# Quality-centric Feature Model Configuration using Goal Models

Mahdi Noorian
University of New Brunswick
Fredericton,Canada
m.noorian@unb.ca

Ebrahim Bagheri
Ryerson University
Toronto, Canada
bagheri@ryerson.ca

Weichang Du
University of New Brunswick
Fredericton,Canada
wdu@unb.ca

## ABSTRACT

In software product line engineering, a feature model represents the possible configuration space and can be customized based on the stakeholders' needs. Considering the complexity of feature models in addition to the diversity of the stakeholders' expectations, the configuration process is viewed as a complex optimization problem. In this paper, we propose a holistic approach for the configuration process that seeks to satisfy the stakeholders' requirements as well as the feature models' structural and integrity constraints. Here, we model stakeholders' functional and non-functional needs and their preferences using requirement engineering goal models. We formalize the structure of the feature model, the stakeholders' objectives, and their preferences in the form of an integer linear program to automatically perform feature selection.

## CCS Concepts

•**Software and its engineering → Rapid application development;**

## Keywords

Feature Model; Goal Model; Configuration Process

## 1. INTRODUCTION

The Software Product Line Engineering (SPLE) provides the means for capturing the commonalities of all possible products of a given domain and also addresses variability by covering a comprehensive set of dissimilarities between the products [10]. There are basically two lifecycles in software product lines, namely *domain engineering* and *application engineering*. Domain engineering consists of activities to understand and analyze the commonalities and variabilities of applications in the target domain. In this context, *Feature models* are widely used as variability modeling techniques to depict the shared and variable functional characteristics of a set of similar systems. In essence, each functional characteristic is referred to as a *feature* and the set of all features of a domain form the *feature space*. On the other hand, the application engineering phase captures stakeholders' requirements and derives an appropriate application instance by choosing the right elements from the domain model through a *configuration process* [5].

Software product line configuration is a complicated process, which can be seen as multi-criteria optimization problem [2]. The software product line research community has developed interesting mechanisms and tools for configuring product line models such as feature models [3, 5, 13]. The basic assumption of these methods is that the set of initial desirable features is already known to the stakeholders or at least to the software product designers. However in reality regardless of the configuration process itself, the selection of the initial set of desirable features is both important and very difficult to do for the stakeholders and product designers. The selection of these features depends on the restrictions placed by and objectives of the stakeholders and the requirements of the target deployment environment. Therefore, an important research challenge is to develop methods or processes that can help identify the set of desirable features to fulfill the stakeholders' needs.

In this paper, we propose a framework for feature model configuration. Our main focus in this paper is to facilitate the application engineering process by effectively communicating with the stakeholders and understanding their needs and preferences. This research is built on the QcFM method [9], which integrates the target feature model with its related domain level goal model. In fact, we use domain level goal models as a reference model to communicate with the stakeholders and understand their needs by identifying desirable hard goals, as functional requirements, and softgoals as non-functional requirements. We then adopt AHP and BST ranking methods to prioritize stakeholders' requirements. Finally, we propose an optimization model based on the Integer Linear Programming (ILP) method to automatically perform feature selection.

## 2. AN ILLUSTRATIVE EXAMPLE

In this section, we present an illustrative example. In the remaining parts of the paper, we refer to this case-study as a running example. As it is represented in Figure 1, the example consists of two parts: the feature space, which is represented using feature model formalism [10], and the intention space, which is represented using goal model formalism [14]. The right side in Figure 1 depicts a feature model. It consists of three main features: "Connectivity", "Messaging", and "OS". The left side in Figure 1 shows a sample goal model representing the high level goals for the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

*SAC 2016, April 04-08, 2016, Pisa, Italy*
© 2016 ACM. ISBN 978-1-4503-3739-7/16/04…$15.00
DOI: http://dx.doi.org/10.1145/2851613.2851959

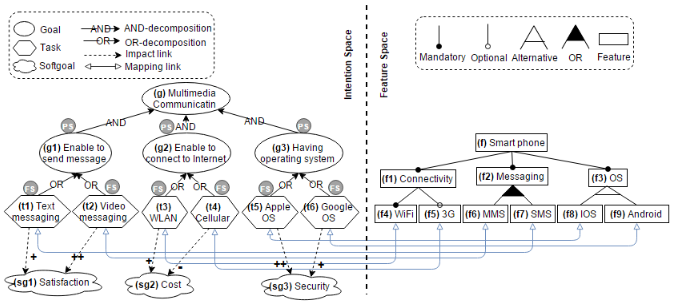

**Figure 1: Integrated feature and goal model for smart phone domain.**

mobile domain. It can be seen that there is three high level goals: "Enable to send message", "Enable to connect to Internet", and "Having operating system". The goal model shows how these goals can be operationalized with the related tasks. Moreover, the tasks are related with the softgoals, e.g., "WLAN" task has positive impact on satisfaction of "Cost" softgoal. As shown in Figure 1, the feature space is connected to intention space via mapping links using the method that is developed in [9]. For details description of feature and goal models and the integration mechanism, the interested readers can refer to [9].

