International Journal of Electrical and Computer Engineering (IJECE)
Vol. 9, No. 4, August 2019, pp. 2648~2658
ISSN: 2088-8708, DOI: 10.11591/ijece.v9i4.pp2648-2658

Feature model configuration based on two-layer modeling in Software Product Lines

Elham Darmanaki Farahani[1], Jafar Habibi[2]
[1]Kish International Campus, Sharif University of Technolog, Iran
[2]Department of Computer Engineering, Sharif University of Technology, Iran

Article Info

Article history:

Received Jun 4, 2018
Revised Mar 4, 2019
Accepted Mar 12, 2019

Keywords:

Configuration management
Feature model
Implementation infrastructure
Software product line

ABSTRACT

The aim of the Software Product Line (SPL) approach is to improve the software development process by producing software products that match the stakeholders' requirements. One of the important topics in SPLs is the feature model (FM) configuration process. The purpose of configuration here is to select and remove specific features from the FM in order to produce the required software product. At the same time, detection of differences between application's requirements and the available capabilities of the implementation platform is a major concern of application requirements engineering. It is possible that the implementation of the selected features of FM needs certain software and hardware infrastructures such as database, operating system and hardware that cannot be made available by stakeholders. We address the FM configuration problem by proposing a method, which employs a two-layer FM comprising the application and infrastructure layers. We also show this method in the context of a case study in the SPL of a sample E-Shop website. The results demonstrate that this method can support both functional and non-functional requirements and can solve the problems arising from lack of attention to implementation requirements in SPL FM selection phase.

Corresponding Author:

Elham Darmanaki Farahani,
Kish International Campus, Sharif University of Technolog,
Iran
Email: efarahani@ce.sharif.edu

1. INTRODUCTION

Software Product Line (SPL) is an important approach in the field of software engineering. It is a systematic approach to software reuse that focuses on managing families of related software products. In an SPL, a set of related products are produced through the composition of reusable core assets together with product-specific variable assets[1] [1].

In particular, it is argued that the nature of a SPL is to manage the commonality and variability of products by means of a ''Requirements Engineering (RE)" process [2]. RE is concerned with the real-world goals for, functions of and constraints on software systems[3]. Compared with RE for a single custom-built system, RE for a family of software-intensive systems focuses more on systematic reuse, not only from the technical perspective, but from the organizational, marketing, and process perspectives as well [4].

The techniques, most notably the modeling techniques, are different from single-system RE. Single-system requirements are often modeled from the use perspective, e.g., use cases, sequence diagrams, etc., SPL requirements are modeled from the reuse perspective by explicitly representing the commonality and variability information, e.g., feature models, orthogonal variability models, etc. In this paper, our focus is on Feature Model (FM).

As can be seen in Figure 1, the Software Product Line Engineering (SPLE) framework [1] has two major processes: Domain Engineering and Application Engineering. Domain Engineering involves modeling target domain and producing a set of core assets. On the other hand, Application Engineering involves developing a domain-specific software product through the customization of artifacts that are developed in the domain engineering phase. The Application Requirements Engineering sub-process encompasses all activities necessary for developing the application requirements specification. FM configuration is the main activity of this phase.

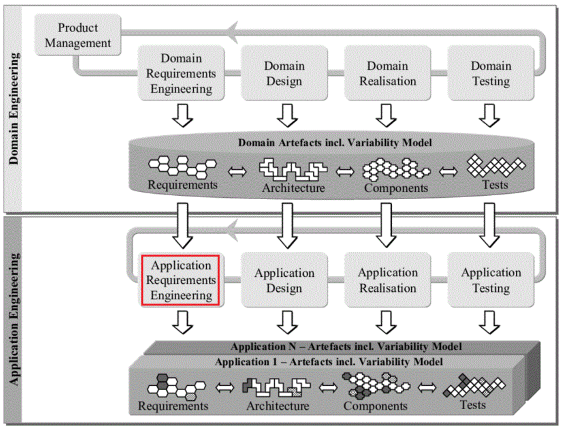

Figure 1. The SPLE Framework

A FM that describes a range of products generated from an SPL has a key role in the configuration process of the SPL. An FM consists of: i) features and sub-features organized in a feature tree, and ii) optional constraints such as “excludes” or “requires” to describe the products of a product line in terms of the features that should be excluded (“excludes” constraints) and/or needed (“requires” constraints) by each product. Each feature in a Feature Model represents a property of a product that will be visible to the product user. Selecting a set of desirable features based on stakeholders’ needs is a complex process because:

1) There are normally some constrains between features that must be considered during the feature selection process by stakeholders.
2) In addition to functional requirements (FRs), stakeholders may have some non-functional requirements (NFPs) as well. However, it may not be straightforward to express how the NFRs can be satisfied in terms of features in FM.
3) Stakeholders may have some restrictions in the implementation of the features in FM due to, for example, lack of adequate hardware infrastructure.

