmap, we used a divergent color scale going from red to blue to encode failure and success in task completion. We also decided to combine the colors of the cells for the whole set of participants related to the specific cell to depict a summary of the results per cell. Notice that, for each cell we have a number of participants and four different possible levels of difficulty. For this reason, we linked a donut chart presenting the part-to-whole quantitative relationship of this variable. Hence, data analysts can not only have more information about the results represented by each cell but also have a general view about the results presented in all the combinations of tasks and background in the matrix.

Figure 5 shows the map of each level of knowledge versus results of the tasks performed by participants of an experimental study. Each cell is colored with the combination of colors by percentage of results. For example, if we want to evaluate the impact for "tool 1" of participants with "skill 5" knowledge and their performance on the "task 4", data analysts can move the mouse over a specific cell and a donut chart with the result task of these participants appears showing details about the proportions of each result. In addition, the center of the donut shows the percentage of answers of the participants that have this knowledge and perform this task using "tool 1".

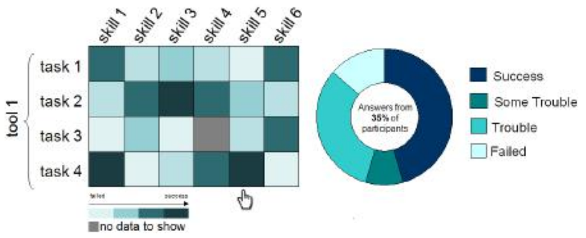

Figure 5: Example of Heatmap and Donut

3.6 The Bar and Donut Charts Views

The heatmap and donut charts provide a general view of the data set and the results obtained by an empirical study. However, a possible limitation is that they do not combine skills or tasks in an *ad hoc* manner and see how these variable can impact on the results obtained by that participant in the performed test. With this objective, we proposed an interactive combination of bar and donut charts to show at first glance both the distributions of the different backgrounds and the proportions of the results obtained in each task. Taking into account the human perceptual system and specially the memory limitations when charts to be compared are distant, we presented the whole set of charts organised in a dashboard where data analysts can compare and contrast results in a straightforward manner.

Figure 6 shows the overview of this dashboard that depicts the background versus task results. This figure shows the task results, knowledge, tasks and tools for all participants. Our goal is to analyze whether the background of participants impacts on the use of the analyzed tools and, as the dashboard is highly interactive, different combination between the skills can be done in order to analyze the results of tasks in each tool. For instance, supposing that we want to evaluate only the results obtained by highly skilled participants, we can select this level in all possible categories (backgrounds) and analyze the results obtained by only this group on the respective donut charts.

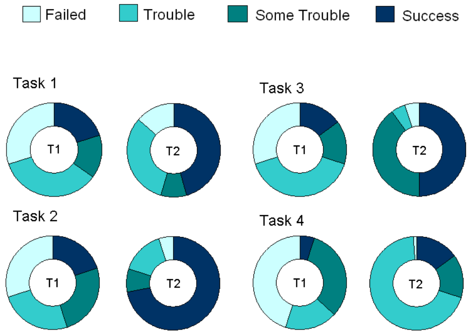

Figure 6: Impact on the use of each tools

In Figure 7 outlines the background of participants. Figure 7(a), each bar is intended to show the percentage of skill levels of participants. We used bars to encode the respective values (their distribution) and visually reinforce the independent nature of these knowledge and their levels. Figure 7(b) shows that the data analyst can interact by selecting and deselecting any bar representing each possible level of each of the possible types of knowledge.

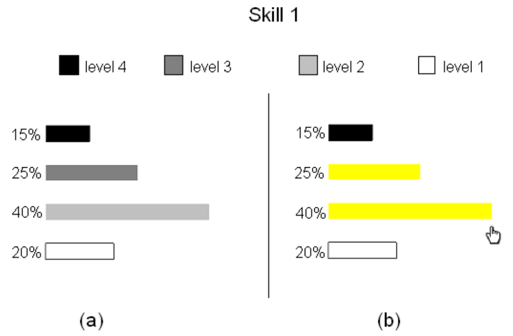

Figure 7: Selecting (a) or Deselecting Background (b)

Figure 8 shows donut charts which are later used to summarize the task results. This figure depicts the percentage of result task performed by participants. Besides, each donut chart is intended to summarize the results of one task in one specific tool. The legend in the center of each donut is to identify the matching tool. For instance, *"T1"* means "Tool 1" and *"T2"* means "Tool 2".

