ITC 3/50
Information Technology and Control
Vol. 50 / No. 3 / 2021
pp. 424-442
DOI 10.5755/j01.itc.50.3.27622

A Data Mining Technique to Improve Configuration Prioritization Framework for Component-Based Systems: An Empirical Study

Received 2020/09/07 | Accepted after revision 2021/08/04

http://dx.doi.org/10.5755/j01.itc.50.3.27622

HOW TO CITE: Ali, A., Hafeez, Y., Ali, S., Hussain, S., Yang, S., Malik, A. J., Abbasi, A. A. (2021). A Data Mining Technique to Improve Configuration Prioritization Framework for Component-Based Systems: An Empirical Study. *Information Technology and Control*, 50(3), 424-442. https://doi.org/10.5755/j01.itc.50.3.27622

A Data Mining Technique to Improve Configuration Prioritization Framework for Component-Based Systems: An Empirical Study

Atif Ali, Yaser Hafeez, Sadia Ali
University Institute of Information Technology, Pir Mehr Ali Shah Arid Agriculture University; Rawalpindi 46000, Pakistan; e-mails: atif.alii@yahoo.com; yasir@uaar.edu.pk; sadiaalief@gmail.com

Shariq Hussain
Department of Software Engineering, Foundation University Islamabad, Islamabad 44000, Pakistan; e-mail: shariq@fui.edu.pk

Shunkun Yang
School of Reliability and Systems Engineering, Beihang University, Beijing 100191, China; e-mail: ysk@buaa.edu.cn

Arif Jamal Malik
Department of Software Engineering, Foundation University Islamabad, Islamabad 44000, Pakistan; e-mail: arif.malik@fui.edu.pk

Aaqif Afzaal Abbasi
Department of Software Engineering, Foundation University Islamabad, Islamabad 44000, Pakistan; e-mail: aaqif.afzaal@fui.edu.pk

Corresponding author: ysk@buaa.edu.cn

In the current application development strategies, families of products are developed with personalized configurations to increase stakeholders' satisfaction. Product lines have the ability to address several requirements due to their reusability and configuration properties. The structuring and prioritizing of configuration requirements facilitate the development processes, whereas it increases the conflicts and inadequacies. This results in increasing human effort, reducing user satisfaction, and failing to accommodate a continuous evolution in configuration requirements. To address these challenges, we propose a framework for managing the prioritization process considering heterogeneous stakeholders priority semantically. Features are analyzed, and mined configuration priority using the data mining method based on frequently accessed and changed configurations. Firstly, priority is identified based on heterogeneous stakeholder's perspectives using three factors functional, experiential, and expressive values. Secondly, the mined configuration is based on frequently accessed or changed configuration frequency to identify the new priority for reducing failures or errors among configuration interaction. We evaluated the performance of the proposed framework with the help of an experimental study and by comparing it with analytical hierarchical prioritization (AHP) and Clustering. The results indicate a significant increase (more than 90 percent) in the precision and the recall value of the proposed framework, for all selected cases.

KEYWORDS: Configurable systems, Requirement prioritization, Semantic analysis, Software product line, Component-based systems.

1. Introduction

The software functional requirements play an important role in the development of component-based systems for agility and timely delivery of products, with high quality. Inefficient prioritization of requirements and the reuse of requirements in component-based systems degrades development and promotes system failures [27]. These systems are built on reuse concepts to reduce complexity, and developers adopt components of the product from different sources. The sources may be open source, build new, get from a distributed source, or used pre-built components by other organizations [12, 18, 39]. Therefore, most component-based systems are a combination of previously built components that are relevant to the same domain of new products' development. Currently, for the implementation of reused components, and features to manage requirements of new products, different configurations of components are enabled by the developing team [36, 44]. As software products in the advanced intelligent technology era are a combination of software and hardware with a large number of customization options (for example, a searching browser provides different options like setting, home, sign in, history, language, pop up window, etc. for the users and every user has own options selection scheme) to use products in several ways [27, 36]. Hence, the configuration is the customization options in software and hardware products for tailoring functional and non-functional requirements of components-based systems.

The component-based systems are highly configurable systems due to reuse, structuring, and flexibility [8, 12]. In these configurable systems, the focus is on historical information of some requirements or reuse of implementation information of requirements in the previous version or other similar domain requirements [13, 27]. These systems consist of several variant features that may exist in the same domain-specific configurable system, with the integration of other feature combinations [35, 37]. This organization of improving requirements features management, and reuse for a family of software products introduced an interrelated software system. This helps to fulfil the demand of large relevant customers by managing and sharing common features in similar products [11, 13, 47]. The main goal of software product lines (SPLs) is to enhance the quality and production by adopting the reuse of efforts and assets, thereby reducing cost and time [3, 10, 41]. It is an efficient engineering process to minimize the development time for product families by providing a common model. The central concept of SPLs is to provide a set of common and diverse components for the system which are identified to ensure a consistent series of products [3, 6, 47]. In SPLs, to deal with upcoming product changes, prediction of requirements is used to suggest to reuse existing re-

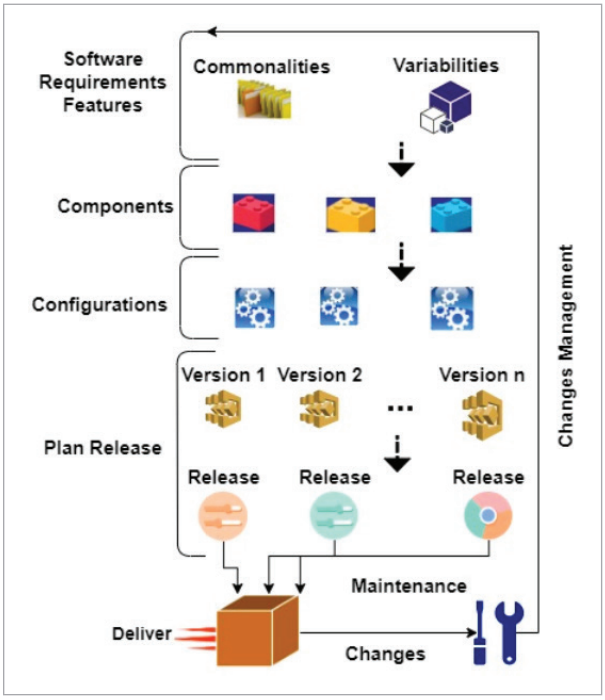
quirements for changes at the early product development stage [31, 44]. These products have several features and components, which increases significance for systematic reuse with complexity in configuration management [5, 20, 24]. To simplify the configuration management of features, the extraction of a feature model is adopted for recognizing common and variant features of highly configurable systems.

The application of a configurable system in application engineering adopts the process of reusing domain-specific assets and structuring the configurations of the components for version updating or new version release management, as shown in Figure 1.

As shown in Figure 1, a set of requirements are defined after gathering requirements from stakeholders. Thus, requirements sets are the combinations of different components when the system updated or new series of the system are released after the implementation of the changes. Firstly, commonalities and variabilities are identified from the set of requirements according to new system requirements. Secondly, components from the repository are selected against the requirements of the product based on reuse functionality. Thirdly, configuration of these commonalities and variabilities of components are generated according to functional and non-functional requirements of the system to facilitate users. After configuration selection and interaction, releases a new product of a similar domain or updates the previous version and process continues with the evolution in the system.

The interaction of configuration and their features with other configurations' features creates ambiguity and errors due to improper prioritization and selection of features or configuration to fulfil system requirements [6, 29, 36, 47]. For a configurable system, to correctly identify and manage the requirements, and to avoid reuse of irrelevant requirements of other applications, it is necessary to improve the structuring, and prioritization process of configurable systems requirements, with the active participation of stakeholders [15, 30, 34]. Hence, these configurable systems have several variability possibilities, and these are all interdependent. If an incorrect ranking is assigned, the interfaces collapse after integration, which increases the cost and promotes system failures. The main issues in the prioritization of configurable applications are the incompleteness, inconsistency, ambiguous configurations, inefficient

Figure 1
Configurable system organization



adaptation to changes, and tracing components in all phases of software development due to an inadequate configuration prioritization (CP) process. Therefore, prioritization of configuration in components-based systems is an important activity for correct and accurate reuse, and structuring of configuration in a continuous software evolution environment [2, 15, 36].

Therefore, CP is an important and critical activity for requirements configuration identification, prioritization, and validation before software release [1, 17, 32]. An outstanding CP is necessary for any well-run project. It ensures that the project concentrates on the important configurations and their correct interactions as firstly perceived by both stakeholders and the developing team. There are many techniques such as analytical hierarchical prioritization (AHP), 100-dollar, value-based, and fuzzy logic that are helpful for the prioritization of requirements according to the stakeholder's needs, time, and cost, as well as the nature of the project [15, 19].