The need for addressing these problems leads to increased complexity of the FM configuration process. Therefore, selecting the best set of features while considering the stakeholders’ requirements and implementation infrastructure is a hard task.

Due to the importance of the issue of RE in SPL, many studies have been done.According to [5] in this area, there is a lack of tool support and comparative studies [5]. Also in another research, it has been stated that inappropriate communication and communication, long repetition cycles, and lack of compliance and flexibility in RE phase of SPL engineering could increase effort and mitigate disruption during product development [6]. In other study, because of importatnce of RE phase in quality of final produts, the security and related verification method in RE has been discussed [7].

Additionally, various configuration methods have previously been developed to help FM configuration (The most important issue in RE phase) by automating the selection of features to satisfy FRs, NFRs and constraints [8]-[10]. Some others have focused on the constraints satisfaction problem and proposed a method to build optimal configurations [11]. The main problem with this approach is performance inefficiency. Another technique is based on staged configuration of FM that gives more importance to the role of stakeholders in feature selection but could not solve the NFRs satisfaction problem [12]. Among the solutions proposed in this area the automated planning in [13] is more complete than the others because it covers FRs, NFRs and constraints satisfaction and also automates the feature selection process but it does not solve complexity of simultaneous presentation of Application and Infrastructure features in one-layer FM.

The above issues motivated us to address the following research questions: How can we determine the infrastructure needed for implementation of selected FRs and NFRs by stakeholders in FM? And if any conflict is found between the infrastructures needed versus that available, how should the selected features be changed to resolve the conflict?

To address these questions, we looked into some FM design techniques and found a multi-layer SPL with a reference model concept in [14] which is a common approach for manage highly complex product families. In [14], a multi-level feature trees has been proposed that consist of a tree of feature models in which the parent model serves as a reference feature model for its children. Base on proposed model in [14], we guessed that the two-layer form of SPL together with the vertical composition can help us to first define constraints between the features in application layer of SPL and then map the features to the needed infrastructure for their implementation.

So we propose a two-layer FM, comprising an “Application layer” and an “Infrastructure layer”. Application layer is to include functional and non-functional features of SPL and the purpose of infrastructure layer is to deal with the hardware and network requirements that have a major role in implementation of any product instances.

In addition to constraints applied to each feature in the FM of application layer, we can define constraints between the two layers of FM, and thereby, specify the necessary infrastructure for each set of stakeholders’ requirements.

In the context of FM configuration, the main contributions of this paper are as follows:

- A new method to represent FM as a two-layer model with the ability to specify the constraints between the features in the same level (called “Inner Constraints”) and also constraints between the features in different layers (called “Intra constraints”),
- An easy way to show NFRs in application layer of FM,
- We also show how the stakeholders can be helped to select NFRs from FMs to reflect the infrastructure necessary for NFRs’ implementation.

The rest of this paper is organized as follows: section 2 gives an overview of the basic related concepts; in section 3 we discuss the challenges in current FM configuration methods and in section 4 we propose a new method that covers all of the problems described in section 3; this is followed with a case study of the proposed method in section 5. Section 6 systematically compares our approach with related works, and finally, the paper concludes in section 7.

2. FOUNDATION

In this section we describe the basic concepts used throughout the paper.

2.1. Feature models (FMs)

In software development, FM is a structured representation of all the products (generated by an SPL) in terms of their “features”. FMs are widely used especially during application requirements engineering phase, where the output of this phase can be used in producing other assets such as documents, architecture definition, or pieces of code.

According to FODA in [15], a feature model has a tree-like structure that visually depicts features and also their dependencies as constraints. The relationship between a parent feature and its child features in FM are typically classified as follows:

- Mandatory – child feature is required.
- Optional – child feature is optional.
- Or – at least one of the child-features must be selected.
- Alternative (xor) – one (and only one) of the child-features must be selected

Also we can define some cross-tree constraints between the features in FM. The most common constraints of this type are:

- A requires B – The selection of A in a product implies the selection of B.
- A excludes B – A and B cannot be part of the same product.

In an FM the main functionalities of products that are common between all products derived from the SPL are specified as mandatory features. Figure 2 shows a subset of the FM of the example E-Shop website. Here, the “catalogue” functionality is assumed to be the minimum facility of any E-Shops, so it is set to mandatory in the FM, furthermore the “Bank transfer” feature requires the “High security” feature.

