Visualization Support for Multi-criteria Decision Making in Software Issue Propagation

Youngtaek Kim[1 3] * Hyeon Jeon[1] † Young-Ho Kim[2] ‡ Yuhoon Ki[3] § Hyunjoo Song[4] ¶ Jinwook Seo[1] ‖

[1] Seoul National University, Seoul, Republic of Korea
[2] University of Maryland, College Park, United States
[3] Samsung Electronics, Suwon-si, Gyunggi-do, Republic of Korea
[4] Soongsil University, Seoul, Republic of Korea

Abstract

Finding the propagation scope for various types of issues in Software Product Lines (SPLs) is a complicated Multi-Criteria Decision Making (MCDM) problem. This task often requires human-in-the-loop data analysis, which covers not only multiple product attributes but also contextual information (e.g., internal policy, customer requirements, exceptional cases, cost efficiency). We propose an interactive visualization tool to support MCDM tasks in software issue propagation based on the user's mental model. Our tool enables users to explore multiple criteria with their insight intuitively and find the appropriate propagation scope.

Index Terms: Human-centered computing—Visualization—Visualization systems and tools—Visualization toolkits; Software and its engineering—Software notations and tools—Software maintenance tools—Issue Propagation

1 Introduction

In Software Product Lines (SPLs) development, an engineer has to decide the propagation scope for many issues related to defects, requirement changes, or design modifications [5,16]. Thus, determining the appropriate scope of propagation candidates can effectively reduce the propagation cost if it is possible to avoid reviewing and applying non-essential propagation. However, deciding the scope is a rather complicated task, which has to consider contextual information (e.g., internal policy, customer requirements, exceptional or irregular cases) as well as product-related attributes. For instance, when an engineer patches a defect in a software module, one can identify affected products using the same module. However, when it comes to selecting the defect propagation candidates, certain products can be excluded under certain circumstances even though they are affected (e.g., update plan, patch policy, or cost efficiency).

Since one has to consider such multiple aspects, the selection process is known as a Multi-Criteria Decision Making (MCDM) problem. Resolving the MCDM problem often requires intervention by domain experts. While the experts are aware of relevant information, visual aid is crucial to enhance perceptual capabilities when exploring and investigating large data.

In this paper, we investigated the engineers' underlying mental model in the workflow of deciding the scope of propagation candidates through expert interviews. Based on the model, we extracted design guidelines and implemented an interactive visualization tool for anonymized Solid-State Drive (SSD) product line data. The efficacy and usability of the tool have been demonstrated through a qualitative study. We focused on MCDM problems, including mainly categorical values to accommodate the characteristics of the product data.

*e-mail: ytaek.kim@hcil.snu.ac.kr
†e-mail: hj@hcil.snu.ac.kr
‡e-mail: yghokim@umd.edu
§e-mail: yuhoon.ki@samsung.com
¶e-mail: hsong@ssu.ac.kr
‖e-mail: jseo@snu.ac.kr

2 Related Work

In several decades of research, the field of MCDM has heavily investigated for solving decision problems. Zanakis et al. conducted comparative studies of eight MCDM methods [17] and Velasquez and Hester analyzed 11 common MCDM methods by a literature review [13].

Also, several prior works proposed visualization for solving MCDM problems. Lineup [1] enabled users to interactively explore the ranking of items with stacked bar charts by weighting values of multiple criteria. Zhao et al. proposed SkyLens [18] exploring and comparing skyline points. Stephan et al. proposed WeightLifter [8] supporting weight-based MCDM for exploring weight spaces of criteria. The above studies focused on numeric attribute analysis. On the other hand, Upset [2] employed a matrix-based layout to show intersections and aggregation of sets composed of categorical data. The techniques support tasks for understanding set data with multiple criteria.

In this paper, the design of our visualization is based on the set visualization techniques proposed in Upset for analyzing categorical data. Also we exploited the Simple Additive Weighting (SAW) as a baseline method for calculating the confidence level with weights of each criterion due to its intuitiveness and simplicity.

Lopez et al. surveyed that various visualization techniques had been studied to support the SPL engineering activities such as domain requirements engineering, design, and modeling [3]. Nestor et al. employed visualization techniques to address SPL tasks such as variability management and product derivation [7]. Murashkin et al. proposed a bubble-chart-based visualization tool to explore a multi-dimensional space of optimal variants [6]. Urli et al. [11] presented a visualization to support large feature models, and Martinez et al. [4] built feature relations graphs on understanding feature constraints.