However, these techniques fail in large scale projects, specifically in highly configurable systems where multi-stakeholders are involved and larger sets of

variant features are present in the product series [23, 38, 46]. Therefore, to address this problem, researchers and industries have implemented and proposed different methods for prioritizations, with the integration of data mining, artificial intelligence, and machine learning techniques. However, these still lack prioritization to handle larger sets of configurations due to reuse interfaces of different components for core assets and variant management in the dynamic and frequently changing environments, with lesser stakeholders' participation and additional efforts.

Accordingly, to resolve these issues, we adopted a data mining technique based on historical information for the active involvement of stakeholders, using their continuous feedback and reducing effort for the prioritization process. To represent significant attributes of information employed data mining method that helps in successful identification of important attributes. The determination of important hidden attributes is important to increase the quality of information [26]. Thus, to determine information about important configurations of features in component-based systems, different approaches are adopted according to the nature of the information. Consequently, in the present era large and complex information is being generated which increases the need for data mining during the software development process. Application of data mining helps in improving software quality and productivity but still, different challenges exist like inconsistency, ambiguities, incompleteness, etc., during information attribution extraction [26, 36].

Therefore, different types of algorithms are required to extract accurately different aspects of information like text, sequence, graph, etc., mining. Information in software development consists of historical data, change in information, execution, trace links, user access of components, and bug reports [14]. This information helps to track product progress, history, and evolution by analyzing previously hidden and important data statistics [36, 37]. Researchers and practitioners try to explore potential information according to product nature using data mining methods, for high quality and within limited resources and time frame [28, 29]. Data mining tools are beneficial in the identification of hidden important patterns of information effectively. Therefore, we adopted in our research the Weka tool for mining software requirements information to improve the prioritization process of configurable systems [4, 25, 29, 49]. Weka tool has built-in most of the data mining algorithms and is widely adopted by researchers [37].

To control ambiguities and redundancy in mined configurations' information of updated or new product, we adopted the mining algorithm based on association rules for the prediction of frequently access or changed configurations to deal with dynamic changes in highly configurable systems to ensure higher customer satisfaction [23, 36, 40, 46]. The frequently accessed configurations are the functionalities of the system which are mostly used by end-users in previous similar product versions or series. Similarly, frequently changed configurations are the functionalities that mostly changed in previous similar product versions or series. Thus, it increases the reusability of priority sequences and reduces stakeholders' conflicts during the prioritization of the configuration of systems. Therefore, we prioritized system functionalities based on stakeholder's priority and historical information of similar products to reduce ambiguities, incompleteness, and inaccuracy. As observed, stakeholders employ prioritization following their viewpoints and the market value of product functionalities [16, 25]. The stakeholder's priority is based on three factors identified from existing literature i.e. functional (system functionality and customization), experiential (personal and social relations), and expressive (personal and social meanings) to avoid homogeneity among stakeholders. Therefore, the proposed framework for managing the process of configuration prioritization in configurable systems employs the merger of stakeholders' priority and mining historical data. The contributions of this study are as follows:

1. In this study, we present a comprehensive framework to mitigate configuration prioritization challenges in component-based systems.
2. The proposed framework provides a platform for correct and accurate configuration prioritization to improve component-based system development using stakeholder's priority and historical data. Firstly, for stakeholder's priority the functional, experiential, and expressive factors used for heterogeneous viewpoints analysis and priority of each system configuration. Secondly, historical data mining is based on association rules using two information either frequently accessed or changed configurations to extract configuration priority. These two processes, in the proposed framework,

are adopted to avoid missing and ambiguous information. Then merged both priorities of configuration and sorted according to the highest priority for accurate interaction of system interaction after changes or evolution.

3 The performance of the proposed framework is evaluated by using an experiment involving three different datasets to compare its results with the AHP and Clustering techniques.

4 Our results indicate the superiority of the proposed framework over other techniques and proved that all selected highest priorities are true and relevant.

The rest of this paper is structured as follows: Section 2 introduces related work to describe the existing literature. Section 3 presents the proposed framework while the performance evaluation and details of the evaluation procedure are described in Section 4. Further, Section 5 reports the experimental results and their discussions followed by the conclusion and future work in Section 6.

2. Related Work

The increasing difficulty, mismanagement, and cost of the configurable systems had led researchers to increase the reuse of similar requirements for product development. SPLs resolves the reuse problem in component-based systems; however, still it is difficult to manage, prioritize, and select requirements with ambiguity, conflicts, and incompleteness especially prioritization of configurations due to frequently reused components and their incorrect interaction with each other. In the existing literature, different studies have focused on requirement management and reuse by proposing techniques for configurable systems. Arias et al. [3] managed requirements reuse by proposing a framework based on semantic analysis. Similarly, Kaindl et al. [24] proposed a method for the reuse of similar features and for matching features, based on previous information by using case-based reasoning to reduce the demand-supply gaps. Ra'Fat et al. [45] identified variant feature locations by using formal analysis of concepts as the configuration requires dynamic monitoring.

The structuring and maintenance of change requirements with reuse concepts bring challenges that can be reduced by using different viewpoints and responsibility analysis. Hence, requirements are the backbone of every project, and accurate prioritization and implementation of these requirements are necessary to obtain a correct and high-quality system. The studies [16, 25, 46] provide insight into the release planning processes used in the software industry to create software product value. It presents the degree to which the significant stakeholders' viewpoints are spoken to in the basic decision-making process. The SPLs strengthened high-level constructive software reuse by exploiting commonality and managing variability in a product family.

Hence, different studies have highlighted the importance of prioritization of features in requirement management and have also implemented reuse to reduce ambiguity, incompleteness, and delivery time [1, 17, 20]. These studies [9, 21, 22, 43] described that prioritization is important for multiuser perspectives in component-based systems. Researchers adopted different optimization algorithms to identify the correct behavior of the system after the integration of different components having variant features with a reduction in efforts and conflict of stakeholders. Similarly, to obtain the full potential of cyber-physical systems (CPSs), both technological and service aspects need to be considered, which results in complex systems due to the dynamic changes in stakeholder requirements. These products mainly focus on the requirement engineering process due to prioritization and reuse requirements. Wiesner et al. [48] proposed an agile method-based framework to manage more cost-effectively the challenges of requirements engineering in CPSs. Prioritizing requirements by focusing on stakeholders' feedback results in a noteworthy increase in development cost because of the time elapsed in several human interactions. In another work, a semi-automated framework has been presented to predict the ratings to reduce human interactions [7]. A prioritization method known as case-based ranking integrates the project's stakeholders' needs with requirements ordering approximations, calculated through automated techniques [42].

Therefore, the prioritization process is of great importance in the configuration management process as well as in the test case and requirements optimization process [4, 33]. These studies [4, 33] identified that there is need of better method for the prioritization of test cases and requirements of larger sets of variants respectively, to determine the reliability of configuration requirements after the implementation while integrating and validating different commonalities and variants for software release.