We used donut charts to represent the results of each task by tool. We know that donuts are as less accurate than bar chats [17]. Besides, we did not show the percentage in each slice to avoid visual pollution and too many information. Nevertheless, Figure 8 shows that the data analyst can interact with specific slices by mouse overing events to see tooltips with the result and its percentage.

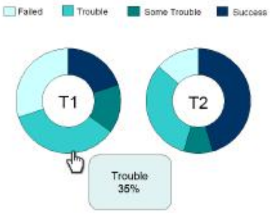

Figure 8: Participants with some knowledge

3.7 The Diverging Stacked Bar Chart View

Figure 9 illustrates a diverging stacked bar chart. This chart provides an effective way to communicate summaries of data collected with Likert and other rating scales [15] [10]. In our study, we used this visualization to illustrate if two different scales correlate in an empirical study. First, we used the diverging stacked bar chart to present the results of tasks when we asked about the problems faced during the tasks performed by participants. In Figure 9, we considered *Failed* and *Trouble* as more difficult to perform the task (negative percentage), shown on the left of the zero line. We considered *Some Trouble* and *Success* as easier to perform the task (positive percentage), shown on the right of the zero line.

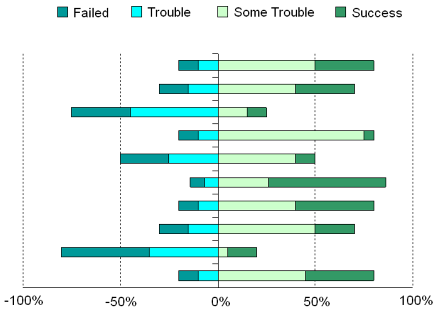

Figure 9: Diverging Stacked Bar Chart for presenting results of rating scales

Second, we used the diverging stacked bar chart to present the participants answers about strengths and improvement opportunities of each tool. We considered strengths as positives and improvement opportunities as negatives points that the participants faced during the tasks.

4. PRELIMINARY EVALUATION

This section presents an preliminary evaluation aiming to illustrate the applicability of ViSPLatform. Section 4.1 describes the configuration of an empirical study about SPL tools. Section 4.2 reports the problems encountered by participants when performing the requested tasks using SPLOT [11]. Section 4.3 focuses the discussion on whether the background of participants can impact on the use of SPLOT. Finally, Section 4.4 discusses the strengths and improvement opportunities of the SPLOT tool.

4.1 Study Settings

This section presents the study configuration of an preliminary evaluation aiming to illustrate the applicability of ViSPLatform. To this end, we collected the data from an empirical study of one SPL tool, namely Software Product Lines Online Tools (SPLOT) [11]. SPLOT is a Web-based tool for creating and sharing feature models and product configuration. It is a free and open source project. In addition, this section explains the training session and tasks assigned to participants in this evaluation.

We used the Bubbles chart to represent the participants involved in this study. They are 41 young developers enrolled in courses related to the Software Engineering area. Figure 10 shows a visualization of these participants. All are graduated or close to graduate since they are mostly participants post-graduated M.Sc. and Ph.D. students from four different academic institutions.

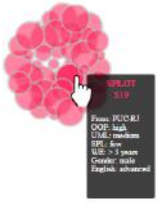

Figure 10: SPLOT Participants

The participants were nicknamed as S1 to S41. Our goal is to use these nicknames while keeping the anonymity of the participants, since we did not repeat participants in the experiments. We performed an analysis of four functionalities provided by SPLOT (Section 3.2).