Regarding the propagation, Montalvillo et al. visualized how propagations were conceived and realized in detail by enhancing the branch graphs in the version control system [5]. As SPL evolves, the scale of variability can be notably complex to manage a huge number of propagation between products [7]. Our research focused

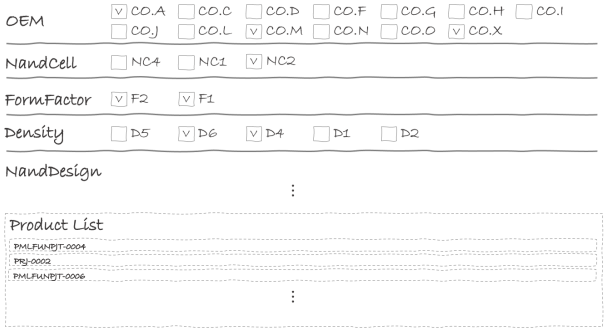

Figure 1: The sketch of existing issue propagation system.

on alleviating the effort to determine the issue propagation scope in SPL through visualization support.

3 Design Considerations

We performed a preliminary interview with domain experts at Samsung Electronics; one software engineer involved in product management and two engineers (including one of the authors) who supported the propagation tasks in the department of software engineering. One in charge of determining the propagation scope relies on the information of the issue to be propagated, products, and development context to perform the determining task. After the determination, the issue is propagated to corresponding products within the scope. The managers of the products review the issue and confirm whether it will be assigned to the product finally or not.

The scope may vary depending on the urgency or severity of the issue. The broader the scope, the more effort product managers put into reviewing the issue. The smaller the scope, the more likely there exists products missing the issue. Hence, one sets the obvious products first based on certain conditions and then selects an appropriate number of candidates based on one's own contextual information with some uncertainty. Sometimes, one may have insufficient information on the conditions but know a partial list of products to be propagated. In that case, one can expand the scope by reviewing the products' attributes and inferring the missing conditions.

Currently, the tool to identify a list of products filtered by attributes' values with employing the multi-selection interaction was deployed internally as sketched in Figure 1. However, users mainly depend on a spreadsheet form containing related information instead of utilizing the tool due to inconvenience and inefficiency issues.

Through the interview, we reviewed the available information the interviewees own and their mental model. The users' information can be categorized as either *Certain* or *Uncertain*. The *Certain* (Figure 2a) consists of confirmed conditions of selections determining the products are candidates or noncadidates such as "Propagate this issue to any products using controller B or with capacity greater than 500GB", "Do not propagate this issue to SLC (Single Level Cell) products." However, a group of conditions can conflict with each other, and such conditions are categorized as *Uncertain*.

The *Uncertain* (Figure 2b) consists of unsettled conditions based on domain knowledge or experience. The combination of such conditions determines the confidence level that a product can be a candidate or not. Products get selected as candidates when the confidence level is above a certain threshold set by the user. As a result, products to be issue-propagated can be divided into candidates, non-candidates, and the rest having the confidence level. In our system, we put the rest on the grey area (Figure 2c).

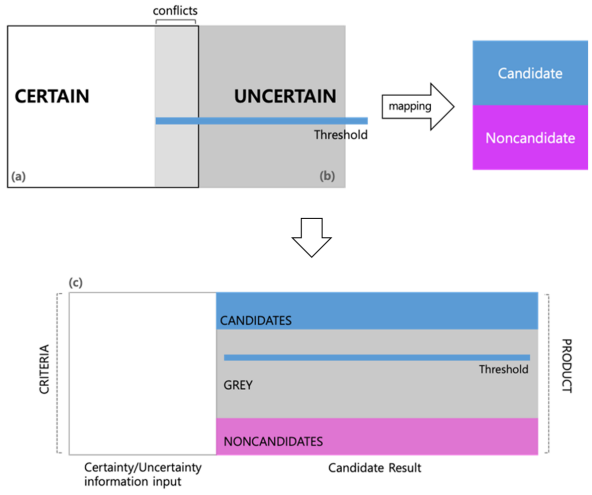

Figure 2: User mental model (a, b) and visualization layout (c)

We then established design guidelines based on the interview and the mental model.

