A Review of Feature Model Position in the Software Product Line and Its Extraction Methods

Saba Pedram *pedram.saba@gmail.com*
Department of Computer
Islamic Azad University, Science and Research Branch
Tehran, Tehran, Iran

Mehran Mohsenzadeh *mohsenzadeh@srbiau.ac.ir*
Department of Computer
Islamic Azad University, Science and Research Branch
Tehran, Tehran, Iran

Amir Azimi Alasti Ahrabi *amir.azimi.alasti@gmail.com*
Department of Computer
Islamic Azad University, Shabestar Branch
Tabriz, East Azerbaijan, Iran

Abstract

The software has become a modern asset and competitive product. The product line that has long been used in manufacturing and construction industries nowadays has attracted a lot of attention in software industry. Most importance of product line engineering approach is in cost and time issues involved in marketing. Feature model is one of the most important methods of documenting variability in product line that shows product features and their dependencies. Because of the magnitude and complexity of the product line, build and maintain feature models are complex and time-consuming work. In this article feature model importance and position in product line is discussed and feature model extraction methods are reviewed and compared.

Keywords: Software Product Line, Feature Model, Extraction Method Review.

1. INTRODUCTION

The software has become a modern asset and competitive product [1]. The product line that has long been used in manufacturing and construction industries nowadays has attracted a lot of attention in software industry. Most importance of product line engineering approach is in cost and time issues involved in marketing; but product line engineering is also supports other business purposes [2].

According to the definition provided by Carnegie Mellon University, Software product line includes a family of system that have a series of technical assets in common between all of them (core assets) and variability parts that were considered in order to meet customers' specific requirements. Software product line engineering adds a lot of values for developer companies. Reusability, short time product presentation and quality are all of the aspects that make software product line development as a cost-effective approach [2].

Primarily, feature models were introduced in 1990 in feature-oriented domain analysis methods. FODA method support reusability in functional and architectural levels. Features are logical units of behavior that is described by a set of functionality and quality requirements and represent an important value for the users [3]. In features oriented design and implementation, feature models are standard visual presentation. Feature diagram describe features and integrity constraints between them as a tree. A feature generally indicates the abstract of a domain. But feature is more than a name in domain modeling; its other properties must be considered as well.

Some of the potential characteristics that domain experts can collect for the features as followed:

- Describes the features and their corresponding requirements
- Relationship with other features, especially the hierarchy, order and grouping
- The estimated or measurement cost of an achievement of a feature
- Configuration knowledge, such as “enabled by default”
- Constraints, such as “include feature X and exclude feature Y”
- Relationship between potential features

Features do not easily compose [4]. All features cannot be composed and some features may require the presence of other features. So feature model describes the relationship between the features and valid feature selections [5].

Implementation of the variability in the features level has an important effect on conceptual integrity of the system. Features used for domain modeling, variability management, guidance of feature planning and as the basis for communication between stakeholders in the system or used as general guidelines for the system design [6].

In order to emphasize the complexity of the design position in the product line, the following table compares product line approach and single software development briefly.

Criteria	Product line	Single system
Production area	Family of products	Single system
Development approach	Tow life cycle	Single life cycle
Reusability	Strategic and technical reusability	Technical reusability
Practical purpose	Meet the needs of multiple customers in certain segments of the market	Meet the needs of a customer
New product time to market	Short	Long
Development costs per system in long term	Decreasing (stable)	Increasing
Required architecture type	Reference architecture	Single system architecture
Variability	Numerous and mandatory	Limit
Requirement analysis complexity	High	Depend on product type high/low

TABLE 1: Compares product line approach and single software development.

The task of constructing feature models can be very arduous for requirements engineers, especially if they are presented with heterogeneous and unstructured requirements documents (such as interview transcripts, business models and technical specifications) from which to derive the feature model. The task of identifying core and variant features can be particularly difficult in the case of extractive or reactive SPLE [7].

In this article, feature model extraction methods are discussed. In section 2, importance and necessity of feature model in product line is reviewed from different viewpoint. Section 3 describes the feature model extraction methods. And in Section 4 feature model extraction methods have been compared.

2. FEATURE MODEL IMPORTANCE AND POSITION IN SOFTWARE PRODUCT LINE

Software product line engineering is an affordable approach for development of product family. Product line engineering approach focuses on the creation of reusable infrastructure. The key of its success is in having the correct view of the scope of software product line, complete identification of common variables, products and interconnections between the features. Software product line domain is analyzed by feature oriented domain analysis methods and modeled by feature model.

Features are the first product line engineering concerns. Describing feature is difficult since on the one hand is stakeholders and the other hand is design and implementation concepts [4]. At first glance, it seems that feature model is visual; but creation of good feature model that covers most of commonalities and variability and make reusable asset development easily, is difficult [8].