## 3. APPROACH OVERVIEW

The proposed feature model configuration approach consists of three major steps:

**Step 1.** The first step involves the identification of the stakeholders' requirements and preferences. A domain level goal model represents functional and non-functional requirements in the form of goals and softgoals. Using goal model and by aid of two well accepted requirement prioritization methods, the stakeholders requirements and preferences is captured.

**Step 2.** Once the requirements and preferences have been identified, the next step is to develop an optimization model. The purpose of this step is to develop an Integer Linear Programming (ILP) optimization model that can be used to automatically select a set of features from the feature space according to the stakeholders' defined requirements.

**Step 3.** In the third step, the developed optimization model is initially fed into an ILP optimization engine in order to automatically select a set of features from the feature space. Subsequently, based on the selected features, the feature model configuration is developed.

## 4. USER REQUIREMENT ELICITATION USING GOAL MODELS

In this work, we employ goal models as a reference model and capture and prioritize stakeholders' requirements. To this end, the stakeholders will be asked to select a set of goals and softgoals as their intended functional and non-functional requirements. Afterward, the selected requirements will go through a prioritization procedure in order to rank the captured requirements. In fact, goals and softgoals prioritization enables stakeholders to focus on those requirements that are the most important to add value on prospective software product. The literature is replete with various approaches that address requirement prioritization issue. We benefit from two prominent approaches called BST based ranking method [7], which is employed to prioritize the stakeholders'

**Table 1: Softgoals prioritization using AHP method.**

a) relative importance.

| | $sg_1$ | $sg_2$ | $sg_3$ |
|---|---|---|---|
| $sg_1$ | 1 | 3 | 1/7 |
| $sg_2$ | 1/3 | 1 | 1/5 |
| $sg_3$ | 7 | 5 | 1 |

b) normalized matrix

| | $sg_1$ | $sg_2$ | $sg_3$ | $sum$ | $pValue$ |
|---|---|---|---|---|---|
| $sg_1$ | 2.97 | 6.7 | 0.88 | 10.55 | 0.16 |
| $sg_2$ | 2.06 | 2.99 | 0.45 | 5.5 | 0.08 |
| $sg_3$ | 15.65 | 31 | 2.98 | 49.63 | 0.76 |

functional requirements (goals), and AHP [12], used to rank the stakeholders' non-functional requirements (softgoals).

### 4.1 Non-functional Requirement Prioritization

In order to determine the priority of the non-functional requirements for the stakeholders, we use the AHP method. By using the AHP method, the stakeholders' all desired softgoals are considered and pair-wise comparison process is undertaken to produce a ranked list of softgoals. In fact, the priority value, $pValue$, shows the priority order of the softgoal in the feature selection process.

According to the running example, assume that the stakeholder select the three softgolas from the goal model, ($sg_1$) "Satisfaction", ($sg_2$) "Cost", and ($sg_3$) "Security", as her intended non-functional requirements. In order to prioritize them, the first step is to create a comparison matrix $M\,[3 \times 3]$ (Table 1 part (a)), whose entries show the relative importance of each pair of softgoals (the stakeholder will be asked to provide these values). For instance, $M\,[1, 2] = 3$, which shows the softgoal $sg_1$ is moderately more important that softgoal $sg_2$. In the second step, the $pValue$ needs to be calculated for each softgoals based on the provided relative importance values (Table 1 part (b)).

### 4.2 Functional Requirement Prioritization

In this work, we employed Binary Search Tree (BST) based ranking method [7] which is an efficient method for prioritizing large-scale items. The binary tree structure can be used to rank stakeholders selected goals in which each node of the tree represents a goal. According to the process described in [11], the ranking of goals and the normalized ranking value, $rValue$ will be assigned to each goal ($rValue = 1/1+$ (ranking value)) .