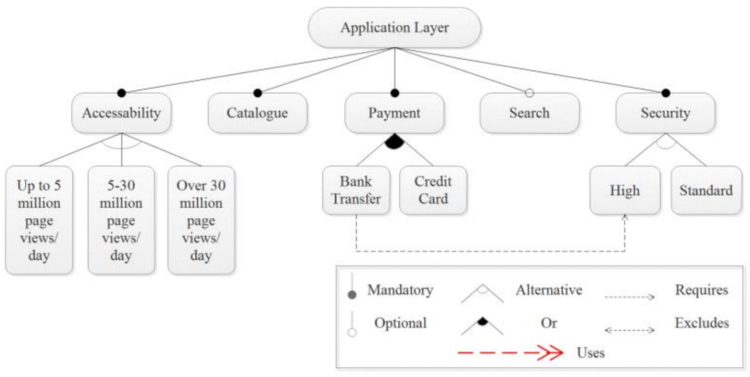

Figure 2. A subset of E-Shop FM

2.2. Functional and non-functional requirements

As a part of the software development process, requirements engineering involves identification, representation, documentation, and the management of the set of needs, desired features and preferences of the stakeholders [16].

In a software system, requirements are categorized into functional and non-functional groups [16]. The term FR refers to the characteristics that specify the functions the system must perform, while NFR refers to the constraints on how the system must perform those functions. In general, FRs describe the behavior of the system whereas NFRs elaborate on the performance characteristic of the system.

NFRs are mostly known as system qualities and typically fall into areas such as: efficiency, security and accessibility. An example of a functional requirement would be: “A system must send an email whenever a certain condition is met” and a related non-functional requirement for this system may be: “Emails should be sent with a latency of no greater than 12 hours after the related condition is met.”

Representation of FRs can be achieved through features in an FM. But representing the NFRs in an FM is not a simple task, although there are proposals for how this can be achieved and presented to stakeholders [13, 17]. One of the contributions of this paper is a method for representing NFRs in FMs in a simple way. This will be discussed in the next sections.

3. PROBLEM STATEMENT

This section highlights the major challenge involved in selecting the necessary features from the FM by stakeholders. As mentioned in section 1, one of the main problems in selecting the desired features is the mismatch between the application requirements and the available infrastructure. This problem occurs because the stakeholders can only see the features (Functional or Non-Functional) in the FM but cannot have any information about the infrastructure (Network and Hardware) required for implementation of all the selected features.

For example, let us consider an FM for the SPL of a website where one of the important non-functional requirements can be accessibility in face of a high number of simultaneous online visitors per minute. If the proper hardware and network configuration is not provided, the website could become inaccessible to the users during heavy traffic periods. To avoid this problem, proper and adequate hardware infrastructure should be provided in the relevant parts of the system.

In the FM of this website, we can have a feature named “Accessibility” and stakeholders can select the predicted number of simultaneous visitors per minute. But the problem arises when the stakeholders select the predicted number in the FM without having any information about the infrastructure required to support the predicted number of number of simultaneous visitors. It is possible that the stakeholders select a high number of simultaneous visitors in the FM but in practice it may not be possible to provide the hardware and network infrastructure required to handle the predicted of simultaneous visitors.

As a result, all the stakeholders’ requirements related to the selected features, especially the NFRs in the FM, could not necessarily be implemented. To solve this problem, we must find a way to present to the stakeholders the infrastructure required for each special functional or non-functional feature in the application requirements engineering phase. In the next section we describe our method that could solve this major challenge.

4. PROPOSED FM CONFIGURATION METHOD

To solve the problem described in section 3, in this section we propose a new method for designing FMs in any SPLs. To address the challenges that stakeholders face in feature selection in application requirement engineering phase, we must find a method to able to simultaneously show to stakeholders the functional and non-functional features and the infrastructure required to support those features. In this way the stakeholders can view the properties, operational capabilities and the available infrastructure at the same time. So in the following we describe both a new method for feature modelling, called “Two-layer FM”, and also a related algorithm describing the steps involved in FM configuration.

4.1. Two-layer FM

This section introduces a new method for feature modelling called “Two-layer FM” consisting of two layers that each one is a FM (One for application features and another for infrastructure features). The dependencies between the features in the same layer of the FM are expressed using “Inner constraints” which take the form of “Requires” or “Excludes” relations. Furthermore, the constrains between the features in the application and infrastructure layers are defined via the “Intra constraints” which provide the “Uses” relation between one feature in application layer and another one in infrastructure layer.