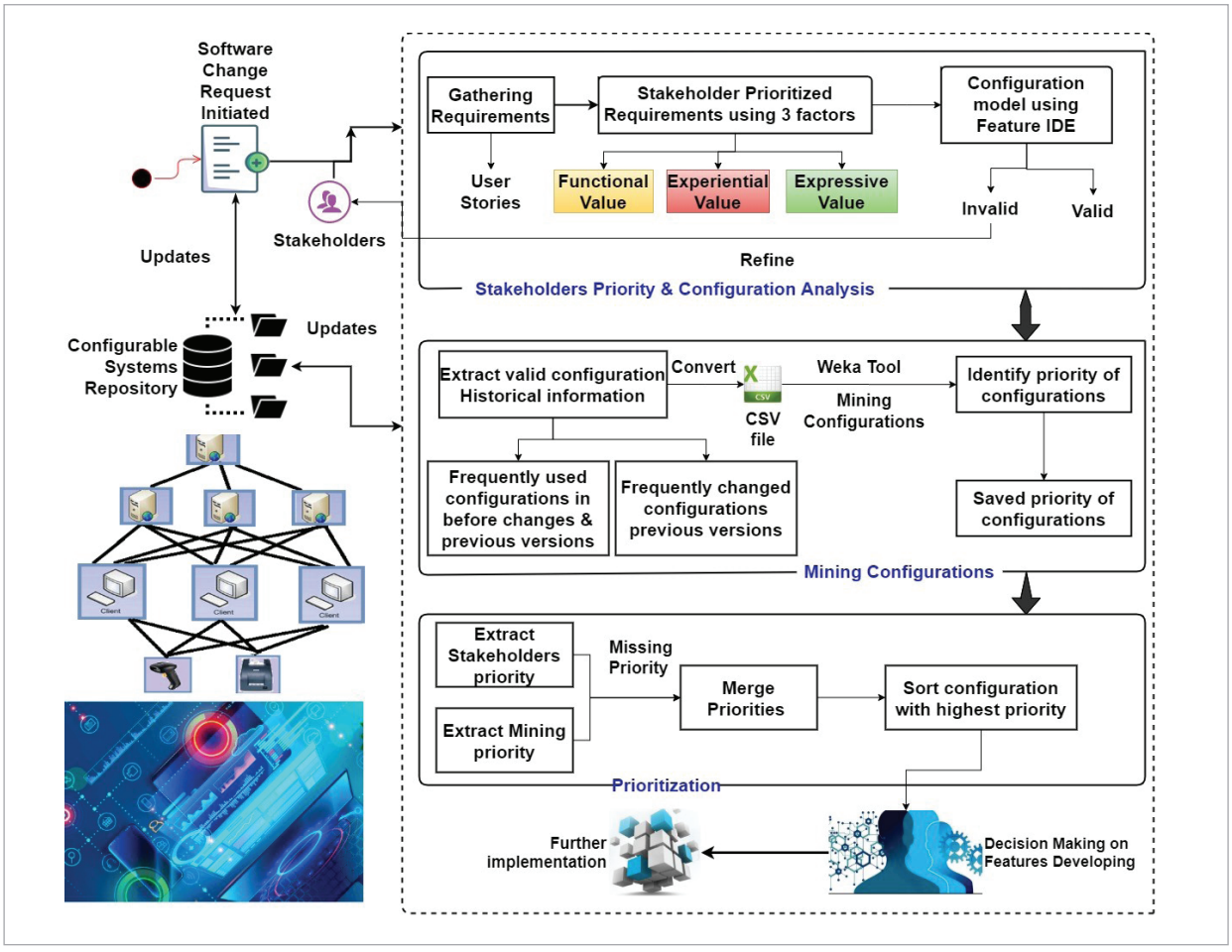
Thus, from the literature, we conclude that accurate extraction of a diverse perspective of stakeholder's priority is necessary to identify the sequence of configurations for integration to fulfil stakeholders' requirements accordingly. Consequently, it is difficult to implement the bulk of requirements configurations in the first iteration or version accurately with higher satisfaction and quality, owing to the limited resources involved in continuous integration and changing environment. Therefore, an improved prioritization process is required that can identify accurate sequence and value of high priority requirements' configuration for the first iteration or version to accurately change and reuse components' implementations, and reduce efforts, conflicts, and ambiguity in accordance with diverse stakeholders' viewpoints and developing team.

3. Proposed Framework

The proposed Framework for configuration Requirements Prioritization (FRP) is presented for the configuration prioritization of highly configurable component-based systems to reduce complexity, human effort, incompleteness, ambiguity, and increase user satisfaction, considering the limited resources. For prioritization, used processes i.e. stakeholder's priority according to their heterogeneous viewpoints and mining historical data relevant to configurations of component-based SPLs products. The FRP consists of three main phases, i.e., stakeholder's priority and configuration analysis, mining configurations, and prioritization, as shown in Figure 2. All processes of FRP are updated in the repository for future use, to reduce the complexity in developing phases.

Figure 2

The proposed framework for configuration requirement prioritization



3.1. Stakeholder's Priority and Configuration Analysis

This phase is initiated when the stakeholders initiate a change in requirements of the system for updating the previous version or in a new series of a similar product. This phase is divided into three steps which are explained as follows.

3.1.1. Gathering Requirements

The requirements of similar domain products are gathered using user stories methods, in which each requirement is defined in the form of a story or need of requirements.

3.1.2. Stakeholders Priority

A large set of stakeholders involved in a component-based system with a heterogeneous perspective about system configuration according to their use. Therefore, we categorized requirements into three factors to extract priority according to heterogeneous perspectives or viewpoints of each stakeholder which increases complications in the case of homogenous behavior of stakeholders. These factors are described in Figure 3 and helps in extracting priority based on heterogeneous viewpoints and involvement of stakeholders instead of homogenous participation of stakeholders to reduce ambiguity, human effort, and incompleteness.

3.1.3. Configuration Model

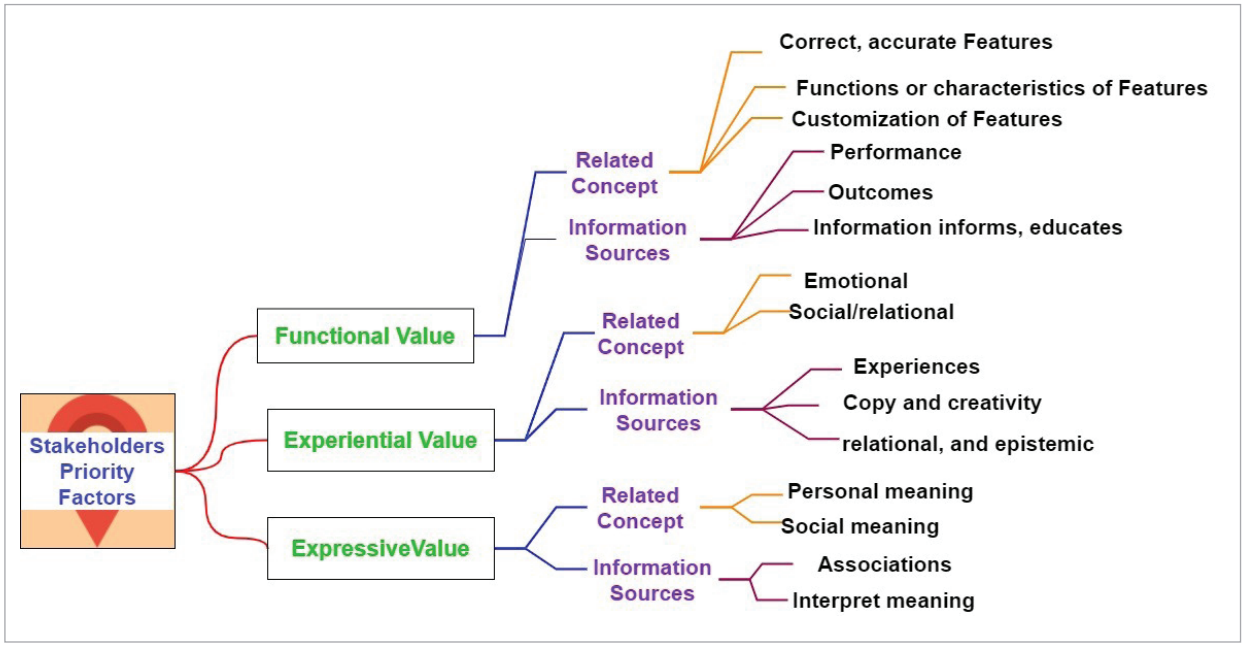
As in configurations, features of component-based systems are divided into core assets and variant features. The core assets are the common features that are used in every version of the configurable system. Contrarily, the variants are those features that differ in all versions of the system to provide new and updated functionalities, according to the current requirements of technology demands. The changes in both types of core assets and variant configurations are due to two processes.

Update – In this type, the core assets and variant configuration features can be added, deleted, and modified to ensure compatibility in accordance with the required changes.

New – The new configuration features are developed from scratch for user satisfaction due to changes.

The focus of configuration analysis is associating each configuration feature with a set of common-

Figure 3
Stakeholders' priority factors



alities and variabilities of system functionalities. For the construction of the configuration feature model based on system requirements, we used the FeatureIDE tool to identify valid and invalid configurations. The FeatureIDE tool is used for feature-oriented software development integrated with an Eclipse environment to supports all product development phases i.e. requirements analysis, domain analysis and design, implementation, and validation. Therefore, using the tool, configurations are generated based on changed requirements analysis. Any invalid configuration is refined by stakeholders and the process continues until validation of all configurations. The valid configurations are saved in the repository and retrieved in phase 2 for priority extraction by mining historical data.

3.2. Mining Configurations

In this phase, parameters for priority are generated based on association rule mining of historical information using the Weka tool.