- **R1**: Support multiple selection of *Certain* and *Uncertain* conditions interactively.
- **R2**: Present the suggested propagation scope separately as *candidate*, *noncandidate*, *grey* area.
- **R3**: Present confidence level of propagation for products in *grey* area with how the level was determined.
- **R4**: Provide condition adjustment guidance for inferred missing condition.
- **R5**: Provide visual feedback for the change of the propagation scope by setting the conditions.

4 Interactive Visualization

Following the design guideline, we developed an MCDM visualization tool as a single page application implemented in JavaScript with React. We designed an overall layout of the visualization, as shown in Figure 2c. In the criteria pane (left), one can interactively adjust the *Certain*/*Uncertain* conditions and the resulting propagation scope appears on the right. In the product pane (right), propagation candidates (blue background color) and noncandidates (magenta background color) are presented. *Uncertain* products are shown in the *grey* area along with the confidence level. One can adjust the threshold to manipulate the scope of candidates. Since the anonymized data obtained from the SSD product line mainly consist of categorical variables, we focused on managing such data types.

The tool consists of two panes: criteria and product. One can set the conditions on the criteria pane and identify the scope on the product pane. The tool also guides the inferred conditions to help users adjust missing conditions.

4.1 Criteria Pane

In Criteria Pane, there are multiple categorical values in each criterion. The user can change the state of each criterion value individually by clicking the value glyph (Figure 3) and the state cycles between three states: *Candidate*, *Noncandidate*, and *Weight* (R1). Former two *Certain* states force corresponding products to place in the *candidate* or *noncandidate* area (R2). As for the *Weight* state, it shows the vertical range slider, which could adjust *Uncertain* conditions (Figure 3): Moving up the slider adds the weight to the confidence level of propagation, and moving down reduces its weight. The weight direction is aligned with the position of each area on product pane. If the state changes from *Weight* to the others, the slider disappears, and the weight value is discarded.

On the other hand, the horizontal range slider (Figure 4b) under a criterion name is used for adjusting the importance of the criterion. Since the importance values are normalized, their ratios are relative to each other. Each criterion is encoded with a distinct color, which is identical to *Weight* vertical slider widgets, *candidate* column, criteria legends, and stacked bar charts on the product pane.

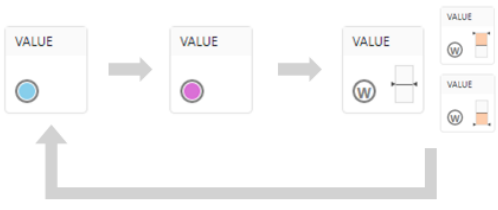

Figure 3: Toggling the value glyph changes the state: *Candidate*, *Noncandidate*, *Weight* (from left). If the *Weight* state is set, a user can control the weight of the value by manipulating the vertical slider.