4.2. Proposed configuration algorithm

This section describes the new FM configuration algorithm based on our proposed two-layer FM. The algorithm specifies the steps involved in feature selection leading to the final customized FM.

```
FMC Algorithm
Input: Two-layer FM
Output: Final Customized FM
  (1) Stakeholders select the desired functional and non-functional features from application layer of FM.
  (2) Stakeholders select the possible implementation equipment (as hardware and network) from
infrastructure layer of FM.
  (3) Check whether the inner constraints in both application and infrastructure layers are satisfied.
  If there is any conflict between the selected features and constraints
     Until there is no conflict do:
        (3-1) Send error message to stakeholders to change the selected features until there is no conflict.
  (4) Check the intra constraints between application and infrastructure layers are satisfied by one of the
existing methods as SAT solver [18] or FMVA [19].
  If there is any conflict between a selected feature from application layer and a selected available
equipment from the infrastructure layer (which means that the intra constraints were not satisfied), two
solutions will be proposed to stakeholders (only one solution can be selected)
        (4-1) Stakeholders can change the selected features in application layer based on ticked available
        equipment for implementation by going backward from infrastructure to application layer.
        (4-2) Stakeholders can provide the equipment required for implementation of the corresponding
        feature from the application layer according to predefined intra constraints and then change the
        selected equipment in infrastructure layer.
end
```

5. CASE STUDY AND EVALUATION

To demonstrate the feasibility of our approach, we performed a case study using the presented FM. For this purpose, in Figure 3 we provide two-layer FM for an E-Shop website previously depicted in a simple model in Figure 2. Within the case study, we are particularly interested in answering two following research question:

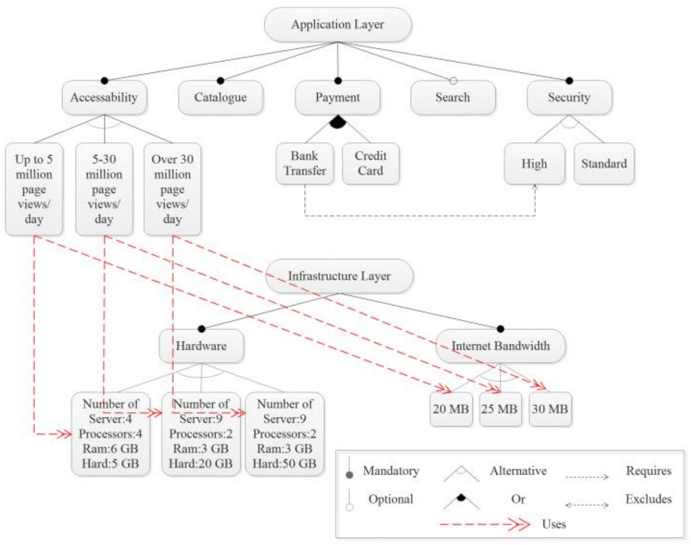
Application Layer
Accessability
Catalogue
Payment
Search
Security
Up to 5 million page views/ day
5-30 million page views/ day
Over 30 million page views/ day
Bank Transfer
Credit Card
High
Standard
Infrastructure Layer
Hardware
Internet Bandwidth
Number of Server:4 Processors:4 Ram:6 GB Hard:5 GB
Number of Server:9 Processors:2 Ram:3 GB Hard:20 GB
Number of Server:9 Processors:2 Ram:3 GB Hard:50 GB
20 MB
25 MB
30 MB
Mandatory
Optional
Alternative
Or
Requires
Excludes
Uses

Figure 3. Proposed Two-layer FM for E-Shop website

5.1. RQ1 (Effectiveness): Is the method effective for FM configuration?

The main aim of RQ1 is to determine whether our method can generate reliable results for application engineers, and also, which level of automation is supported by it.

In our proposed approach, the application engineers' tasks, based on stakeholders' requirements, are limited to: i) specifying the functional and non-functional features in the application layer of FM, ii) selecting the required infrastructure (or configuration that can be provided) in the infrastructure layer of FM. The configuration tool can automatically check whether the predefined inner and intra constraints are satisfied by one of the existing method, for example SAT solver [18] or FMVA [19] and notify the application engineers about the conflicts found between non-functional requirements and the selected infrastructure.

As a conclusion, we are able to answer RQ1 positively. Because we can conclude that: i) the final result of our approach is correct, and ii) an automatic solution for FM configuration can be generated where the stakeholders need to perform the minimum number of manual tasks.

5.2. RQ2 (Scalability): Can the method configure FMs in a reasonable time, based on functional and non-functional requirements?

The purpose of RQ2 is to evaluate whether our proposed method can be used to generate an FM configuration in a reasonable length of time when dealing with a large number of feature conditions.

We can see that accessibility and security are two important NFRs in this FM. When we initially design the FM of E-Shop SPL we have no information about the context of the final customized website, so we cannot include the maximum number of visitors in the FM. Therefore, we prefer to add accessibility features to the FM. The stakeholders can subsequently use these accessibility features to choose their prediction about the number of visitors per day.

If the stakeholders' prediction is incorrect, it might lead to service failure at peak times due to lack of dedicated network bandwidth or incompatible servers' hardware configuration.