Further details about the distribution of participants, background questionnaire, and additional information about the participants (id of each participant, gender, the SPL tool used, and University that each participant is enrolled) can be observed by using ViSPLatform via its website [18].

4.2 Problems Faced by SPLOT Participants

Our goal in this section is to analyze the problems that developers may have to carry out tasks in SPLOT. In other words, we aim to answer the following research question.

RQ1. *What functionalities are hard and easy to use?*

For this evaluation, participants answered a questionnaire with closed questions (Section 4.1) asking for the completion of the tasks. After each task of the experiment, participants had the following options to answer: (i) *I was unable to perform*, (ii) *I performed with major problem,* (iii) *I performed with minor problem*, and (iv) *I had no problem performing the task*. In order to answer the research question RQ1, we rely on data presented in Figure 11. This visualization summarizes the results grouped by functionality and tool. We

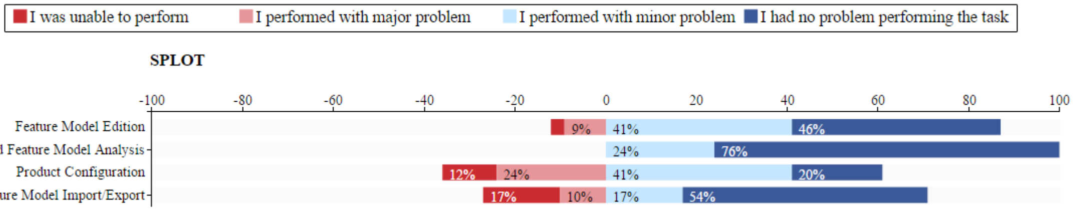

Figure 11: Percentage of problems reported by participants to complete their tasks

defined a Y-axis to quantify the cumulated results, where the negative values mean "hard to use" and positive values mean "easy to use" the respective functionality. Thereby, the general observation is that participants had minor problems or no problem to perform the tasks using SPLOT. We also analyzed other characteristics of these results in this section. Some conclusions were also supported by qualitative feedback from the study participants.

Product Configuration seems the most challenge functionality to use by SPLOT participants. The visualization presented in Figure 11 shows that about 12% of participants were unable, and 24% had major problems to perform the Product Configuration task. On the other hand, 24% of participants of SPLOT had minor problems and 76% performed without problems the Automated Feature Model Analysis task. These results endorse one major goal of this SPL tool, which is to support developers with automatic statistic computation [11], such as depth of the feature tree, number of possible configurations, and others. SPLOT also focuses on critical debugging tasks, such as checking the consistency of feature models and detecting the presence of dead and common features.

4.3 Background Influence

Our goal is to analyze whether the background of developers can impact on the use of SPLOT. In other words, we aim to answer the following research question.

RQ2. *Does the background of developers impact on the use of the SPLOT tool?*

In order to answer RQ2, Vi**SPL**atform presents two different visualizations interactively linked, heatmap with donut chart. Figure 12 shows the heatmap of each level of knowledge versus results of the tasks performed by participants of our experimental study. Finally, we used a divergent color scale going from red to blue to encode failure and success in the task completion in each cell.

In this visualization, data analysts can not only have more information about the results represented by each cell but also have a general view about the results presented in all tasks and skill levels in the matrix. Analogously, the visualization about the problems faced by participants (Figure 11), the data analyst can see with more details the task results of each level skill. Thereby, there are a great number of success or some troubles (blue intensity) for the *Automated Feature Model Analysis* task for SPLOT. Note that, there is no participants who claimed to have "no" knowledge in OOP for SPLOT. In this manner, this insight was not easily visible in the other visualization techniques. In these two cases, when data analysts interact with any one of these cells (gray cell) only a message "no data to show" appear for them. The limitation of this visualization is that the data analyst can not combine different skills level to analyze the background influence on the tasks. Thereby, we proposed another highly interactive visualization presented in Vi**SPL**atform, with different combination between the skills, in order to analyze the results of tasks (RQ2).