Data mining techniques deal to extract hidden and vital aspects of information from data repositories using different methods or algorithms such as classification, clustering, and association rules. Classification is one of the important methods to assign classes for data according to predefined criteria by mapping future data patterns with unidentified patterns into one unique class. Clustering is partitioning a large set of similar information whereas, in the association, rule mining association rules are applied to find the connection between data items in a transactional database. Association rule mining is used to discover frequent associations based on the well-known apriori algorithm. In this context, to extract relevant data of frequent patterns accessed and changed the configuration from historical data apriori is most suitable [35, 40]. It is the most uncomplicated algorithm, which is employed for mining repetitive patterns from the transactional database [35, 40]. Hence, the algorithm is adopted in various problem-solving phases during software development with high configuration and complexity. This algorithm extracts the list of configurations of the highest priority, based on frequently accessed or changed configurations by users in the previous version of a similar domain product to ensure that these features are implemented before others. The technique reduces ambiguity, incompleteness, and human effort within limited resources, as well as increases satisfaction level. Algorithm 1 describes the steps of the algorithm in which CT and β are the inputs, and Fa is the output. If the value of a feature is less than β, it is removed from the list of features in the first phase of implementation, during system development. The two parameters used for frequent patterns mining from historical information are; a) frequently accessed configurations, b) frequently changed configurations. The reason for selecting two parameters is that all configurations of products are not used by most of the users or with no user or have the same frequently accessed frequencies then frequently changed frequency selected to avoid mismatch and ambiguities during implementations. Therefore, if any configuration of the product's frequently accessed information is not available or more than configuration has the same accessed frequency then we used their changed frequency for prioritization during the configuration mining information phase.

With the increase in awareness of data mining techniques, importance is increasing in mining information without ambiguity and completeness. The researchers and practitioners are trying to automate mining techniques to increase accuracy and reduce human efforts. Therefore, various tools available in the market for data mining such as RapidMiner, R software, and Weka tool. Most of the tools perform few or specific data mining tasks whereas the Weka tool supports almost all types of mining techniques.

Thus, in this work, we used the Weka (Waikato Environment for Knowledge Analysis) tool which is popular among researchers to implement machine learning techniques during research work and is written in the Java language with an exhaustive collection of data mining algorithms. It is portable to any modern platform and provides simple user graphical interfaces.

Then, in the next phase, mapped priority of identified configurations using the FeatureIDE tool, extracted from stakeholder's priority and by mining historical information for new priority.

Algorithm 1
Parameter settings for mining

Algorithm

INPUT: **CT** (Configurations in the repository)
$\mathbf{C_K}$ (Configurations where $\boldsymbol{k} = 1, 2, ..., \boldsymbol{n}$)
β (minimum frequency of accessed or changed Configurations)
OUTPUT: $\mathbf{F_a}$ (Frequently accessed or changed Configurations)
INITIATE
STEP 1: Search CT and generate C_{C1} (Count for each Configuration)
STEP 2:
Compare each $\mathbf{C_{c1}}$ configuration's count with a minimum frequency of accessed or changed configurations
In C_{c1}
for K = 1 to ***n***
if (count ($\mathbf{C_K}$) > = **β**)
then generate LR_1(list of C_k that satisfy β) and neglect others
end if
end for
STEP 3: Create C_{c2} (a set of configurations created by linking with itself)
$C_{C2} = C_{C1} \infty C_{C1}$ (C_{C1} link C_{C1})
STEP 4:
Repeat step1, step2, step3 until F_a obtained
Return F_a
END

3.3. Prioritization

In this phase, the obtained priority of configuration is saved in the repository for a component-based system, which is extracted in phases one and two of the process. The merging of both priority lists helps to identify missing configurations' priorities for correct interaction among system configurations. As a result, the new configuration's priority value is evaluated and updated in the repository for further implementation of configuration in the development phases.

4. Performance Evaluation

For the performance evaluation of the FRP, we considered three cases for providing a complete guideline to the industry into adopting the FRP process.

4.1. Experimental Design

We selected three datasets of the software development organizations which have workexperience in developing configurable component-based systems. In Table 1, the detail of datasets and representative properties are listed that we used during the controlled experimentation. These properties are required for requirements change management and correct prioritization of their configurations to identify the accurate combination of functional and non-functional features in each configuration option.

Table 1
Datasets properties

Detail	Dataset 1	Dataset 2	Dataset 3
LOC	226,35	120,897	30,300
Requirements	21,057	13,041	310
Configurations	728	1425	197
Versions	23	3	4

4.1.1. Controlled Experiment

For the experimentation process, we selected 30 participants to perform various tasks from data collection to the performance evaluation of FRP. These participants include graduate and undergraduate students of computer science with experience in both research and development. We developed and deployed representative systems to collect data for frequently assessed features. We divided these volunteers into two volunteer participant groups (PGs): PG1 and PG2. The members of both groups participated in requirement prioritization based on the following processes:

- FRP: Prioritization based on frequently accessed features, frequency, and stakeholder/ customer value creation priority factors.
- AHP: A statistical assessment process used to find an optimal substitute to set priority and resource allocation.
- Clustering: Categorization of data into groups with similar attributes to extract relevant data.

The AHP and Clustering techniques are the state of art practices and are suitable for the prioritization process as identified from the existing literature. These methods are mostly adopted and used for comparison by most of the novel existing approaches.

4.1.2. Research Questions

We proposed a prioritization framework for the configuration prioritization process based on the reuse and structuring of component-based systems using the data mining technique with stakeholder's diverse viewpoint analysis. To evaluate the FRP's performance in structuring and facilitating the component-based system after configurations requirements, we defined two research questions (RQs) as follows:

RQ1: Does the FRP improve the configuration prioritization process in component-based systems?

The RQ1 evaluates the efficiency of the FRP by using conventional methods with higher user satisfaction and less involvement of stakeholders for configuration prioritization. This is chosen for the reliability of the satisfaction-level statistical analysis.

RQ2: Does the FRP select the correct configuration during the prioritization process in component-based systems?

RQ2 is used to identify the effectiveness of the FRP using conventional methods with higher user satisfaction and less involvement of stakeholders by using evaluation metrics. The metrics used are precision (P) and recall (R) [26, 37] for analyzing the competence of FRP. P symbolizes the accuracy percentage of positive identification of features by using Equation 1, whereas R describes the correctly identified priority of selected features through Equation 2. In Equation 3, the F measure is used to select all correct configurations from the total available configurations in the current system version for prioritization.

$$P = N_{\text{Selected Features}} / N_{\text{Total Features}} \quad (1)$$

$$R = N_{\text{Correct Features}} / N_{\text{Selected Features}} \quad (2)$$

$$F = (2*P*R) / (P + R). \quad (3)$$

These metrics are used to verify the accuracy of the highest configurations which are selected for implementation firstly on the highest priority, after merging stakeholder priority and mining configuration priority. This configuration mining is based on frequently accessed or changed configurations in all versions of component-based systems.

For a complete explanation of FRP working, selected an online shopping system as an example case study. The system consists of several configurations and we selected a few configurations of application i.e. C_1: product list, C_2: payment mode, C_3: searching, and C_4: discounted product. The stakeholder initiated a change for discount rate (decrease in two product prices) and updated product list (delete 1 product from the product list, and add new 3 products). For the execution of said example, selected randomly 10 products available before changes, and on 4 products discounts were available out of these 10 products. Then developing team extracted the priority of these changed requirements of configurable components using three factors as described above (i.e. Functional value (FV), Experiential value (EV), and expressive value (XV)) and saved in the repository. Further, selected three stakeholders for priority assignment to execute example using FRP, AHP, and clustering. These stakeholders have a different perspective regarding configuration, as there is S_1: end-user of the product who used application, S_2: a developer who develops product and S_3: project manager who meet all business and organization goal during development (as described in Table 2). Subsequently, heterogeneous stakeholders, using the FRP, prioritized according to their perspective, and evaluated an average of three factors to find out the total priority of each stakeholder against every configuration.