Therefore, in the application requirement engineering phase it is necessary that the stakeholders have the complete knowledge about the infrastructure needed for implementation of their functional and non-functional requirements. Our proposed method, two-layer FM, provides the possibility for stakeholders to have the complete knowledge about the features and their relations in the application and infrastructure levels of their desired products at a glance.

This, for example, means that if, based on a predefined set of intra constraints, there is any conflict between the selected features in application layer of FM from one side and the hardware and network configuration in infrastructure layer on the other side, the configuration tool based on our method could detect this conflict, display an error message and request the stakeholders to undertake one of the following actions: i) change the desired features in application layer of FM, or ii) provide the required hardware and network configuration according to the infrastructure layer and then change the selected hardware and network features in the FM accordingly. In this way the intra constraints would be satisfied and the implementation of a customized product for stakeholders would be possible. Therefore, we are able to answer RQ2 positively.

Also for an instance, we can apply our proposed FMC algorithm to the FM configuration of the E-Shop website as shown in Figure 4.

1) Stakeholders select their desired non-functional features from application layer. For instance, “3-50 million page views per day” for accessibility, “Catalogue”, “Bank Transfer” and “Credit Card” for payment, and finally, “Standard Security”.
2) Stakeholders select implementation equipment from infrastructure layer comprising 4 servers each equipped with 4 processors, 6GB of RAM and a 5GB hard disk.
3) Check the satisfaction of inner constraints in both application and infrastructure layers by FMVA method: There are no conflicts between inner constraints. Therefore, step (3-1) in the algorithm is not taken.
4) Check the satisfaction of intra constraints between application and infrastructure layers by FMVA method. Result: A conflict is detected because the desired accessibility of stakeholders cannot be provided through the selected hardware configuration. There are two solutions for resolving this conflict:

a. Stakeholders can change the desired level of accessibility to “Up to 5-million-page view/day” in application layer. This accessibility feature is satisfied by the implementation equipment previously chosen in step 2.
b. Stakeholders provide 9 servers each equipped with 2 processors, 3GB of RAM and a 20GB hard disk. Based on the solution adopted, the selected features and equipment should be changed and the final customized FM will be ready.

As we can see the proposed approach, “Two-Layer FM” is a simple and practical solution for FM customization based on stakeholders’ requirements and available infrastructure.

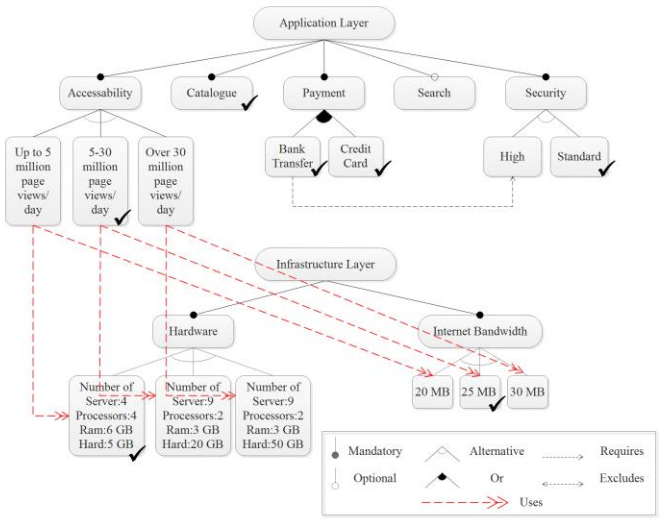
Application Layer
Accessability
Catalogue
Payment
Search
Security
Up to 5 million page views/ day
5-30 million page views/ day
Over 30 million page views/ day
Bank Transfer
Credit Card
High
Standard
Infrastructure Layer
Hardware
Internet Bandwidth
Number of Server:4 Processors:4 Ram:6 GB Hard:5 GB
Number of Server:9 Processors:2 Ram:3 GB Hard:20 GB
Number of Server:9 Processors:2 Ram:3 GB Hard:50 GB
20 MB
25 MB
30 MB
Mandatory
Optional
Alternative
Or
Requires
Excludes
Uses

Figure 4. Sample selected features by stakeholders in E-Shop FM

6. RELATED WORK

This section presents a systematic comparison between the main contribution of our work and the previous contributions in this area. To achieve this, we need to define a set of criteria that should be supported by any FM configuration approach. We have adopted the criteria set defined in [13] and modified it for our proposed approach. We do not claim that this criteria set is perfect, but it provides the necessary aspects to compare our work with others’. These criteria include: 1) Managing NFRs, 2) Optimization, 3) Ensuring FM constraints, 4) Automating configuration process, 5) Providing tooling support, 6) Time efficiency, and finally, 7) Supporting the definition of the infrastructure needed for implementation of the desired FM of stakeholders.