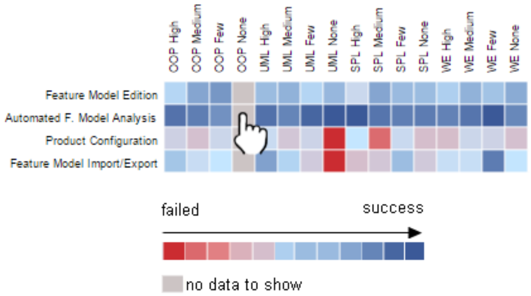

Figure 12: Level of knowledge versus results of the tasks performed by participants

For instance, we can classify the participants by their level of knowledge and work experience. Group 1 (Strong Background) includes participants that claimed to have high and medium knowledge in OOP, UML, and more than 1 year of work experience. Group 2 (Weak Background) includes participants that answered few and no knowledge in OOP, UML, and less than 1 year of work experience. Figure 13 shows donut charts summarizing the results. Similarly to Figure 11, this figure depicts the percentage of participants who performed tasks. Charts on top indicate results for participants in the strong background group (32%) and charts on the bottom of Figure 13 indicate participants with weak background (18%). Besides, each donut chart summarizes the result of one task in one specific tool.

Based on the visualizations of Figure 13, we compared these two groups for each dimension. For Feature Model Edition, for instance, we realized that SPLOT showed some differences between these two groups. That is, about 11% of participants with weak background (Group 2) reported they were unable to conclude their tasks while 99% of strong background participants complete their tasks related to Feature Model Edition. In addition, the percentage of participants who had few problems or had no problem did not change from Group 1 to Group 2. The reason for this result may be due to the Web interface of SPLOT and for these participants of both groups do not seemed familiar with it.

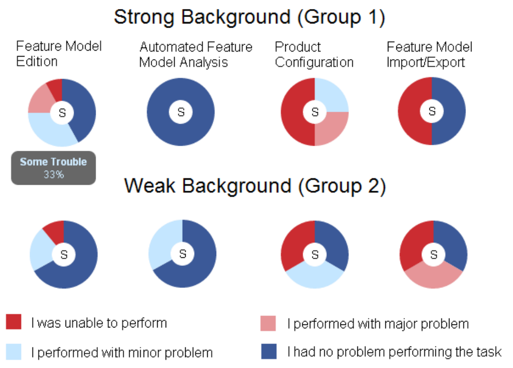

Figure 13: Strong and weak backgrounds

Therefore, we noticed the percentage of success is related to the skill level of participants in these cases. Good knowledge in OOP and UML may have contributed positively to success of participants in Feature Model Edition task because this task of editing a feature model involves creating an abstract representation and relationships, similarly to software modeling with UML.

With respect to Product Configuration, in spite of SPLOT has been developed with focus on this task [11], both groups had similar performance. However, Group 2 had 33% failures, while Group 1 had only 13%. The configuration of a product seems a simple activity. At first glance, this task seems to require only selecting or deselecting features of the model. However, other knowledge about the SPL model is also important, such as comprehension of the notations, the relationships between features, and constraints.

As shown in Figure 13, for Feature Model Import/Export, participants who used SPLOT presented big difference in the results when comparing both groups. The percentage of failures increased from 17% in Group 1 to 33% in Group 2. Although the repository of model is an interesting functionality of this tool, the participants of this study seem not familiar with it. Hence, it was difficult for participants with weak background to perform this task in SPLOT. However, this task is very common for experienced software developers, what may have contributed positively to the task in Group 1.

4.4 Strengths and Improvement Opportunities

Our goal is to investigate some of the strengths and improvement opportunities of SPLOT. In other words, we aim to answer the following research question.

RQ3. *What are the strengths and improvement opportunities of SPLOT?*

Figure 14 shows diverging stacked bar chart of the strengths and improvement opportunities of SPLOT. This visualization is supported by Vi**SPL**atform. The items about strengths and improvement opportunities, presented in Section 3.3, are sorted in alphabetical order. Participants could also freely express about other strengths or improvement opportunities they encountered during the tasks.

