Diploma/Master Thesis Topics in the Field of Software Product Line Configuration

Supervisor: Juliana Alves Pereira
juliana.alves-pereira@ovgu.de

The growing need for developing larger, ever-more complex and expensive software applications demands better support for reusable software artifacts. Software product lines (SPL) have proved to be an efficient and effective strategy by exploiting large-scale reuse and dealing with many challenges of today’s software development. A SPL is a software architecture that can be customized and reused across different applications by varying software artifacts (called features), in order to satisfy the specific needs of a particular market segment [1][2]. The development lifecycle of a SPL encompasses of two main phases, namely domain engineering and application engineering. In domain engineering phase, the requirements of a particular market segment are understood and analyzed, by identifying the variabilities and the commonalities between a set of applications of that domain. The outcome of this phase is a comprehensive formal representation of the domain products through variability model, in order to establish the reuse platform. On the other hand, the application engineering phase is concerned with the effective reuse of software artifacts. However, a key challenge during application engineering is how efficiently reuse the artifacts developed during the domain engineering to derive a concrete product configuration from the variability model in order to optimize the stakeholders’ requirements. Product configuration is challenging for a product engineer due to the following reasons: (i) feature models tend to be inherently big and complex, with several types of variability relations and cross-tree constraints between the features; (ii) many factors including stakeholders’ constraints and preferences must be considered in the configuration process; and (iii) the number of possible configurations typically grows exponentially as a result of the high number of possible combination features. In this context, the goal of the following three thesis topics is identifying, classifying, assessing, and developing techniques to support the product configuration process in SPL.

Literature:


1. A Feature Recommendation System for Configuration Upgrade in Software Product Line

To start competitive in today’s marketplace, an industry must understand the unique and particular needs each customer in order to recommend a personalized products list. SPL allow product engineers personalizing configurations through the consecutive feature selections. When selecting features, engineers need to be able to understand the impact of decisions that are made gradually. However, they are often unable to understand the purpose of the target system and turn these objectives into priorities in order to follow choosing between competitive features in the product line. Moreover, the selection manual of individual features from shared software artifacts is often a time-consuming, error prone, tedious task, and expensive activity in industry. Therefore, in order to support the difficulties
during the product configuration process, recommendation systems have been increasingly adopted in the software industry. They are essential to evaluate the potentially overwhelming number of alternative items that an SPL may offer (an example is the Linux kernel product line, generating more than 12,000 different product variants) and assist the stakeholders find information and to make decisions. Therefore, the novelty of this research is to investigate two relevant software engineering topics, namely recommendation systems and product-line configuration.

In state-of-art, there are some techniques that use recommendation system to support product configuration in SPL [1][2][3][5][6][7]. However, these techniques focus just on guiding stakeholders understanding the non-functional properties of product lines and identifying their preferences during the configuration process. On the other hand, we do not know any approach that use recommendation techniques for suggestions of configuration upgrades. Taking into account that over time a customer will inevitably need to evolve their product (i.e., introduce new features) in order to satisfy your future requirements (such as, expansion of the business), this type of support is essential to guide business offering significant product upgrades to customers. Therefore, the goal of this thesis is to propose a feature recommendation system to support the problem of manual and individual feature selections during the product configuration upgrades. The system provides recommendations to users about which features are the best option to upgrade the product under certain conditions based on observed behavior of other customers by calculating similarities between styles, preferences, and configurations via classification, in order to predict whether a particular customer will like a particular feature and identify a set of features that will be of interest to a certain customer [4]. In this context, the extension of the FeatureIDE tool (see http://wwwiti.cs.uni-magdeburg.de/iti_db/research/featureide/) with the decision support technique, as well as its evaluation, is part of the work on the thesis.

Literature:


2. FeatureIDE Extension with Non-Functional Properties: An Approach Based in Metrics

Feature models are de facto standard variability modeling technique in SPL to document the common and variable features [6]. They provide a formal representation of how software artifacts can be reused, by means of dependencies among them and interrelationships among variation points, ensuring they are reused appropriately. A common format for a feature model is a feature-tree structure, where nodes illustrate the features, and the edges illustrate the interrelationships between parent and child features (e.g., show in Figure 1) [6]. In SPL, there are common features found on all products of the product line (known as mandatory features) and variable features that allow distinguishing between products in a product line (generally represented by optional or alternative features). Variable features define points of variation and their role is to permit the instantiation of different products by enabling or disabling specific features according stakeholders’ requirements. To represent stakeholders’ requirements on SPL, extended feature model notation with non-functional properties are used. Figure 1 illustrates smart-home extended feature model, representing non-functional properties, as well as functional properties and their relationships. For example, the feature *siren* is annotated with quantitative non-functional properties values (i.e., response time = 500 milliseconds, reliability = 98%, and estimated cost = $600.0) and qualitative non-functional properties values (i.e., high customer satisfaction and high security).

Recently, a large number of studies [1][2][3][5][9][10][11][12] have investigated the non-functional properties notation in product lines. However, in most of these studies, the impact of each feature on non-functional properties is estimated based on the knowledge of domain experts manually assigning values to each non-functional properties of the model. There are few techniques available that use metrics to automatically measure appropriate non-functional properties values. Therefore, the goal of the thesis would be to investigate and propose the use of metrics to measure the non-functional properties values categorized in the literature by Soares et al. [11]. Moreover, an extension of the FeatureIDE tool (see http://wwwiti.cs.uni-magdeburg.de/iti_db/research/featureide/) with support for extended feature model notation and automatic measurement of non-functional properties is also encouraged to be implemented. Finally, the experiments are planned to take place in a real environment in order to accurately estimate the efficiency of the proposed algorithms and effectiveness of the results.

Literature:


Figure 1. An extended feature model with non-functional properties for a smart-home product line (adapted from Cetina et al. [4]).


Having developed a feature model to a formal representation of the domain products (see Figure 1), a concrete product can be derived by the product engineer based on the dependencies and interrelationship among the reusable features (optional and alternative features), and requirements of the domain stakeholders (non-functional properties values). This is possible through of the selection or exclusion of variables features in an SPL. However, the challenge is that with hundreds or thousands of features, it is hard to analyze each possible configuration to find a set of optimal features that maximizes customer satisfaction (e.g., in Figure 1 smart-home platform can generate 14388 different product variants). Moreover, many factors must be considered in the configuration process, including stakeholders’ constraints and preferences, creating interdependency relation between non-functional properties. An interdependency relation means that to satisfy a non-functional properties value affects...
mutually the value of another non-functional properties (e.g., increasing the value of security has negative impact on the response time value). Therefore, multi-objective optimization arises to handle interdependency relation between non-functional properties in SPL. In these cases, the variability resolution works automatically in the background in order to balance the competing options and quickly customize a combination of features that fits the customer requirements.

There are few techniques in the literature to support product engineers during the multi-objective configuration process [1][2]. Moreover, empirical results show that to solve multi-objective optimization problems, exact and approximate algorithms still require substantial efforts to find the optimal configuration. Therefore, this thesis aims to conduct an evaluation from real and randomly generated product lines, in order to measure the efficiency and effectiveness of the proposed technique and make a comparison with the existent techniques. In addition, we encouraged the extension of a configuration language for product requirements management [1]. Finally, an extension of FeatureIDE tool with the proposed technique of decision support for automatic product configuration would be part of the work on the thesis.

Literature:


4. Other topics

Besides the presented topics above, I encourage students to propose their own topics in the field of software product line and related areas.