The configuration model is generated based on the reuse factor by using the FeatureIDE tool by considering their relevant artifacts like requirements detail, code, and functions to identify valid and invalid configurations. The priority of selected configurations is based on the identified changes. Apriori algorithm applied using the Weka tool on selected configurations using the two historical information i.e. frequently accessed configurations (FAC) and frequently changed configurations (FCC). FAC of web-based shopping applications are product list, discounted product, and searching. As most of the end-user or visitors visit these pages and few of the visitors check the payment mode configuration component. Subsequently, FCC is products list and discounted products. The detail of stakeholder's priority and configuration mining priority is described in Table 2. The developing team merges both priorities to identify a new and correct priority list of configurations. Then, sorted configu-

Table 2
Example results

Conf. Id	Stakeholders (S) Priority					Mining Priority of Confi.			Merging Priority	New Priority
	S_Id	FV	EV	XV	Total	FAC	FCC	priority		
C_1	S_1	0.9	0.7	0.7	0.76	0.83	0.95	0.83	1.59	1.526
	S_2	0.6	0.8	0.5	0.63			0.83	1.46	
	S_3	0.7	0.8	0.6	0.70			0.83	1.53	
C_2	S_1	0.7	0.8	0.7	0.73	0.44	0.49	0.49	1.22	1.12
	S_2	0.6	0.7	0.5	0.60			0.49	1.09	
	S_3	0.9	0.5	0.3	0.56			0.49	1.05	
C_3	S_1	0.6	0.5	0.6	0.56	0.44	0.61	0.61	1.17	1.09
	S_2	0.5	0.5	0.4	0.46			0.61	1.07	
	S_3	0.6	0.3	0.4	0.43			0.61	1.04	
C_4	S_1	0.9	0.8	0.9	0.86	0.93	0.97	0.93	1.79	1.68
	S_2	0.7	0.7	0.3	0.56			0.93	1.49	
	S_3	0.8	0.9	0.8	0.83			0.93	1.76	

rations based on the highest priority value and sorted priority results of FRP are depicted in Table 3. The results of FRP show that C_4 has the highest frequency and C_3 the lowest value. Therefore, sequence of configuration priority using FRP are; $C_4 > C_1 > C_2 > C_3$.

Table 3
Example priority using FRP

Conf. Id	Sorted Priority
C_2	1.68
C_1	1.526
C_4	1.12
C_3	1.09

Similarly, the priority of all four-configurations calculated using AHP and Clustering techniques for comparing results with FRP results to prove its effectiveness. The results of both AHP and Clustering is described in Table 4. Hence, sequences of configuration priority using AHP are: $C_2 > C_3 > C_1 > C_4$ and Clustering are; $C_3 > C_2 > C_1$ and C_4. According to AHP, payment mode is more important than product list and discounted product, which is frequently accessed in all versions of the application. Similarly, in the Clustering technique, searching is more important than other configurations. The reason for ambiguity and differences in all methods is that Clustering and AHP used all three stakeholders as homogenous. Whereas, FRP considered all stakeholders as heterogeneous and got priority from stakeholders according to their perspectives. Thus, FRP outperforms as compared to other techniques and demonstrates that factors and historical data are important for improving configurations' priority in component-based systems.

Table 4
Example priority using AHP and Clustering

AHP		Clustering	
Conf. Id	Sorted Priority	Conf. Id Clusters	Sorted Priority
C_1	0.4	$Cluster_1$: C_1; C_4	0.44
C_3	0.5	$Cluster_3$: C_3	0.83
C_2	0.6	$Cluster_2$: C_2	0.61
C_4	0.3		

The experimental results of selected datasets are explained in the following section. In the experiment, all three techniques (i.e. FRP, AHP, and Clustering) were used to evaluate the priority of configurations.

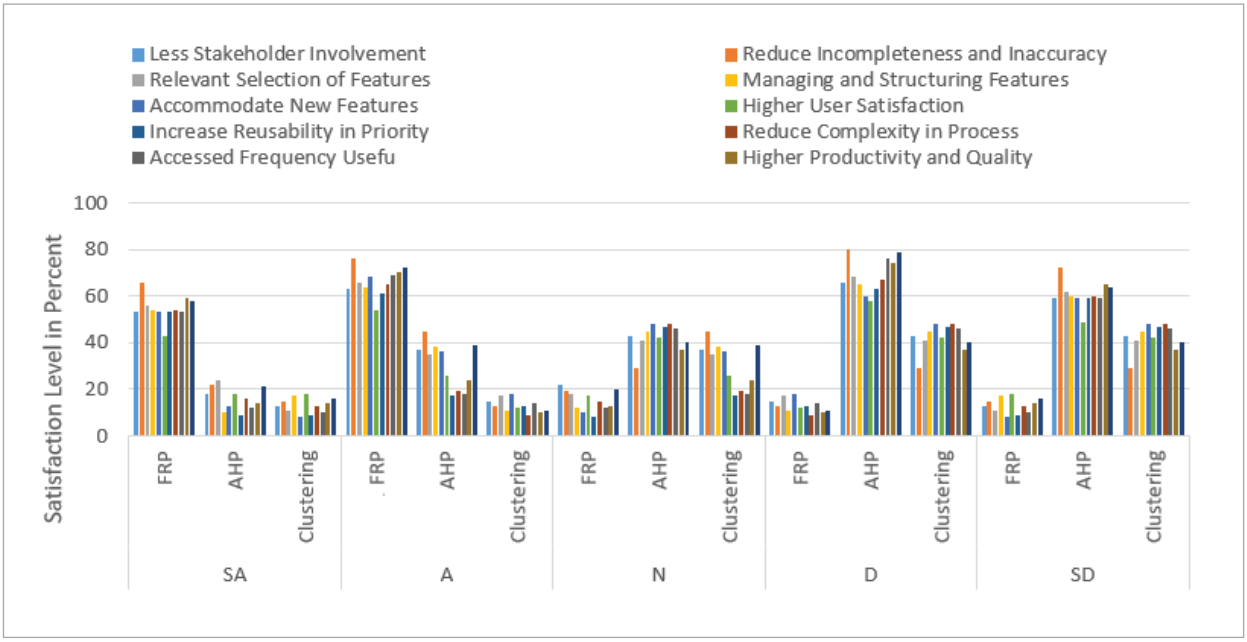
5. Results and Discussion

This section presents the experimental results and their discussion with respect to the defined RQs.

RQ1 is analyzed for efficiency and the FRP is compared with conventional methods using questionnaire-based participant review. Here, PG_1 behaves as a treatment group (TG), PG_2 behaves as a controlled group (CG), and vice versa. This implies that if PG_1 is used as the FRP method for prioritization of three datasets, it is considered as TG, while PG_2 is used as the AHP and clustering methods and considered as CG, and vice versa.

The following parameters, for improved prioritization methods, are selected from existing literature based on the existing techniques evaluation and challenges identified: less stakeholder involvement, reduce incompleteness and inaccuracy, relevant selection of features, managing and structuring features, accommodate new features, higher user satisfaction, increase reusability in priority, reduce complexity in the process, accessed frequency useful, higher productivity and quality, and customer value usefulness. Figure 4 shows the satisfaction level of each volunteer after using FRP, Clustering, and AHP methods. The x-axis describes the satisfaction points of all methods and the y-axis describes the percentage of satisfaction level. The satisfaction level is measured on a five-point Likert scale i.e. strongly agreed (SA), agreed (A), neutral (N), disagreed (D), and strongly disagreed (SD).

Figure 4
Satisfaction level



The overall analysis of the parameters while implementing FRP and the other two methods demonstrated a more than 50% satisfaction ratio for FRP as compared to others. The comparison results of Figure 5 show that the FRP performance is significantly increased compared with that of AHP and Clustering, with higher user satisfaction based on quality and fulfilling of customer needs.

Considering RQ2, P and R values are calculated to verify the effectiveness and accuracy of feature selection and their priority. The results are depicted in Figures 6 and 7 respectively. From the results, the values of P and R for FRP (i.e., greater than 0.95%) outperform those of AHP and Clustering, thus validating the effectiveness and accuracy of FRP