About 47% of its participants pointed the interface as the biggest improvement opportunity. 37% indicated the hot-key as a problem. In addition, 24% of them pointed product configuration and examples available as problems to understand the SPLOT tool, as shown in Figure 14. Although SPLOT is not a development tool, the integration with source code was another improvement opportunity pointed freely by participants. On the other hand, the two most voted strengths were: automatic analysis of the models (76%) and editor of model (61%). For other participants, 32% indicated that the tool interface was easy and intuitive and 24% indicated the product configuration as an other positive point. Other strengths pointed freely by participants are the online tool and repository for sharing the models.

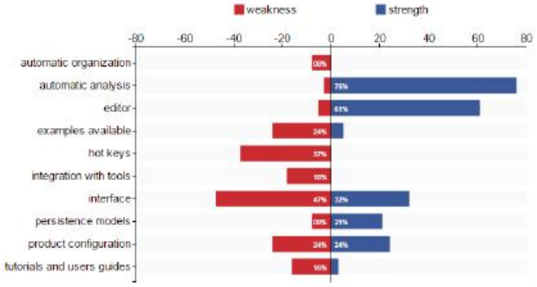

Figure 14: Strengths and improvement opportunities of SPLOT

4.5 Threats To Validity

This section discusses the different types of threats to validity of our preliminary evaluation with respect to the four categories [19]: construct validity, internal validity, external validity, and conclusion validity.

Construct validity. This type of threat can occur in formulating the questionnaire in our experiment, although we have discussed several times the experiment design. To minimize social threats, we performed the experiment in different institutions. *Internal validity.* A limitation of our preliminary evaluation concerns the absence of balancing the participants in groups according to their knowledge. It can be argued that the level of knowledge of some participants may not reflect the state of practice. To minimize this threat, we provide at least 1.5 hour training session to introduce participants to the basic required knowledge and a questionnaire for help the better characterize the sample as a whole.

External validity. A major external validity can be the selected tool and participants. We choose SPLOT tool, among many available ones, and we cannot guarantee that our observations can be generalized to other tools. About participants, we tried to minimize this threat by working with random participants from various groups. We worked with both new and experienced developers. They are graduated or close to graduate since the course targets post-graduated M.Sc. and Ph.D. students. *Conclusion validity.* The conclusion analyzed could be another if it was done by other researchers or, if it was done by just one of us. To minimize this threat, we discuss the results data to make a more reliable conclusion.

5. RELATED WORK

Visualization techniques have been used in software engineering as a support for understanding, maintaining and evolving software systems [7]. In the following sentences we present some tools that use visualization to support activities related to SPL.

Visual and Interactive Tool for Feature Configuration [3] is a visual and interactive tool for product configuration. It was based on a meta-model which specifies the main entities such as decisions, characteristics, components and the relationships between them. Visit-FC adds interactive functionality to SPL engineers, allowing clear operation and manipulation of data. It provides a compact and interactive representation of feature hierarchies, allowing configuration with the spread of automatic restrictions, and providing tips for configuration issues and open decisions.

Colored Integrated Development Environment (*CIDE*) [9] is an Eclipse plug-in to support the SPL development. It is based on filters and views on source code in order to visualize and trace features in source code to support especially the variability management in SPL. This tool maintains a direct mapping between features and implementation, based on colors. That is, the portion which code belongs and a given feature are of the same color. When a piece of code belongs to more than one feature, this stretch is a combination of feature colors which it belongs.

We have seen a lack of research focus on the use of visualization resources to understand and analyze empirical data generated by SPL tools related activities. The work presented in this paper goes in this direction.

6. CONCLUSIONS

The main goal of this paper was to describe how the combined and integrated use of visual resources and analytical interaction techniques could support the comprehension of data from research methods in the SPL context. To this end, we implemented a visual platform called Vi**SPL**atform composed by a set of different visualization techniques.