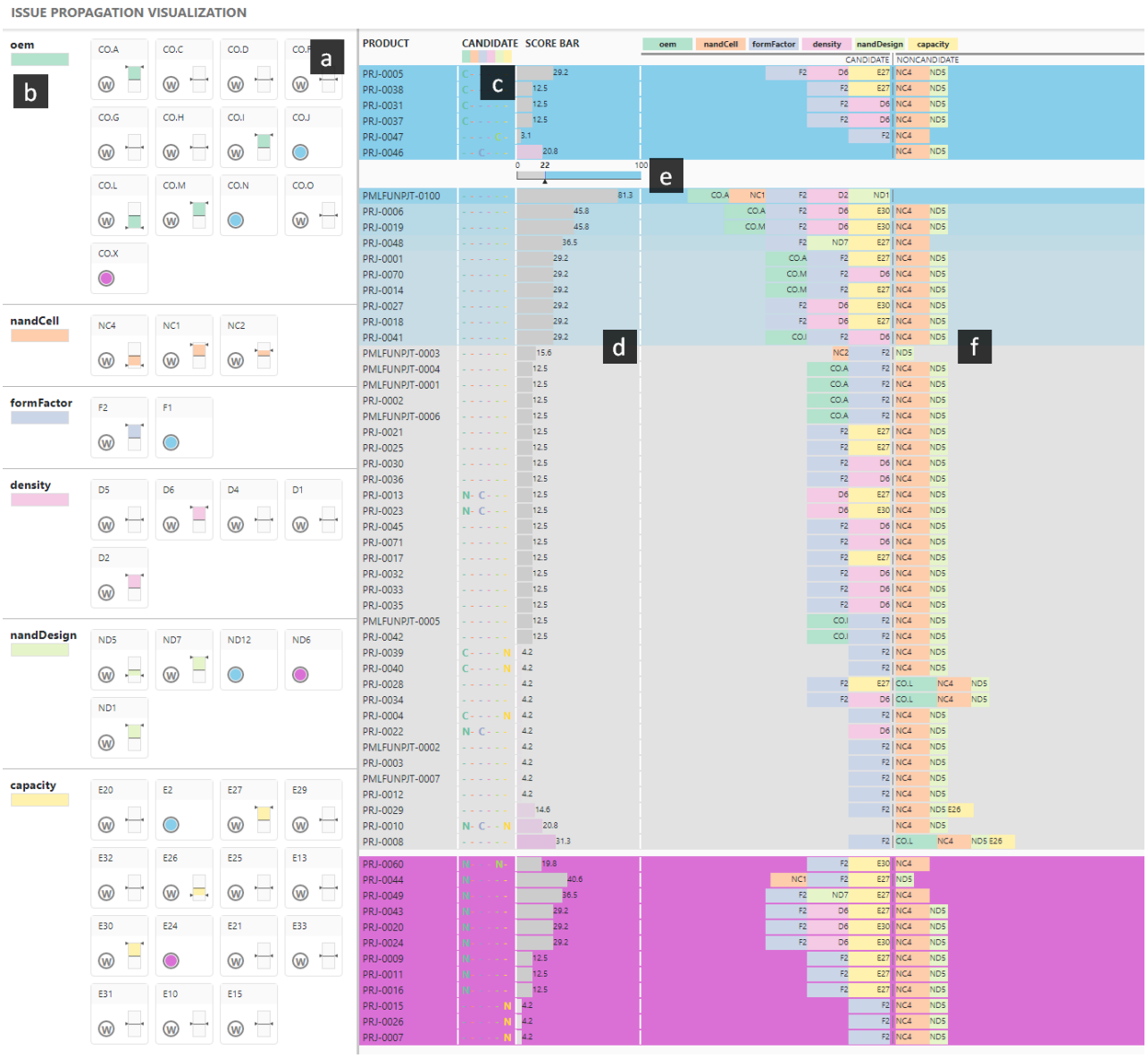

Figure 4: Proposed interactive visualization tool. On the left criteria pane, the user sets (a) the states or the weights of criterion values and (b) the importance of criteria. On the right product pane, products are dynamically listed (c) according to corresponding states. They are ordered according to user interactions: toggling value glyphs, dragging colored sliders, (e) adjusting threshold slider. The information of confidence level is (d) expressed as bar charts, and its details are (f) represented as stacked bar charts.

4.2 Product Pane

Product pane has *candidate*, *noncandidate* and *grey* area with distinctly encoded colors. When a user manipulates widgets in criteria pane, products are dynamically placed in their area where they belong according to the input from the criteria pane (R1, R5).

The *candidate* column (Figure 4c) shows whether criteria of products are set to *Candidate* or *Noncandidate* (R2). The more one assigns criteria set to *Candidate*s, the greater the chance product is placed on the *candidate* area, and vice versa. If a product has both *Candidate* and *Noncandidate* criteria, it means there is a conflict, and it will be placed on *grey* area.

We exploit the SAW model in calculating the confidence level since its intuitiveness for users to understand [14, 17]. The sum of the weights is expressed as a score and is encoded as a length of a bar (Figure 4d), along with the actual value on the right. The score detail is expressed as a stacked bar on the right (R3; Figure 4f). The color of each bar matches the color of the corresponding criterion. The value of the criterion is also presented in the bar to identify which value affects the confidence level.

In addition, the user can set the threshold value of the score (Figure 4e) by adjusting the horizontal range slider. On the grey area, if the product's score exceeds the value, the background color tone changes to blue with gradation by the score: the product row with the higher score are painted darker (R5).