6.1. Feature model configuration approaches

The first significant contribution is by Czarnecki et al. [12] who introduced Staged configuration. They described a stepwise specialization of feature models where the configuration choices made in each stage are defined by separate feature models. This approach is motivated by the characteristic of a realistic development process, where different stakeholders make configuration choices in different stages. In this method the constraints between the features in the FM are not significant and automatic configuration is not considered. This method could be implemented by a configuration tool but it does not affect the time required to execute the configuration management process.

Benavides et al. in [11] presented how an FM (with or without considering cardinalities) can be translated into a Constraint Satisfaction Problem (CSP). In that way, it is possible to use off–the–shelf constraint satisfaction solvers to automatically accomplish several tasks such as calculating the number of possible configurations and detecting possible conflicts.

White et al. [9] introduced a Filtered Cartesian Flattening (FCF) method to select optimal feature sets according to resource constraints. In their approach, the feature selection problem is mapped to a multi-dimensional, multi-choice knapsack problem (MMKP). By applying existing MMKP approximation algorithms, they provided partially optimal feature configurations in polynomial time.

Siegmund et al. [20] proposed a technique for showing non-functional properties in FM and applied CSP to find optimal configuration based on user defined objective functions. In their technique there are some preprocessing steps to reduce the search space for optimal configuration.

White et al. [21] also formalized stage configuration and proposed a Multi-Step Software Configuration probLEm solver (MUSCLE) that provides a formal model for multi-step configuration. They considered non-functional properties such as cost constraints between two configurations and formalized them as CSP constraints. Their approach is only applicable for multi-stage configuration and focuses on creating new configurations from existing product configurations.

Mendonca et al. [22] introduced a translation of basic feature models based on propositional logic and used Binary Decision Diagrams (BDD) as the reasoning system. Their approach focuses on validating feature models and does not offer a facility for automated configuration. Their solution can be used in a multi-stage configuration process for validation of the results of every specialization in one FM (called interactive configuration). An interactive configuration only checks the structural constraints of FMs and does not consider preferences and non-functional requirements. A tool was implemented to support software developers in validation.

Gue et al. in [23] addressed the challenge of optimizing feature model configuration and covered this problem with an approach named GAFES which employs Genetic Algorithms to optimize feature selection. Machado et al. in [24] introduced SPLConfig as a tool that supports automatic product configuration in SPLs. The main goal of this tool is to derive an optimized features set that satisfies the customer requirements. The main contribution of their work is to achieve the balance between cost and customer satisfaction while also taking into account the available budget of customer. The main shortcoming of this tool is that it could not support non-functional features as constraints.

Batory in [25] defined a particular tool chain for product specification. The chain starts with a tool that uses a feature model configuration to specify a product. The model is maintained in a Logic-Truth Maintenance System (LTMS) and uses a propositional satisfiability (SAT) solver to prevent inconsistent specifications. The feature-based specification can be mapped onto a grammar from which various techniques can be used to produce products.

Sultana et al. in [13] employed the HTN planning process [26] for Artificial Intelligence (AI) planning and described a configuration process based on this method. They also proposed an optimal configuration framework that supports stakeholders' constraints over non-functional features.

6.2. Comparing the approaches

Table 1 summarizes the comparison between the previous approaches based on the criteria identified in section 6. As can be seen, none of the previous approaches (except ours) cover all the criteria. Below, we describe each criterion in detail.

Table 1. Comparative analysis of related works: "✓": criterion met, "-" criterion not met

Approach	Criteria						
	NFR	Optimization	Constraint	Automation	Tool Support	Time Efficiency	Infrastructure Support
Staged Method [21]	-	-	-	-	✓	-	-
CSP [11]	-	✓	✓	✓	✓	-	-
FCF [9]	-	✓	✓	✓	-	-	-
SPL Conqueror [20]	✓	✓	✓	✓	✓	-	-
MUSCLE [14]	-	✓	✓	✓	-	-	-
BDD [22]	-	-	-	✓	✓	-	-
GAFES [23]	-	✓	✓	✓	✓	-	-
SPLConfig [24]	-	-	-	✓	✓	-	-
LTMS Based Tool [25]	-	✓	✓	✓	✓	-	-
Sultana Framework [13]	✓	✓	✓	✓	✓	-	-
Our Approach	✓	✓	✓	✓	✓	✓	✓

Modeling NFRs: The modelling of the functional features is a default ability of any FM. Therefore, we focus on NFRs which are not supported by all approaches to FM configuration. Among the previous works, only SPL Conqueror [20] and Sultana Framework [13] provided a solution for modelling NFRs. Also our method supports NFRs as explained earlier.