In the first view, we used bubbles to represent participants of the empirical study. In this case, it was possible to interact with the bubbles chart view to obtain information and background of a specific participant. In the second view, we combined heatmap and donut chart to have an overview about the impact of participant' background on the results of tasks related to the concepts *feature model edition*, *automated analysis feature model*, *product configuration* and *import/export feature model*. We also implemented the third view to detail this influence/impact. For that, we used a combination of bar and donut chart. Finally, we used divergent stacked chart to present and explore problems faced by participants during the experiment in addition to strengths and improvement opportunities of each tool pointed by them.

We believe that this set of visualization supported by Vi**SPL**atform and the respective interaction techniques can be used in other empirical data helping to answer research question in different studies that shared similar characteristics or configuration explained and presented in this paper.

7. ACKNOWLEDGMENTS

This work was partially supported by CAPES, CNPq (grant 485907/2013-5), and FAPEMIG (grant PPM-00382-14).

8. REFERENCES

[1] D. S. Batory. Feature models, grammars and propositional formulas. In *9th Int. SPLs Conf. (SPLC)*, 2005.

[2] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *IEEE Transactions on Visualization and Comp. Graphics*, 17(12), 2011.

[3] G. Botterweck, S. Thiel, C. Cawley, D. Nestor, and A. Preußner. Visual configuration in automotive software product lines. In *32nd Annual IEEE Int. Comp. Soft. and Appl. Conf. (COMPSAC)*, 2008.

[4] S. K. Card, J. D. Mackinlay, and B. Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999.

[5] K. Constantino, J. A. Pereira, J. Padilha, P. Vasconcelos, and E. Figueiredo. An empirical study of two software product line tools. In *11th Int. Conf. on Evaluation of Novel Soft. Approaches to Soft. Eng. (ENASE)*. SCITEPRESS, 2016.

[6] K. Czarnecki and U. W. Eisenecker. *Generative programming: Methods, tools, and applications.* Addison-Wesley, 2000.

[7] S. Diehl. *Software visualization: visualizing the structure, behaviour, and evolution of software.* Springer Science & Business Media, 2007.

[8] O. Djebbi, C. Salinesi, and G. Fanmuy. Industry survey of product lines management tools: Requirements, qualities and open issues. In *15th IEEE Int. Requirements Engineering Conf.*, 2007.

[9] J. Feigenspan, C. Kästner, M. Frisch, R. Dachselt, and S. Apel. Visual support for understanding product lines. In *IEEE 18th Int. Conf. on Prog. Comp.*, 2010.

[10] R. M. Heiberger and N. B. Robbins. Design of diverging stacked bar charts for likert scales and other applications. *Journal of Statistical Soft.*, 2014.

[11] M. Mendonça, M. Branco, and D. Cowan. S.p.l.o.t.: Software product lines online tools. In *24th Conf. on OO Prog. Sys., Lang., and Appl. (OOPSLA)*, 2009.

[12] J. Pereira, C. G. Souza, E. Figueiredo, R. Abilio, G. Vale, and H. Costa. Software variability management: An exploratory study with two feature modeling tools. In *Brazilian Sym. on Soft. Comp., Arch., and Reuse (SBCARS)*, 2013.

[13] J. A. Pereira, K. Constantino, and E. Figueiredo. A systematic literature review of soft. product line management tools. In *Soft. Reuse for Dynamic Systems in the Cloud and Beyond.* Springer, 2015.

[14] K. Pohl, G. Böckle, and F. J. Van der Linden. *Software product line engineering: Foundations, principles and techniques.* Springer-Verlag, 2005.

[15] N. B. Robbins and R. M. Heiberger. Plotting likert and other rating scales. In *Proceedings of the 2011 Joint Statistical Meeting*, 2011.

[16] R. Spence. *Information Visualization: An Introduction.* Springer Int. Publishing, 2014.

[17] E. R. Tufte and P. Graves-Morris. *The visual display of quantitative information*, volume 2. Graphics press Cheshire, CT, 1983.

[18] ViSPLatform-2016. http://homepages.dcc.ufmg.br/∼kattiana/sbsi16/.

[19] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in software engineering.* Springer Science & Business Media, 2012.