4.3 Weight Inference for Guiding Missing Conditions

One can identify the attributes values of products on the product pane, as shown in Figure 5 by toggling show values button. As aforementioned in Section 3, it helps users infer the missing condi-

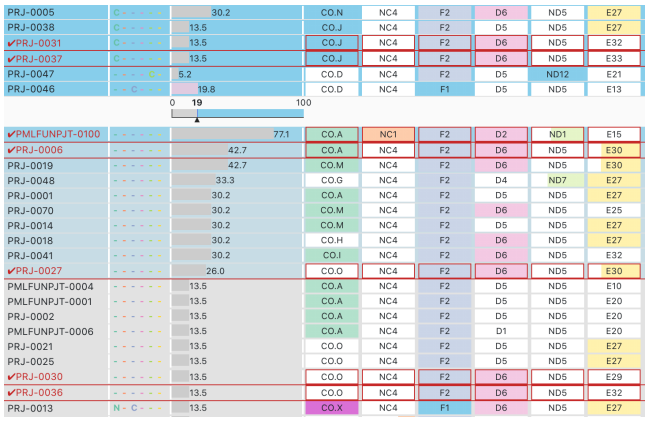

Figure 5: One can identify the attribute values of products on the product pane if necessary by clicking show values button. One can also select product candidates to infer the weights of criteria. Selected products and their attributes' values are highlighted in red.

tions by reviewing the products' attributes. For instance, if all the selected products have a common attribute value, increasing the value's weight will likely be one of the missing conditions reversely. However, the manual investigation into all the selected products' values is hard to capture the missing conditions when the number of selections or attributes is large.

We provide the missing condition guidance by inferring the weights of criteria and their values from user-selected products. After selecting products expected as candidates, the tool guides users which criteria value is necessary to be increased and the amount of increment, as shown in Figure 6.

To effectively extract missing conditions, the tool provides minimum value weight guidance that reflects selected candidates' commonalities. Hence, the weight inference method first computes each criterion's significance and distributes the significance into each criterion value weight. The significance of a criterion is computed based on its uncertainty, which is defined as Shannon Entropy [10] due to the selected products' ratio with its different criteria values. The detailed formula is as follows: for criteria values $V = \{v_1, v_2, \cdots, v_n\}$ of a certain criterion C, the uncertainty μ_C of C is calculated as

$$\mu_C = -\sum_{v_i \in V} Prob(v_i) \log Prob(v_i),$$

where $Prob(v_i)$ is defined as the ratio of the selected products which have v_i as criteria value. If the selected products all have the same criteria value for C, the products certainly have such criteria value as common, and the uncertainty of C goes lower. On the other hand, if they all have different criteria values, we can say that such criteria cannot be a common feature of the products. Therefore uncertainty pf C will increase. Shannon entropy directly reflects such intuition, as it measures the uncertainty of a probabilistic event [9] as the number of bits required to explain the event [12]. In this case, the ratio of selected products of each criteria value becomes the probabilistic distribution.

The tool then computes each criterion's significance by dividing the minimum uncertainty among criteria with each criterion uncertainty. Therefore, the criterion's significance with minimum uncertainty becomes 1, and the other criterion's significance decreases as their uncertainty grows. Finally, each criteria value weight is inferred by multiplying the significance by the ratio of the criteria value weight among selected products. The detailed algorithm of the entire weight inference procedure is described in Algorithm 1.

The criteria pane provides a guide for the weight by using the criteria value wights values that were finally presented (Figure 6)

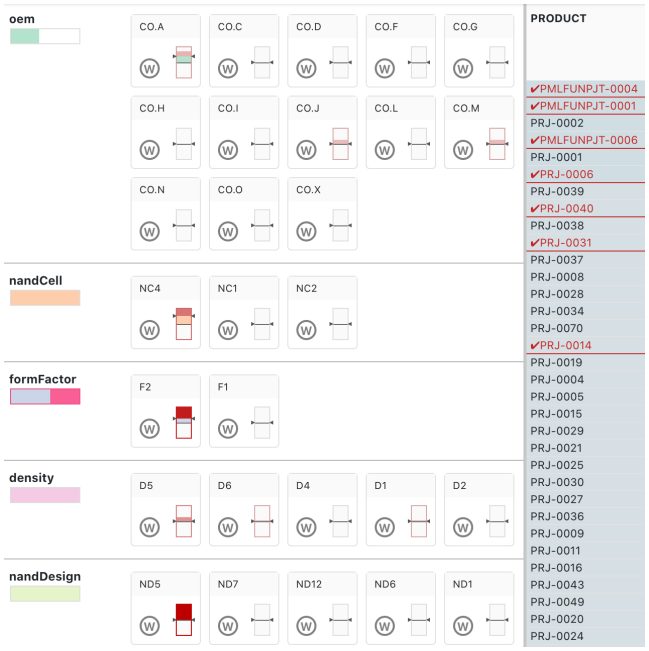

Figure 6: The weights of criteria and criteria values of the selected products are inferred. The red-bordered sliders are guided to increase their weights. The amount to increase is encoded as red bars in the sliders, and the amount is linear to the difference between the current weight and the inferred one.