Figure 5
Reviews analysis

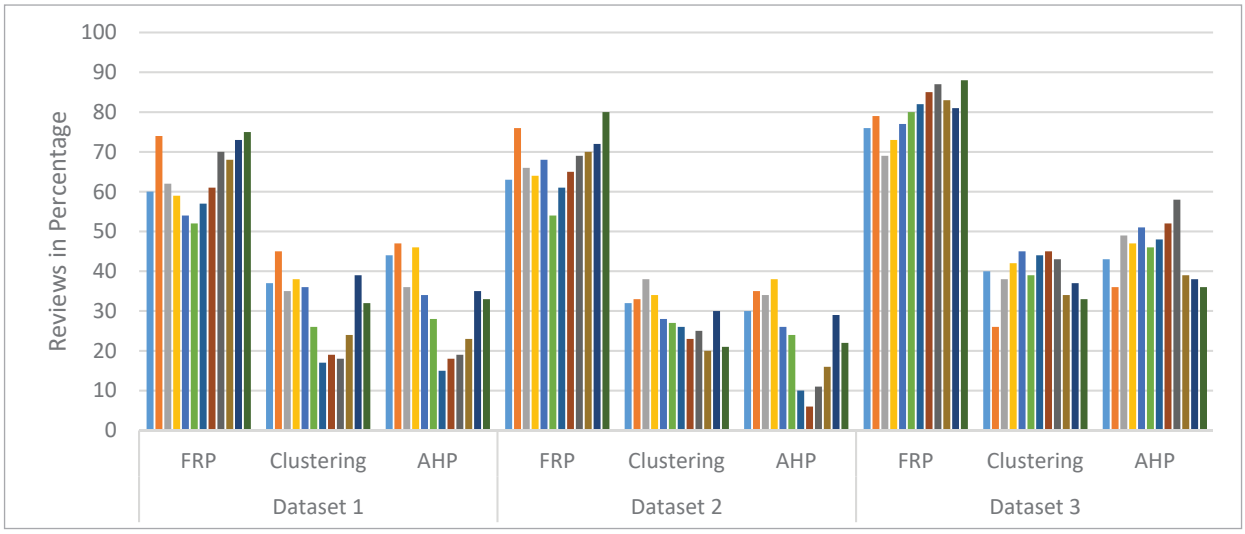


Figure 6
Comparison between P for FRP ad AHP

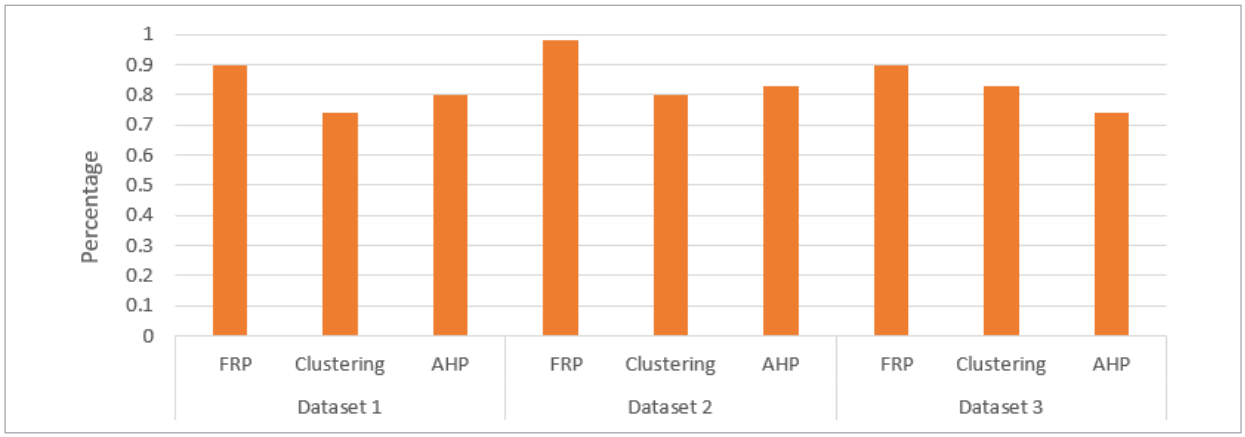


Figure 7
Comparison between R for FRP and AHP

The comparison of both P and R values in all three datasets or cases shows that FRP outperforms AHP. To further validate these results, we compared P, R, and F values for randomly selected versions of each case. The results show that FRP metrics are closer to 1 or 100% than the AHP and clustering metrics, as shown in Table 5 and Figure 8.

In Figure 8, the y-axis describes the percentage of all metrics in each version, while the x-axis depicts metrics distribution in all cases using all three methods i.e., FRP, Clustering, and AHP. The results and discussions show that FRP has more significantly identified features of highly configurable systems for prioritization to reduce stakeholder interaction and ambiguities in features implementation.

The parametric reviews and statistical analysis proved that the prioritization process improved significantly. The statistical analysis described in Table 6 shows that the difference in the mean and variance is greater between representative methods in all three cases, showing the significance of FRP over AHP and Clustering.

Figure 8
Versions of metrics comparison

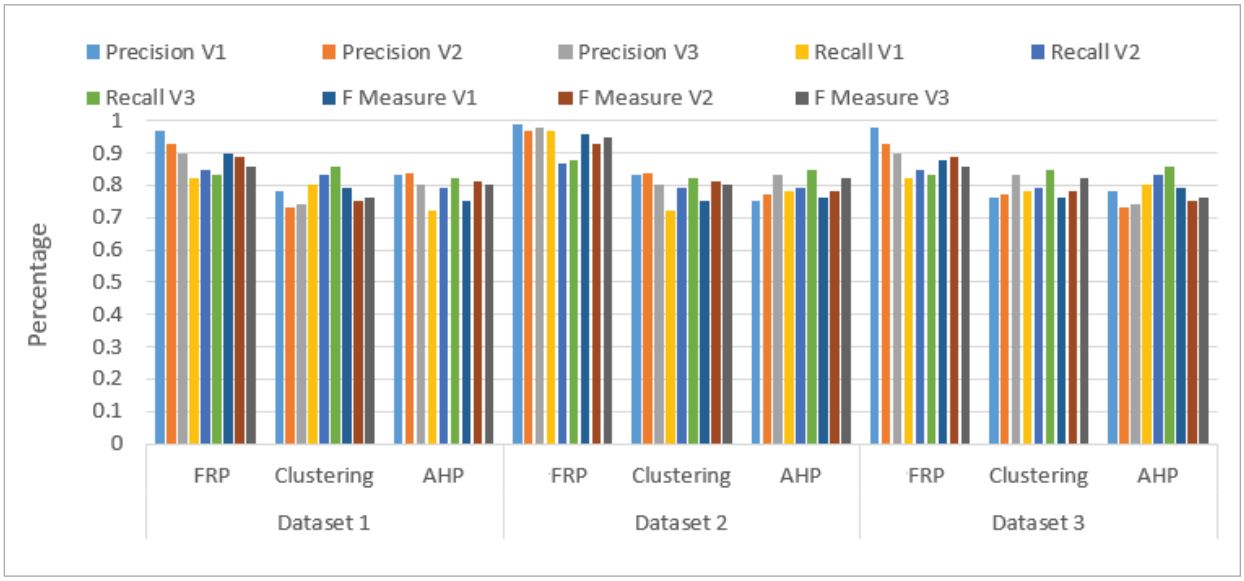


Table 5
Version comparisons

Metrics	Versions	Dataset 1			Dataset 2			Dataset 3		
		FRP	Clustering	AHP	FRP	Clustering	AHP	FRP	Clustering	AHP
Precision	V1	0.97	0.78	0.83	0.99	0.83	0.75	0.98	0.76	0.78
	V2	0.93	0.73	0.84	0.97	0.84	0.77	0.93	0.77	0.73
	V3	0.90	0.74	0.80	0.98	0.80	0.83	0.90	0.83	0.74
Recall	V1	0.82	0.80	0.72	0.97	0.72	0.78	0.82	0.78	0.80
	V2	0.85	0.83	0.79	0.87	0.79	0.79	0.85	0.79	0.83
	V3	0.83	0.86	0.82	0.88	0.82	0.85	0.83	0.85	0.86
F Measure	V1	0.90	0.79	0.75	0.96	0.75	0.76	0.88	0.76	0.79
	V2	0.89	0.75	0.81	0.93	0.81	0.78	0.89	0.78	0.75
	V3	0.86	0.76	0.80	0.95	0.80	0.82	0.86	0.82	0.76

Table 6
Statistical results of the one-sample test

	Test Value = 0					
	t	Df	Sig. (2-tailed)	Mean Difference	95% Confidence Interval of the Difference	
					Lower	Upper
Dataset1-FRP	68.371	14	.000	80.533	77.57	83.49
Dataset1-AHP	34.717	14	.000	46.233	40.67	46.11
Dataset1-Clustering	41.430	14	.000	38.450	39.78	47.14
Dataset2-FRP	63.713	14	.000	81.533	73.77	81.94
Dataset2-AHP	24.717	14	.000	44.533	40.67	48.40
Dataset2-Clustering	38.420	14	.000	37.640	41.47	47.40
Dataset3-FRP	47.137	14	.000	82.733	78.97	86.50
Dataset3-AHP	28.972	14	.000	42.867	39.69	46.04
Dataset3-Clustering	21.380	14	.000	34.170	36.44	43.42

The p-values are calculated using the following hypothesis:

H0: There is no significant difference in the requirement prioritization process followed by FRP, AHP, and Clustering.

H1: There is no significant difference in the accuracy followed by FRP, AHP, and Clustering.

For statistical analysis, we used SPSS 23 software, which helps to automate the process of desired statistical method applied to information. It can be seen from Table 3 that there are more differences among FRP mean values as compared to other techniques. The same patterns are followed in the case of both hypotheses either for requirement prioritization or accuracy, and both improved using the FRP. Table 3 shows that both requirement prioritization and accuracy are less significant in the case of AHP and Clustering as compared to FRP.