For example, assume that the stakeholder selects three goals ($g_1$) "Enable to send message", ($g_2$) "Enable to connect to Internet", and ($g_3$) "Having operating system" as her intended functional requirements. First she selects "Enable to connect to Internet" goal and put it as a root node. Then, she selects the "Enable to send message" goal and compares it with the root goal. Since, the "Enable to send message" goal is less important than "Enable to connect to Internet" goal it needs to be placed in the left side of the root goal in the ranking tree. Finally, the third goal "Having operating system", has the least importance among the two other goals, so it needs to be placed at the left side of the second goal in the ranking tree. Table 2 shows the ranking value for each goal and the related normalized $rValue$.

**Table 2: Goals prioritization using the BST method.**

| Goal | $g_1$ | $g_2$ | $g_3$ |
|---|---|---|---|
| BST ranking | 2 | 1 | 3 |
| $rValue$ | 0.33 | 0.5 | 0.25 |

## 5. DEVELOPING OPTIMIZATION MODEL

In this section, we describe how the problem of feature selection can be formulated as an Integer Linear Programming (ILP) [8]. The ILP model can be characterized by three constituents 1) a set of decision variables ($f$) with a domain $D = \{0, 1\}$; 2) a set of constraints ($C_{FM}$, $C_{GM}$, and

$C_{Mapping}$); and 3) objective function ($O$). According to ILP problem definition [8], the ILP optimization for feature selection can be formalized as:

$$Maximize \sum_{i=1}^{|F|} U(f_i) \times f_i \quad (1)$$

$$Subject\ to\ C_{FM}\ and\ C_{GM}\ and\ C_{Mapping} \quad (2)$$

The output of ILP problem is a set of features that maximize the objective function which is the summation of decision variables ($f_i$) multiplied by $U(f_i)$, where each decision variable $f_i$ shows the inclusion or exclusion of the corresponding feature from feature selection set and the $U(f_i)$ indicates the utility of a feature by considering its satisfaction toward stakeholders' selected goals and softgoals as their intended requirements.

## 5.1 The Objective Function

The utility value $U(f)$ is the key part of objective function and shows to what extent a feature $f$ can satisfy the stakeholders' functional, non-functional requirements and preferences. Using the integrated feature and goal models and through the mapping links that are established between the models, the utility of each feature can be determined by tracing the feature from feature space to intention space and by evaluating the satisfaction degree of its associated task over the stakeholders selected goals $G$ (as functional requirements) and softgoals $SG$ (as non-functional requirements). Here, we define $U(f)$ as:

$$U(f) = Impact_{f \to G} \bigoplus Impact_{f \to SG} \quad (3)$$

Assume that the stakeholder selects $\{g_1, g_2, g_3\}$ as her functional requirements. In addition, she selects $\{sg_1, sg_2, sg_3\}$ as her intended non-functional requirements. In order to develop a objective function, we need to first identify what are the features in feature space that can satisfy these elements; second we need to calculate the utility of each features in feature space that can satisfy the captured goal and softgoals. For this example, the objective function can be developed as: $Maximum\ U(f_4)f_4 + U(f_5)f_5 + U(f_6)f_6 + U(f_7)f_7 + U(f_8)f_8 + U(f_9)f_9$ . For instance, for calculating $U(f_4)$, we need to trace feature ($f_4$) "WiFi" from feature space to intention space and see to what extent it impacts on the related goal, ($g_2$) "Enable to connect to Internet", and softgoal, ($sg_2$) "Cost". The utility value for feature $f_4$ can be calculated as: $U(f_4) = Impact_{f_4 \to g_2} \bigoplus Impact_{f_4 \to sg_2}$.

### 5.1.1 Calculate Feature Impact over Goals

In order to compute impact of a feature $f$ over goal $g$, $Impact_{f \to g}$, we adopt a diagrammatic reasoning approach named *forward label propagation* [6]. This algorithm starts from lower level goals and works its way to the top goals; therefore, it is able to estimate to what extent high level goals are satisfied given the satisfiability of the lower level goals and tasks. The interpretation of this algorithm for our context would be to select a specific feature and see how it relates to the stakeholders selected goals and to what extent it is able to satisfy them. Using this algorithm, the utility degree of feature based on its impact over the goal element can be calculated as:

$$Impact_{f \to G} = rValue(G) \times \prod_{\forall\, g\, in\, P_f^G} satValue(g) \quad (4)$$

Where, $satValue$ shows to what extent each child goal can satisfy its parent goal. These values need to be calculated and addressed in the goal model by requirement analysts during the goal model design and development. The $rValue$ shows what is the priority of goal that the feature can satisfy. In order to transform qualitative satisfiability and deniability labels (FS, PS, FD, PD) to qualitative values, we adopt the conversion schema from [6].