If the scope of the problem is determined to be incomplete, interrelated features may be not implemented, or implement features may not use. Such problems cause unnecessary complexity and increase development and maintaining costs [9]. To avoid these problems, software product line domain is usually modeled by feature model. Such as other model-based approaches, product line engineering faced with large-scale models and thousands uncommon features [10]. Construction and maintenance of such large models are very complex.

3. FEATURE MODEL EXTRACTION METHOD

The early roots of modeling features can be found in the architectural community. Architects use feature models for the production of simple and abstract overview of the architecture. The UML class diagram or package is an example of this application. Feature model describe commonalities and variables in the form of mandatory, alternative and optional form, also it highlights constraints. A basic feature model is a ‘and-or’ graph with some constraints for identifying valid product line configuration [11].

Feature model extract in different ways, in the following the methods are described briefly.

Mathieu Acher [12] used tabular format product description in their proposed feature model extraction method. This process is parameterized through a dedicated language and high-level directives (e.g., products/features scoping).in this method, several tabular data file that is collection of products description from different perspectives documents, is used for feature model extraction. They describe a specific merging algorithm that first computes the feature hierarchy and then synthesizes the variability information using propositional logic techniques. The main contributions of this paper are extraction process parameterized by a dedicated language and an automated procedure that synthesizes an FM based on product descriptions.

Darvil [1] proposed new approach for creating feature model from descriptions of products that are available on online repository of products and Web sites such as CNET softPedia. As each individual product descriptions show only a part of domain’s features, a large collection of descriptions can provide a fairly comprehensive coverage. In this method tow initial phase are introduced. One advantage of this method is availability of product description for each type of products to the public, it means that this method can also be used in an organization that has not developed the software for the target domain.

This [13] work presents an approach based on formal concept analysis that analyzes incidence matrices containing matching relations as input and creates feature models as output. New optimal approach presented here, do this conversion in a reasonable time even when the product is a high number. Incidence matrix is introduced as a new concept which describes the common and different artifacts of variants. Using this method leads to obtain feature model graph from conceptual graph directly.

Horatio [14] use mining techniques on public product description and use clustering algorithms for finding domain specific features and use probabilistic model. After they use association rules and k-means machine learning strategies for defining specific characteristics of a special product.

Mathieu [15] review the challenges of extracting a lot of feature models from the same input configuration that only a few of them are meaningful and can be maintained, so authors define a specific criteria for their separation. The challenge is to study various configuration models that are possible to obtain from one configuration, but only few of them are meaningful and sustainable. Various features model are identified primarily, and then key characteristics that distinguishing them is determined. Finally, with a public policy that is based on the understanding and knowledge of user, the feature model is constructed.

Krzysztof Czarnecki et al 's [16] approach introduce probability feature model and show how to obtain probability feature model from data mining techniques from feature configuration set. The author believes that the results are the basic foundation to build reverse engineering software product line tools.

Nathan Weston et al's method [7] introduce a tool suite which automatically processes natural-language requirements documents into a candidate feature model, which can be refined by the requirements engineer.

4. CONCLUSION

In order to compare reviewed methods, some criteria are proposed. These criteria are: customer requirement priority, product quality predictability, non-functional requirements selectivity, considering integrity constraint and Feature model extraction source.

Suggested criteria , customer requirement priority, considering integrity constraint are inspired from [17]. Product quality predictability and non-functional requirements selectivity are from [18] and 19]. Feature model extraction source is considered to compare method's precondition and input.

- customer requirement priority [1], [12], [3] and [3] don't have any priority. [20] do requirements priority in the form of user-defined functions that reflect the priorities of their target states.

- Product quality predictability: Since the product line include huge number of features and the number of products that can be produced with these features is growing exponentially, so the possibility of making all products and to measure their quality is far-fetched. The ability to predict the quality of individual products without making desirable products is desirable and effective manner in managing product line and marketing. Since and [15] and [1] extract feature model after product development, haven't this ability. [3] have not expressed any predictive view for their quality products [20] provide a way to measure and predict the product's quality.

- Non-functional requirements selectivity: Non-functional requirements in the development of a single product collected and documented before development. During development, such as non-functional requirements frameworks tools, help the developers design decisions that affect the final product properties. But in product line approach, the product is designed and produced for a range of clients with different non-functional requirements. Sometimes different customers may need opposing non-functional requirements. So the ability of choosing non-functional features for each customer is a mean to enhancing product quality and meet customer demands. [3], [12], [7] don't consider this capability. [20] method has this ability to predict product quality before product development by considering selected features.