one can adjust the weights following the guidance highlighted in red. If the criteria weight itself is low and insufficient to increase the criteria value weight, guidance to increase the criteria weight is also provided. The saturation of the highlighted color encodes the amount of the difference between the current weight and the guided weight.

5 Qualitative Study and Discussion

We performed an online case study for 60 minutes with one of the domain experts (I1), one of the authors who participated in the preliminary interview. The interviewee had never experienced the tool before the study. We demonstrated the tool and discussed its usability, effectiveness, and further works to be deployed in the real-world.

5.1 Usability and Usefulness

I1 primarily appreciated the entire layout based on the user mental model. She was satisfied with the weighting system for uncertain areas and also was satisfied that the product pane shows candidates and

Algorithm 1 Computing Pairwise Distortion & Weight

Input Criteria set $\mathscr{C} = \{C_1, C_2, \cdots C_n\}$ where C_i consists of criteria value $V_i = \{v_{i,1}, v_{i,2}, \cdots, v_{i,m}\}$
Input Selected Candidates Products $P = \{p_1, p_2, \cdots, p_l\}$
Output weights for each criteria value
for C_i in $\mathscr{C}$ **do**
    $\mu_{C_i} = -\sum_{v_i \in V} Prob(v_i) \log Prob(v_i)$
    $\mu_{\min}$ = minimum value among every μ_{C_i}
for C_i in $\mathscr{C}$ **do**
    $\sigma_{C_i} = \mu C_i / \mu_{\min}$ ▷ Significance of C_i
    for v_j in V_i **do**
        $w_{v_i} = \sigma_{C_i} \cdot Prob(v_i)$ ▷ weight of v_i
return weight W_v for every criteria value v

noncandidates simultaneously. If noncandidates are not displayed, it can decrease the chance of reviews for noncandidate products, which occasionally occurs in the existing systems with a filtering-out based approach. She could understand the way each component worked during the demonstration session. However, she pointed out that the value glyph button having the symbol *W* was not intuitive for state changes. And she also wanted there existed a definite border between the certain and candidate area because the similar color hindered the separation between them.

During the demonstration session, I1 found which features to use as attributes in this tool would be critical to its performance and found that some contextual information can be a good attribute. Currently, base elements were selected as attributes, which consist of hardware/software-related features.

I1 addressed this tool would work efficiently for determining the propagation of an instantly occurred issue. However, she noted that supporting additional issue-related functions is essential to cover various propagation practices in the industrial field. We discussed possible requirements for the purpose: Detailed information about each issue can be provided with a connection to the existing issue tracking systems; Also, supporting the propagation of multiple issues simultaneously is necessary, which occasionally occurs in real practices; Visualizing clusters of issues based on their similarity would help users alleviate the effort of processing many issues.

5.2 Limitation and Future works

The data used in this study was strongly anonymized without rich information about products or issues to be propagated due to internal security issues. Also, the data size was not large enough to reflect the real-world scale. The number of products is increasing as SPL evolving, and also the variety of the product attributes will increase as well; the current design need more scalability to cope with them. Hence clustering techniques for both products and issues with appropriate visualizations can alleviate exploring a large number of items. Additional text-based data refined with natural language processing techniques can also give more clues to perform the task. We will solve this issue with active collaboration and offline field evaluations.

Moreover, setting all criterion weights is laborious and tedious as the number of the criterion or the criterion values increases. Analyzing user provenance [15] would help users understand historical patterns to determine the propagation scope. We can reduce the burden of setting the weights by utilizing frequent or similar condition sets.

To be deployed in the real environment, I1 suggested practical features such as personalized customization for registering attributes, sharing the reasoning process of how the scope is determined in a summarized form, and reporting the results. We leave these issues for future works.