**Table 3: Conversion schema for satisfiability and deniability labels.**

| Qualitative satisfaction value | FD | PD | PS | FS |
|---|---|---|---|---|
| Quantitative $satValue$ | -1 | -0.5 | 0.5 | 1 |

For example, using Equation 4, the impact of feature $f_4$ "WiFi" over goal $g_2$ can be calculated: $Impact_{f_4 \to g_2} = rValue(g_2) \times satValue(t_3) = 0.5 \times 1 = 0.5$ . In another example, the impact of feature $f_7$ over goal $g_1$ can be calculated as: $Impact_{f_7 \to g_1} = rValue(g_1) \times satValue(t_1) = 0.33 \times 1 = 0.33$.

### 5.1.2 Calculate Feature Impact over Softgoals

Using the integrated feature and goal model, the feature impacts over non-functional properties can be identified through the impact links. Based on the impact values and the prioritization value that we obtained in step 1, the impact of the features over the stakeholders selected non-functional requirements (softgolas) can be calculated as follows:

$$Impact_{f \to SG} = \sum_{\forall sg \in SG} impDegree(sg) \times pValue(sg) \quad (5)$$

Where, $impDegree$ shows to what extent each feature can satisfy a softgoal (non-functional property) in the goal model. The $pValue$ shows the priority values of softgoal that is calculated using the AHP method. In order to quantitatively calculates the impact degree, we adopt the conversion schema [4] to transform the qualitative satisfaction labels.

**Table 4: Conversion schema for impact degree.**

| Qualitative impact value | - - | - | ? | + | ++ |
|---|---|---|---|---|---|
| Quantitative $impDegree$ | -1 | -0.5 | 0 | 0.5 | 1 |

For example, feature ($f_4$) "WiFi" is related to softgoal ($sg_2$) "Cost". Using Equation 5, the impact of feature $f_4$ over softgoal $sg_2$ can be computed as: $Impact_{f_4 \to sg_2} = impDegree(sg_2) \times pValue(sg_2) = -0.5 \times 0.08 = -0.04$. Hence, the objective function based on the calculated utilities can be developed as: $Maximum\ (0.46)f_4 + (0.54)f_5 + (0.49)f_6 + (0.41)f_7 + (1.01)f_8 + (0.63)f_9$.

## 5.2 Linear Constraints Derivation

In this section, we preset the transformation rules for feature and goal models. The linear constraints in the optimization model ensure that all these constraints will be satisfied in the optimized feature selection process. The transformation rules are represented in Table 5. The first row shows the feature model relations and the second row presents the linear constraint mapping rules for each individual relation. As indicated before, each feature in the feature model corresponds to a binary decision variable with a domain D∈{0,1}, where if the feature is selected for the final product the values is 1 and 0 is otherwise. Based on this, the transformation rules for feature model structural and integrity constraints are defined. For instance, assume $f_p$ is a parent feature and $f_c$ its child with a mandatory relation. The corresponding mapping rule is $f_c = f_p$. Because, in mandatory relation the presence of parent feature ($f_p$=1) enforce the presence of child feature ($f_c$=1). Following the running example, consider feature $f_3$ "OS" as the parent feature which has Alternative relation with its child features $f_8$ (IOS) and $f_9$

**Table 5: Transformation rules for feature models.**

| Relation | Mandatory | Optional | OR | Alternative | Include | Exclude |
|---|---|---|---|---|---|---|
| Linear constraint | $f_c = f_p$ | $f_c \leq f_p$ | $\forall i \in [1..n] : f_{c_i} \leq f_p$ $\sum_{i=1}^{n} f_{c_i} \geq f_p$ | $\sum_{i=1}^{n} f_{c_i} = f_p$ | $f_A \leq f_B$ | $f_A + f_B \leq 1$ |

"Android". The linear constraints for this relation can be represented as $f_8 + f_9 = f_3$. In another example, consider feature $f_1$ "Connectivity" as the parent feature which has Optional relation with its child features $f_5$ "3G". The linear constraints for this relation can be represented as: $f_5 \leq f_1$. On the other hand, the goal model tree structure and relation between goals and tasks are represented using AND and OR decomposition links. The mapping rules to translate goal model into ILP constraints are represented in Table 6. In order to represent the mapping links between feature and task elements, we need to include the mapping constraints into the linear constraint set. Assume task $t$ is connected to feature $f$ via mapping link. The equivalent linear constraint for this relation can be expressed as $t = f$, which indicates if task $t$ is included then feature $f$ must be included.