Optimization and Time efficiency: Generating optimal FM configurations based on stakeholders' constraints is a difficult task. Almost all the previous approaches tackled the optimization problem except staged configuration [21] and BDD [22] because their main focus had been on stakeholders' satisfaction instead of performance efficiency or optimization. On the other hand, the CSP-based approaches [11, 13, 20, 21] provide optimized solutions but they require high computation time. Our approach provides optimization and also decreases the time required for FM configuration by making the stakeholders' tasks clear to them and also preventing rework of tasks. The latter is achieved by the stakeholders being notified at an early stage about any inconsistencies between their NFRs (in the application layer) and the available infrastructure (in the infrastructure layer).

Considering stakeholders' constraints: An FM may impose certain constraints between its features. These constraints need to be considered by the FM configuration methods and only a configuration satisfying the constraints must be produced. Table 1 shows that only Staged method [21], BBD [22] and SPLConfig [24] do not address constraints. Our proposed method allows the definition of constrains between features in the application layer as well as between features in the application and infrastructure layers of the FM. This covers the requirements of any SPL product implementation.

Tooling support and automation: Almost all the methods in Table 1, except FCF [9], were implemented or can be implemented as an FM configuration tool but none could support complete automation. Our method and Sultana et al. framework [13] has the ability to show all levels of FM configuration in the same view to stakeholders (Figure 3). Other tools only provide basic views of FM configuration to stakeholders.

Infrastructure support: We can state with certainty that none of the previous methods can support the infrastructure configuration of a product and also guarantee the implementation of all functional and non-functional requirements of the stakeholders. In all previous methods we cannot find any mention of the infrastructure needed for product implementation, they only cover the features in the application layer required by the stakeholders.

As we can see in Table 1, our prosoposed approach can cover all the defined critera. The Most important benefit of our approach is the **ease of FM configuration process and time efficiency**. The reason for this claim is that the process of FM configuration can be finilized in one-step based on available needed infrastructure .In all of the previous methods, stakeholders selected Features in first step and in the next step, the required infrastructure will be considered, which will reduce the time efficiency, the quality and the feasibility of implementation of the FM. In contrast, in our approach all the process for selction Features and decision about needed infrastructure for implemetion can be done in only one-step.

7. CONCLUSIONS

In this paper, we discussed an open research question in configuration management of SPLs: How can we guarantee the implementation of the desired functional and especially non-functional features that may need special hardware resources (e.g. processors, memory, hard disk, networking equipment, etc.). To answer this question, we proposed: i) a new "Two-layer" model comprising the application and infrastructure layers, ii) new "inner" and "intra' constraint types for feature modelling, and iii) a FM configuration algorithm describing the steps involved in feature selection leading to the final customized FM. These constitute a complete package to tackle the FM configuration issue in SPLE. Also, we evaluated our approach using a case study in the SPL of a sample E-Shop website. This was followed by a systematic comparison of our approach with previous related works based on a set of criteria. The results show that our approach could help the stakeholders to have complete knowledge about the application and infrastructure levels of their desired products at a glance and choose the features in the application layer according to the availability of the hardware resources in the infrastructure layer. In conclusion, the proposed method can be evaluated appropriately and used in any CM tools for the SPLs. Furthermore, our approach prevents the inclusion of non-functional requests from stakeholders that cannot be implemented with the hardware resources provided in the infrastructure layer. As yet, our approach is not implemented in any configuration management tool, so in future we intend to implement this new approach in the context of a new or existing open source configuration management tools for SPL.

REFERENCES

[1] K. Pohl, *et al.*, “A Framework for Software Product Line Engineering,” Springer, pp. 3-15, 2005.

[2] P. Clements and L. Northrop, “Software Product Lines: Practices and Patterns,” Addison-Wesley, 2001.

[3] Sommerville and P. Sawyer, “Viewpoints: Principles, Problems and a Practical Approach to Requirements Engineering,” *Annals Software Engineering*, vol. 3, pp. 101-130, 1997.

[4] J. Bosch, “Design and Use Software Architectures: Adopting and Evolving a Product-Line Approach,” Addison-Wesley, 2000.

[5] V. Alvesa, *et al.*, “Requirements Engineering For Software Product Lines: A Systematic Literature Review,” *International Journal of Information and Software Technology,* vol. 52. pp. 806-820, 2010.

[6] I. F. D. Silva, *et al.*, “Software Product Line Scoping and Requirements Engineering In a Small and Medium-Sized Enterprise: An Industrial Case Study,” *The Journal of Systems and Software*, vol. 88, pp. 189-206, 2013.

[7] S. Besrour and I. Ghani, “Measuring Security in Requirements Engineering,” *International Journal of Informatics and Communication Technology*, vol/issue: 1(2), pp. 72-81, 2012.