6 Conclusions

We proposed a visualization tool supporting MCDM for finding the propagation scope in SPL based on the users' underlying mental models. We derived the design guidelines based on the users' mental model through the preliminary interview with domain experts. We then implemented the interactive visualization tool supporting issue propagation tasks by considering the design requirements. Through the case study with a domain expert, we confirmed that our tool enables users to explore and investigate the appropriate propagation scope interactively. The implementation of our system is available at https://issue-propagation.github.io/demo/.

Acknowledgments

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. NRF-2019R1A2C2089062), by Korea Electric Power Corporation (Grant number:R18XA01), and in part by Samsung Electronics. Jinwook Seo is the corresponding author.

References

[1] S. Gratzl, A. Lex, N. Gehlenborg, H. Pfister, and M. Streit. Lineup: Visual analysis of multi-attribute rankings. *IEEE transactions on visualization and computer graphics*, 19(12):2277–2286, 2013.

[2] A. Lex, N. Gehlenborg, H. Strobelt, R. Vuillemot, and H. Pfister. Upset: visualization of intersecting sets. *IEEE transactions on visualization and computer graphics*, 20(12):1983–1992, 2014.

[3] R. E. Lopez-Herrejon, S. Illescas, and A. Egyed. A systematic mapping study of information visualization for software product line engineering. *Journal of software: evolution and process*, 30(2):e1912, 2018.

[4] J. Martinez, T. Ziadi, R. Mazo, T. F. Bissyandé, J. Klein, and Y. Le Traon. Feature relations graphs: A visualisation paradigm for feature constraints in software product lines. In *2014 Second IEEE Working Conference on Software Visualization*, pp. 50–59. IEEE, 2014.

[5] L. Montalvillo and O. Díaz. Tuning github for spl development: branching models & repository operations for product engineers. In *Proceedings of the 19th International Conference on Software Product Line*, pp. 111–120, 2015.

[6] A. Murashkin, M. Antkiewicz, D. Rayside, and K. Czarnecki. Visualization and exploration of optimal variants in product line engineering. In *Proceedings of the 17th International Software Product Line Conference*, pp. 111–115, 2013.

[7] D. Nestor, S. Thiel, G. Botterweck, C. Cawley, and P. Healy. Applying visualisation techniques in software product lines. In *Proceedings of the 4th ACM symposium on Software visualization*, pp. 175–184, 2008.

[8] S. Pajer, M. Streit, T. Torsney-Weir, F. Spechtenhauser, T. Möller, and H. Piringer. Weightlifter: Visual weight space exploration for multi-criteria decision making. *IEEE transactions on visualization and computer graphics*, 23(1):611–620, 2016.

[9] H. Schütze, C. D. Manning, and P. Raghavan. *Introduction to information retrieval*, vol. 39. Cambridge University Press Cambridge, 2008.

[10] C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE mobile computing and communications review*, 5(1):3–55, 2001.

[11] S. Urli, A. Bergel, M. Blay-Fornarino, P. Collet, and S. Mosser. A visual support for decomposing complex feature models. In *2015 IEEE 3rd Working Conference on Software Visualization (VISSOFT)*, pp. 76–85. IEEE, 2015.

[12] S. Vajapeyam. Understanding shannon's entropy metric for information. *arXiv preprint arXiv:1405.2061*, 2014.

[13] M. Velasquez and P. T. Hester. An analysis of multi-criteria decision making methods. *International journal of operations research*, 10(2):56–66, 2013.

[14] E. Wall, S. Das, R. Chawla, B. Kalidindi, E. T. Brown, and A. Endert. Podium: Ranking data using mixed-initiative visual analytics. *IEEE transactions on visualization and computer graphics*, 24(1):288–297, 2017.

[15] K. Xu, A. Ottley, C. Walchshofer, M. Streit, R. Chang, and J. Wenskovitch. Survey on the analysis of user interactions and visualization provenance. In *Computer Graphics Forum*, vol. 39, pp. 757–783. Wiley Online Library, 2020.

[16] B. Yu. Quantifying software architecture attributes. 2006.

[17] S. H. Zanakis, A. Solomon, N. Wishart, and S. Dublish. Multi-attribute decision making: A simulation comparison of select methods. *European journal of operational research*, 107(3):507–529, 1998.

[18] X. Zhao, Y. Wu, W. Cui, X. Du, Y. Chen, Y. Wang, D. L. Lee, and H. Qu. Skylens: Visual analysis of skyline on multi-dimensional data. *IEEE transactions on visualization and computer graphics*, 24(1):246–255, 2017.