Therefore, the FRP is capable of producing an accurate priority list as compared to AHP and Clustering. Further, it can be observed that the p-values are more than 0.05 which implies that the null hypothesis can be accepted, stating that the two techniques are producing similar results. Overall, FRP is found to be more computationally efficient in comparison to Clustering and AHP, by effective, comparable, and assuring correct final ranking to support reliability and robustness to ambiguity. FRP also helps to resolve the issue of conflicts, ambiguity, incompleteness collaboration, and dependency among stakeholders and product managing teams within limited resources.

5.1. Threats to Validity

There may be certain limitations that may influence results and need to be identified for the FRP working. Therefore, the main threat is relevant to the accurate sequence of configurations and the biasedness of experimental data. To reduce the threat, relevant to accuracy, we checked the accuracy, F- Measure, and carried out statistical analysis for correct interaction of configurations and their features using the FRP. For reliability, compared results of FRP with other techniques and found that other techniques are comparatively less accurate. The external and scalability is another threat for mitigation of FRP evaluation. We selected three industrial projects for the usefulness of the FRP in different domains. Therefore, detailed steps of the FRP are explained with the industrial example to apply to different types of projects.

For the correct configuration priority list in component-based systems, used multi-criteria-based prioritization and dealt with a heterogeneous perspective of stakeholders. The three datasets used for evaluation of the effectiveness of FRP proved that heterogeneous stakeholder's priority and historical information mining improved configuration priority. For

rigorous verification of theory and results relations, we constructed an empirical study review based on existing industrial research work from literature and presented the FRP as a solution for the improvement of the configuration prioritization process.

6. Conclusion and Future Work

In this work, we proposed a framework for prioritization of configuration for component-based systems considering accuracy, reusability, and less stakeholder involvement. The proposed framework improves the configuration prioritization process and mitigates challenges during the version update or new series of similar product development. Hence, from the review of existing literature, firstly we identify that there is a need to include the priority of stakeholders, based on their heterogeneous perspectives instead of homogenous perspective. Secondly, frequently accessed and changed configurations in previous versions or series of the same product are used as historical information for prioritization of new configurations. Subsequently, in the FRP, the stakeholder priority factor is used to find stakeholder viewpoints with a required value in the target marketplace. Therefore, for stakeholder's heterogeneous perspective, analyzed three factors (i.e. functional, experiential, and expressive values), and for managing priority in a large dataset used apriori technique for frequent pattern extraction and categorization. Then merged both priorities to determine accurate priorities for correct implementation of configurations. The effectiveness of the FRP is evaluated through an experimental study, and implementation on three industrial datasets, and compared with other techniques i.e. AHP and Clustering. The results depicted noteworthy improvement in terms of satisfaction level and configuration, correct implementation with the highest recall value of FRP than that of the AHP and Clustering methods in terms of ranking of a configurable product.

Hence, this research provides a direction to industry and researchers to manage the prioritization and validation of configuration features in a component-based system and to extend the proposed framework to improve configuration testing in a distributed environment with the improvement of requirement management and prioritization process.

Funding

The work reported in this manuscript was supported by the National Natural Science Foundation of China under Grant 61672080.

References

1. Anand, R. V., Dinakaran, M. Handling Stakeholder Conflict by Agile Requirement Prioritization Using Apriori Technique. Computers and Electrical Engineering, 2017, 61, 126-136. https://doi.org/10.1016/j.compeleceng.2017.06.022
2. Angerer, F., Prähofer, H., Grünbacher, P. Modular Change Impact Analysis for Configurable Software. In 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME), Raleigh, NC, USA, October 2016, 468-472. https://doi.org/10.1109/ICSME.2016.69
3. Arias, M., Buccella, A., Cechich, A. A Framework for Managing Requirements of Software Product Lines. Electronic Notes in Theoretical Computer Science, 2018, 339, 5-20. https://doi.org/10.1016/j.entcs.2018.06.002
4. Arrieta, A., Wang, S., Sagardui, G., Etxeberria, L. Test Case Prioritization of Configurable Cyber-Physical Systems with Weight-Based Search Algorithms. In Proceedings of the Genetic and Evolutionary Computation Conference, July 2016, 1053-1060. https://doi.org/10.1145/2908812.2908871
5. Ali, A., Hafeez, Y., Hussain, S., Yang, S. Role of Requirement Prioritization Technique to Improve the Quality of Highly-Configurable Systems. IEEE Access, 2020, 8, 27549-27573. https://doi.org/10.1109/ACCESS.2020.2971382
6. Artz, P., van de Weerd, I., Brinkkemper, S., Fieggen, J. Productization: Transforming from Developing Customer-Specific Software to Product Software. In International Conference of Software Business, Springer, Berlin, Heidelberg, June 2010, 90-102. https://doi.org/10.1007/978-3-642-13633-7_8
7. Asif, S. A., Masud, Z., Easmin, R., Gias, A. U. SAFFRON: A Semi-Automated Framework for Software Requirements Prioritization. International Journal of Ad-

vanced Computer Science and Applications (IJACSA), 2017, 8(12), 491-499. https://doi.org/10.14569/IJACSA.2017.081265

8. Ayala, C., Nguyen-Duc, A., Franch, X., Höst, M., Conradi, R., Cruzes, D., Babar, M. A. System Requirements-OSS Components: Matching and Mismatch Resolution Practices-An Empirical Study. Empirical Software Engineering, 2018, 23(6), 3073-3128. https://doi.org/10.1007/s10664-017-9594-1
9. Baker, P., Harman, M., Steinhofel, K., Skaliotis, A. Search Based Approaches to Component Selection and Prioritization for the Next Release Problem. In 2006 22nd IEEE International Conference on Software Maintenance, Philadelphia, PA, USA, September 2006, 176-185. https://doi.org/10.1109/ICSM.2006.56
10. Barney, S., Aurum, A., Wohlin, C. A Product Management Challenge: Creating Software Product Value Through Requirements Selection. Journal of Systems Architecture, 2008, 54(6), 576-593. https://doi.org/10.1016/j.sysarc.2007.12.004
11. Bhushan, M., Goel, S., Kaur, K. Analyzing Inconsistencies in Software Product Lines Using an Ontological Rule-Based Approach. Journal of Systems and Software, 2018, 137, 605-617. https://doi.org/10.1016/j.jss.2017.06.002
12. Chatzipetrou, P., Papatheocharous, E., Wnuk, K., Borg, M., Alégroth, E., Gorschek, T. Component Attributes and their Importance in Decisions and Component Selection. Software Quality Journal, 2019, 1-27. https://doi.org/10.1007/s11219-019-09465-2
13. Diamantopoulos, T., Symeonidis, A. Enhancing Requirements Reusability Through Semantic Modeling and Data Mining Techniques. Enterprise information systems, 2018, 12(8-9), 960-981. https://doi.org/10.1080/17517575.2017.1416177
14. Eyal-Salman, H., Seriai, A. D., Dony, C. Feature-to-Code Traceability in a Collection of Software Variants: Combining Formal Concept Analysis and Information Retrieval. In 2013 IEEE 14th International Conference on Information Reuse and Integration (IRI), San Francisco, CA, USA, August 2013, 209-216. https://doi.org/10.1109/IRI.2013.6642474
15. Gupta, A., Gupta, C. A Novel Collaborative Requirement Prioritization Approach to Handle Priority Vagueness and Inter-Relationships. Journal of King Saud University-Computer and Information Sciences, 2019, 1-10. https://doi.org/10.1016/j.jksuci.2019.12.002
16. Harzer, T. S. Value Creation Through Mass Customization: An Empirical Analysis of the Requisite Strategic Capabilities (Doctoral dissertation, Hochschulbibliothek der Rheinisch-Westfälischen Technischen Hochschule Aachen), 2013. http://publications.rwth-aachen.de/record/229868/files/4614.pdf
17. Hasan, M. S., Mahmood, A. A., Alam, M. J., Hasan, S. N., Rahman, F. An Evaluation of Software Requirement Prioritization Techniques. International Journal of Computer Science and Information Security (IJCSIS), 2010, 8(9). https://sites.google.com/site/ijcsis/vol-8-no-9-dec-2010
18. Hendradjaya, B. A Proposal for New Software Testing Technique for Component Based Software System. International Journal on Electrical Engineering & Informatics, 2018, 10(1), 60-78. https://doi.org/10.15676/ijeei.2018.10.1.5
19. Hujainah, F., Bakar, R. B. A., Abdulgabber, M. A., Zamli, K. Z. Software Requirements Prioritisation: A Systematic Literature Review on Significance, Stakeholders, Techniques and Challenges. IEEE Access, 2018, 6, 71497-71523. https://doi.org/10.1109/ACCESS.2018.2881755
20. Inoki, M., Kitagawa, T., Honiden, S. Application of Requirements Prioritization Decision Rules in Software Product Line Evolution. In 2014 IEEE 5th International Workshop on Requirements Prioritization and Communication (RePriCo), Karlskrona, Sweden, August 2014, 1-10. https://doi.org/10.1109/RePriCo.2014.6895216
21. Jalila, A., Mala, D. J. Software Components Prioritization Using OCL Formal Specification for Effective Testing. In 2013 International Conference on Recent Trends in Information Technology (ICRTIT), Chennai, India, July 2013, 714-720. https://doi.org/10.1109/ICRTIT.2013.6844288
22. Jamshidi, P., Velez, M., Kästner, C., Siegmund, N., Kawthekar, P. Transfer Learning for Improving Model Predictions in Highly Configurable Software. In 2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), Buenos Aires, Argentina, May 2017, 31-41. https://doi.org/10.1109/SEAMS.2017.11
23. Jeeva, S. C., Rajsingh, E. B. Intelligent Phishing URL Detection Using Association Rule Mining. Human-centric Computing and Information Sciences, 2016, 6(1), 1-19. https://doi.org/10.1186/s13673-016-0064-3
24. Kaindl, H., Mannion, M. Software Reuse and Reusability Based on Requirements: Product Lines, Cases and Feature-Similarity Models. In 2018 IEEE 26th International Requirements Engineering Conference (RE), Banff, AB, Canada, August 2018, 510-511. https://doi.org/10.1109/RE.2018.00010