**Table 6: Transformation rules for goal models and mapping link.**

| AND | OR | Mapping |
|---|---|---|
| $\sum_{i=1}^{n} g_{c_i} = n \times g_p$ | $\forall i \in [1..n] : g_{c_i} \leq g_p$ $\sum_{i=1}^{n} g_{c_i} \geq g_p$ | $t = f$ |

## 6. FEATURE MODEL CONFIGURATION

The final step of our configuration framework is devoted to the feature selection. The main aim of the feature selection is to identify the optimum set of features in the feature space that can maximize the stakeholder satisfaction over their selected goals and softgoals. In the second step, the ILP optimization model is developed. Here, we utilize the IBM Ceplex [1] as an ILP solver. We can run the developed optimization model in order to identify a set of optimized feature set. Following the running example, the optimum features that can maximize the objective function are: $< f_4, f_5, f_6, f_7, f_8 >$. After the optimum feature set is identified, we start to conduct feature model specialization. For this purpose, the specialization guidelines that are introduced in [5] can be adopted to limit the configuration space by selecting the desirable features (features that are selected as optimum feature sets) and filtering the conflicting and unwanted features. The outcome of this process is a specialized feature model. Based on the running example, the features that are included in specialized feature model are: $< f, f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8 >$, where the feature $f_9$ is excluded from the feature model.

## 7. RELATED WORK

In the context of software product line application engineering various techniques have been proposed in order to handle the complexity of configuration process. In [5] Czarnicki et al. introduced an interesting approach named staged configuration process. They divided the configuration process into several steps. In each step, they developed a specialized feature model by removing the unwanted features from the variability points. Following that, Benavides et al. [3] proposed a method to perform automatic reasoning on SPL with respect to the extra functional features. For this purpose, the extended feature model is mapped onto Constraints Satisfaction Problems (CSP), and desired products is developed based on users' constraints and requirements. Finally, in [13] the authors introduced a framework to automatically perform feature selection while satisfying stakeholders' functional and non-functional concerns, preference, and constraints. In their proposed approach, the configuration problem is reduced to Hierarchical Task Network (HTN) planning problem and uses HTN-based planning solver to derive the optimal product.

## 8. CONCLUDING REMARKS

In this paper, we pinpoint the open research question in software product line application engineering, which is: how and what features should be selected for the target software product from the feature model, based on stakeholders' functional and non-functional requirements and preferences. We believe that our work provides the following benefits for the application engineering phase: 1) considering non-functional properties alongside of the functional properties in configuration process; 2) capturing stakeholders preferences by using easy to understand mechanism; 3) using standard requirement engineering method as a communication medium for understanding stakeholders' requirements ; and 4) providing a semi-automated approach for developing an optimal software product.

## 9. REFERENCES

[1] Ibm cplex, http://www.ilog.com/products/cplex/.

[2] M. Bashari, M. Noorian, and E. Bagheri. Product line stakeholder preference elicitation via decision processes. *International journal of knowledge and systems science*, 5(4):35–51, 2014.

[3] D. Benavides, P. Trinidad, and A. Ruiz-Cortés. Automated reasoning on feature models. In *Advanced information systems engineering*. Springer, 2005.

[4] L. Chung and J. do Prado Leite. On non-functional requirements in software engineering. *Conceptual modeling: foundations and applications*, 2009.

[5] K. Czarnecki, S. Helsen, and U. Eisenecker. Staged configuration through specialization and multilevel configuration of feature models. *Software process: improvement and practice*, 10(2):143–169, 2005.

[6] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani. Reasoning with goal models. *Conceptual modeling*, pages 167–181, 2003.

[7] J. Karlsson, C. Wohlin, and B. Regnell. An evaluation of methods for prioritizing software requirements. *Information and software technology*, 39:939 – 947, 98.

[8] Wolsey A. L. *Integer programming*. Wiley, 1998.

[9] M. Noorian, M. Asadi, E. Bagheri, and W. Du. Addressing non-functional properties in feature models: a goal-oriented approach. *IJSEKE*, 24(10):1439–1488, 2014.

[10] K. Pohl, G. Böckle, and F. Linden. Software product line engineering: Foundations, principles, and techniques. 2005.

[11] Z. Racheva, M. Daneva, and L. Buglione. Supporting the dynamic reprioritization of requirements in agile development of software products. In *IWSPM*, 2008.

[12] R.W. Saaty. The analytic hierarchy process what it is and how it is used. *Mathematical modelling*, 9, 1987.

[13] S. Soltani, M. Asadi, D. Gašević, M. Hatala, and E. Bagheri. Automated planning for feature model configuration based on functional and non-functional requirements. In *SPLC*, pages 56–65. ACM, 2012.

[14] E.S.K. Yu. Towards modelling and reasoning support for early-phase requirements engineering. In *RE*, 1997.