- Considering integrity constraint: In addition to the parent-child relationship between the features, integrity constraints for domestic mutual dependencies between features are presented. The most important provisions of the integrity of the proposed are "include" and "exclude". [3], [1] and [12] don't use integrity constraint.in Nathan Weston method just use optional and alternative feature's constraints.

- Feature model extraction source: feature model extraction methods use different inputs for building feature model and cause differentiation between them. Source used in each way is presented in the table given below:

Feature model extraction source	Method
Public product description	Jean-Marc Davril,2013
Tabular data files	Mathieu Acher,2012
Customer requirements and common domain requirements	Conqueror,2011
Common domain requirements	FODA ,1990
Online public product description	Dumitru,2014
Possible configuration set	Acher, Baudry, 2013
Data mining techniques from feature configurations	Czarnecki, She,2008
Using (RDF) Request Defining Language and requirement definition	Nathan Weston, 2009

TABLE 2: Comparison extraction's model feature source.

Although a large variety of different methods have been proposed to extract feature model in product line, the development team must use appropriate method based on their emerging needs and their available inputs.

5. FUTURE WORK

A large variety of different feature model extraction methods have been proposed in software product line. There is the possibility of converting feature model to product line architecture. We will do future research on how to convert feature model to software product line architecture.

6. REFERENCES

[1] Davril, J.-M., et al. *Feature model extraction from large collections of informal product descriptions*. in *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*. 2013. ACM.

[2] Van der Linden, F.J., K. Schmid, and E. Rommes, *Software product lines in action: the best industrial practice in product line engineering*. 2007: Springer Science & Business Media.

[3] Kang, K.C., et al., *Feature-oriented domain analysis (FODA) feasibility study*. 1990, DTIC Document.

[4] Apel, S., et al., *Feature-Oriented Software Product Lines*. 2013: Springer.

[5] Matinlassi, M. *Comparison of software product line architecture design methods: COPA, FAST, FORM, KobrA and QADA*. in *Proceedings of the 26th International Conference on Software Engineering*. 2004. IEEE Computer Society.

[6] Sochos, P., M. Riebisch, and I. Philippow. *The feature-architecture mapping (farm) method for feature-oriented development of software product lines*. in *Engineering of Computer*

Based Systems, 2006. ECBS 2006. 13th Annual IEEE International Symposium and Workshop on. 2006. IEEE.

[7] Weston, N., R. Chitchyan, and A. Rashid. *A framework for constructing semantically composable feature models from natural language requirements*. in *Proceedings of the 13th International Software Product Line Conference*. 2009. Carnegie Mellon University.

[8] Pleuss, A., et al., *Model-driven support for product line evolution on feature level.* Journal of Systems and Software, 2012. **85**(10): p. 2261-2274.

[9] Heradio-Gil, R., et al., *Supporting commonality-based analysis of software product lines.* IET software, 2011. **5**(6): p. 496-509.

[10] Acher, M., et al., *Reverse engineering architectural feature models*, in *Software Architecture*. 2011, Springer. p. 220-235.

[11] Acher, M., et al., *Familiar: A domain-specific language for large scale management of feature models.* Science of Computer Programming, 2013. **78**(6): p. 657-681.

[12] Acher, M., et al. *On extracting feature models from product descriptions*. in *Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems*. 2012. ACM.

[13] Ryssel, U., J. Ploennigs, and K. Kabitzsch. *Extraction of feature models from formal contexts*. in *Proceedings of the 15th International Software Product Line Conference, Volume 2*. 2011. ACM.

[14] Dumitru, H., et al. *On-demand feature recommendations derived from mining public product descriptions*. in *Software Engineering (ICSE), 2011 33rd International Conference on*. 2011. IEEE.

[15] Acher, M., et al. *Support for reverse engineering and maintaining feature models*. in *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*. 2013. ACM.

[16] Czarnecki, K., S. She, and A. Wasowski. *Sample spaces and feature models: There and back again*. in *Software Product Line Conference, 2008. SPLC'08. 12th International*. 2008. IEEE.

[17] Asadi, M., et al., *The effects of visualization and interaction techniques on feature model configuration.* Empirical Software Engineering, 2014: p. 1-38.

[18] Sincero, J., W. Schroder-Preikschat, and O. Spinczyk. *Approaching non-functional properties of software product lines: Learning from products*. in *Software Engineering Conference (APSEC), 2010 17th Asia Pacific*. 2010. IEEE.

[19] Siegmund, N., et al. *Scalable prediction of non-functional properties in software product lines*. in *Software Product Line Conference (SPLC), 2011 15th International*. 2011. IEEE.

[20] Siegmund, N., et al., *SPL Conqueror: Toward optimization of non-functional properties in software product lines.* Software Quality Journal, 2012. **20**(3-4): p. 487-517.