25. Khalique, F., Butt, W. H., Khan, S. A. Creating Domain Non-Functional Requirements Software Product Line Engineering Using Model Transformations. In 2017 International Conference on Frontiers of Information Technology (FIT), Islamabad, Pakistan, December 2017, 41-45. https://doi.org/10.1109/FIT.2017.00015

26. Liu, G., Xie, L., Chen, C.H. Unsupervised Text Feature Learning via Deep Variational Auto-Encoder. Information Technology and Control, 2020, 49(3), 421-437. https://doi.org/10.5755/j01.itc.49.3.25918

27. Khedr, A. E., Idrees, A. M., Hegazy, A. E. F., El-Shewy, S. A Proposed Configurable Approach for Recommendation Systems Via Data Mining Techniques. Enterprise Information Systems, 2018, 12(2), 196-217. https://doi.org/10.1080/17517575.2017.1293301

28. Martinelli, F., Mercaldo, F., Nardone, V., Orlando, A., Santone, A., Vaglini, G. Model Checking Based Approach for Compliance Checking. Information Technology and Control, 2019, 48(2), 278-298. https://doi.org/10.5755/j01.itc.49.3.25918

29. Cai, S., Li, Q., Li, S., Yuan, G., Sun, R. WMFP-Outlier: An Efficient Maximal Frequent-Pattern-Based Outlier Detection Approach for Weighted Data Streams. Information Technology and Control, 2019, 48(4), 505-521. https://doi.org/10.5755/j01.itc.48.4.22176

30. Li, Y., Yue, T., Ali, S., Zhang, L. Enabling Automated Requirements Reuse and Configuration. Software and Systems Modeling, 2019, 18(3), 2177-2211. https://doi.org/10.1007/s10270-017-0641-6

31. Linsbauer, L., Lopez-Herrejon, R. E., Egyed, A. Variability Extraction and Modeling for Product Variants. Software and Systems Modeling, 2017, 16(4), 1179-1199. https://doi.org/10.1007/s10270-015-0512-y

32. Mahmood, S., Khan, M. A. A Degree Centrality-Based Approach to Prioritize Interactions of Component-Based Systems. In 2012 International Conference on Computer and Information Science (ICCIS), Kuala Lumpeu, Malaysia, June 2012, 2, 863-867. https://doi.org/10.1109/ICCISci.2012.6297147

33. Markiegi, U. Test Optimisation for Highly-Configurable Cyber-Physical Systems. In Proceedings of the 21st International Systems and Software Product Line Conference, September 2017, Volume B, 139-144. https://doi.org/10.1145/3109729.3109745

34. Meedeniya, D. A., Rubasinghe, I. D., Perera, I. Software Artefacts Consistency Management Towards Continuous Integration: A Roadmap. International Journal of Advanced Computer Science and Applications (IJACSA), 2019, 10(4). https://doi.org/10.14569/IJACSA.2019.0100411

35. Nafie Ali, F. M., Mohamed Hamed, A. A. Usage Apriori and Clustering Algorithms in WEKA Tools to Mining Dataset of Traffic Accidents. Journal of Information and Telecommunication, 2018, 2(3), 231-245. https://doi.org/10.1080/24751839.2018.1448205

36. Nguyen, S., Nguyen, H., Tran, N., Tran, H., Nguyen, T. Feature-Interaction Aware Configuration Prioritization for Configurable Code. In 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE), San Diego, CA, USA, November 2019, 489-501. https://doi.org/10.1109/ASE.2019.00053

37. Norouzi, M., Souri, A., Samad Zamini, M. A Data Mining Classification Approach for Behavioral Malware Detection. Journal of Computer Networks and Communications, 2016. https://doi.org/10.1155/2016/8069672

38. Oliveira, N., Barbosa, L. S. Self-Adaptation by Coordination-Targeted Reconfigurations. Journal of Software Engineering Research and Development, 2015, 3(1), 1-31. https://doi.org/10.1186/s40411-015-0021-2

39. Pacheco, C. L., Garcia, I. A., Calvo-Manzano, J. A., Arcilla, M. A Proposed Model for Reuse of Software Requirements in Requirements Catalog. Journal of Software: Evolution and Process, 2015, 27(1), 1-21. https://doi.org/10.1002/smr.1698

40. Park, S. H., Synn, J., Kwon, O. H., Sung, Y. Apriori-Based Text Mining Method for the Advancement of the Transportation Management Plan in Expressway Work Zones. The Journal of Supercomputing, 2018, 74(3), 1283-1298. https://doi.org/10.1007/s11227-017-2142-3

41. Pereira, J. A., Matuszyk, P., Krieter, S., Spiliopoulou, M., Saake, G. A Feature-Based Personalized Recommender System for Product-Line Configuration. In Proceedings of the 2016 ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences, October 2016, 120-131. https://doi.org/10.1145/2993236.2993249

42. Perini, A., Susi, A., Avesani, P. A Machine Learning Approach to Software Requirements Prioritization. IEEE Transactions on Software Engineering, 2012, 39(4), 445-461. https://doi.org/10.1109/TSE.2012.52

43. Pitangueira, A. M., Maciel, R. S. P., Barros, M. Software Requirements Selection and Prioritization Using SBSE Approaches: A Systematic Review and Mapping of the Literature. Journal of Systems and Software, 2015, 103, 267-280. https://doi.org/10.1016/j.jss.2014.09.038

44. Qu, X. Configuration Aware Prioritization Techniques in Regression Testing. In 2009 31st International Con-

ference on Software Engineering-Companion, Vancouver, BC, Canada, Volume, May 2009, 375-378. https://doi.org/10.1109/ICSE-COMPANION.2009.5071025

45. Ra'Fat, A., Seriai, A., Huchard, M., Urtado, C., Vauttier, S., Salman, H. E. Feature Location in a Collection of Software Product Variants Using Formal Concept Analysis. In International Conference on Software Reuse, Springer, Berlin, Heidelberg, June 2013, 302-307. https://doi.org/10.1007/978-3-642-38977-1_22
46. Sagar, K., Saha, A. Qualitative Usability Feature Selection with Ranking: A Novel Approach For Ranking the Identified Usability Problematic Attributes for Academic Websites Using Data-Mining Techniques. Human-centric Computing and Information Sciences, 2017, 7(1), 29. https://doi.org/10.1186/s13673-017-0111-8
47. White, J., Galindo, J. A., Saxena, T., Dougherty, B., Benavides, D., Schmidt, D. C. Evolving Feature Model Configurations in Software Product Lines. Journal of Systems and Software, 2014, 87, 119-136. https://doi.org/10.1016/j.jss.2013.10.010
48. Wiesner, S., Marilungo, E., Thoben, K. D. Cyber-Physical Product-Service Systems-Challenges for Requirements Engineering. International journal of automation technology, 2017, 11(1), 17-28. https://doi.org/10.20965/ijat.2017.p0017
49. Zhang, Y., Guo, J., Blais, E., Czarnecki, K. Performance Prediction of Configurable Software Systems by Fourier Learning (t). In 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE), Lincoln, NE, USA, November 2015, 365-373. https://doi.org/10.1109/ASE.2015.15