[8] D. Benavides, *et al.*, “Automated Reasoning on Feature Models,” *Proceeding of 17th International Conference on Advanced Information Systems Engineering*, pp. 491-503, 2005.

[9] J. White, *et al.*, “Selecting Highly Optimal Architectural Feature Sets With Filtered Cartesian Flattening,” *Journal Systems & Software*, vol. 82, pp. 1268-1284, 2009.

[10] D. Mairiza, *et al.*, “An Investigation into the Notion of Non-Functional Requirements,” *Proceeding of ACM Symposium on Applied Computing*, pp. 311-317, 2010.

[11] D. Benavides, *et al.*, “Using Java CSP Solvers in the Automated Analyses of Feature Models,” *Generative and Transformational Techniques in Software Engineering*, pp. 399-408, 2006.

[12] K. Czarnecki, *et al.*, “Staged Configuration Using Feature Models,” *Proceeding of Software Product Lines conference*, pp. 162-164, 2004.

[13] S. Sultana, *et al.*, “Automated Planning for Feature Model Configuration based on Functional and Non-Functional Requirements,” *Proceeding of 16th International Software Product Line Conference*, vol. 1, pp. 56-65, 2012.

[14] M. Reiser and M. Weber, “Multi-Level Feature Trees: A Pragmatic Approach to Managing Highly Complex Product Families,” *International Journal of Requirements Engineering*, vol. 12, pp. 57-75, 2007.

[15] K. C. Kang, *et al.*, “Feature-Oriented Domain Analysis (FODA) Feasibility Study,” *Technical Report CMU/SEI-90-TR-021, SEI, Carnegie Mellon University*, 1990.

[16] G. Kotonya and I. Sommerville, “Requirements Engineering Processes and Techniques,” John Wiley & Sons, 1998.

[17] M. Norian, *et al.*, “Non-functional Properties in Software Product Lines: A Taxonomy for classification,” *Proceeding of 24th International Conference on Software Engineering and Knowledge Engineering*, 2012.

[18] N. Eén and N. Sörensson, “An Extensible SAT Solver,” *Proceeding of 6th International Conference on Theory and Applications of Satisfiability Testin, LNCS 2919*, pp. 502-518, 2003.

[19] E. D. Farahani and J. Habibi, “Feature Model Constraints Control in Stage Configuration of Software Product Lines,” *International Journal of Software Engineering and Its Application,* vol. 11, pp. 1-12, 2017.

[20] N. Siegmund, *et al.*, “SPL Conqueror: Toward Optimization of Non-Functional Properties in Software Product Lines,” *Software Quality Journal*, 2011.

[21] J. White, *et al.*, “Automated Reasoning for Multi-Step Feature Model Configuration Problems,” *Proceedings of the 13th International Software Product Line Conference*, pp. 11-20, 2009.

[22] M. Mendonça, *et al.*, “S.P.L.O.T.: Software Product Lines Online Tools,” *Proceeding of 24th Conference on Object-Oriented Programming Systems, Languages and Applications (OOPSLA)*, pp. 761-762, 2009.

[23] J. Guoa, *et al.*, “A Genetic Algorithm for Optimized Feature Selection with Resource Constraints in Software Product Lines,” *Journal of Systems and Software*, vol. 84, pp. 2208-2221, 2011.

[24] L. Machado, *et al.*, “Splconfig: Product configuration in software product line,” *Brazilian Congress on Software (CBSoft), Tools Session*, pp. 1-8, 2014.

[25] D. Batory, “Feature Models Grammars, and Propositional Formulas,” *Proceeding of the 9th International Software Product Line Conference (SPLC 2005).Lecture Notes in Computer Science*, vol. 3714, 2005.

[26] S. Sohrabi, *et al.*, “HTN planning with preferences,” *Proceeding of 21st International Joint Conference Artificial Intelligence,* pp. 1790-1797, 2009.

BIOGRAPHIES OF AUTHORS

Elham Darmanaki Farahani received accordingly her bachelor and Master of Software Engineering at Years of 2002 and 2004 from Azad University, Central and South Branch in Tehran. She is currently PhD candidate in Software Engineering PhD at the Sharif University of Technology. She has the papers published in national and international conferences and Journals. Her research interests include software engineering, software product line and Configuration Management. Her email address is: efarahani@ce.sharif.edu.

Jafar Habibi received accordingly her bachelor in Computer Engineering and Master of Industrial Engineering at Years of 1980 and 1388 from School of Computer and Tarbiat Modarres University and received PhD in computer science in 1999 from the University of Manchester, UK. He is currently a member of the faculty the department of Computer Engineering Sharif University of Technology and director of the Computer Society of Iran. His research Fields in computer engineering are Performance Evaluation in Computer systems, software engineering, peer to peer networks, Social networks and data mining. His email address is: jhabibi@sharif.edu