# Aspect-Oriented Feature Modelling for Software Product Line

Guoheng Zhang

*School of Electrical Engineering and Computer Science, University of Newcastle*
*Callaghan 2308, NSW, Australia*
[1]Guoheng.Zhang@studentmail.newcastle.edu.au
*Phone Number: [02]4921-6317, Enrolment Date: 12/08/08*

***Abstract*: Complex dependency modelling is one of the critical issues in feature modelling. The complex dependency, a kind of feature relationship among a set of features, is caused by the system concerns, such as system mobility and system security. Existing feature modelling approaches do not have sufficient support to model the system concerns in feature models. Furthermore, the proposed feature dependency types, such as requires and excludes, are not sufficient to represent the complex dependencies. We propose an aspect-oriented feature modelling approach to identify and represent the complex dependencies. In this approach, a concern modelling method, which relates the features with system concerns, will be developed to identify the complex dependencies. Furthermore, aspectual features, which represent the system concerns, will be introduced into feature models. A prototype tool will be developed to support the conceptual work and a case study will be performed to validate the proposed approach.**

***Keywords*— Aspect-Oriented Modelling, Concern Modelling, Crosscutting Concerns, Complex Dependencies, Contribution Analysis.**

## I. INTRODUCTION

Software product line (SPL) [1] [2] is emerging as an attractive software-reuse approach. A SPL consists of a set of software-intensive systems which share a common set of features that satisfy the specific requirements of a particular market. In addition to the shared commonalities, members of a SPL have different characteristics that are called variabilities. Based on the commonalities and variabilities of the SPL members, the core assets of a SPL can be produced. Developing a set of new software systems out of the common set of reusable core assets, as opposed to developing each software system separately, will achieve high-level constructive reuse and bring benefits, such as reducing time-to-market, improving software quality and promoting software development productivity.

Exploiting the commonalities and managing the variabilities are two critical issues in SPL-based software development. Currently, feature modelling has become a promising approach for capturing the commonalities and variabilities of members of a SPL [2]. Feature modelling has two key issues: modelling variabilities and modelling dependencies. Variabilities represent the configurable aspect of a SPL while dependencies specify the inter-relationships among the variabilities. In FODA (feature-oriented domain analysis) [2], Kang et al. proposed the feature models to represent the results of feature modelling and used the feature diagram, a tree-structure modelling language, to represent the feature models. Since then, although several approaches have improved the initial feature modelling approach proposed in FODA, some problems remain unresolved. One of the problems is how to identify and represent complex dependencies in feature models.

Dependencies play an important role in the product configuration during which application engineers select a set of features from the feature model based on the customer requirements [1]. The dependencies among a set of variabilities constrain the selection on variabilities. Existing feature modelling approaches have proposed several dependency types, such as “requires”, “excludes”, “hints” and “hinders” [3]. These feature dependencies represent a one-to-one feature relationship with explicit semantics. However, these simple dependency types are not sufficient to represent more complex dependencies caused by the system concerns, such as mobility and security. Furthermore, existing feature modelling approaches failed to identify these complex dependencies because they didn’t take into account the system concerns.

An aspect-oriented approach is proposed to identify and represent the complex dependencies in feature models. In the proposed approach, a concern modelling method, which aims at detecting the relationships between system concerns and features, will be developed to identify the complex dependencies. A new type of features (aspectual features), which aims at encapsulating the system concerns, will be introduced into feature models. A representation schema for the aspectual features will be developed to represent the complex dependencies. A contribution analysis method will be developed to detect the conflicts among system concerns in feature models. It is expected that the explicit identification and representation of these complex dependencies can improve feature models from the following aspects:

- Improve the completeness of a feature model. The identification of complex dependencies can improve the completeness of a feature model.
- Improve the adaptability of a feature model. The explicit representation of the complex dependencies can improve the ability of a feature model to change itself to adapt to the changed requirements.

The remainder of this paper is organized as follows: section 2 discusses the research problems and identifies the research questions. Section 3 proposes an approach to address the research problems. Finally, section 4 concludes our work.

## II. Research Problems

With respect to modelling dependencies, existing feature modelling approaches propose several simple dependencies, such as "requires", "excludes", "hints" and "hinders". "Requires" or "excludes" mean that the selection of one variable feature implies or excludes the need of selecting another variable feature. "Hints" or "hinders" indicate that the selection of one variable feature has some positive or negative influence on another variable feature [3]. However, these simple dependencies are not sufficient to represent all inter-relationships among features. The identification of these simple dependencies is usually based on the functionalities of features of product line members. However, some system concerns, such as mobility, security and availability, may crosscut a set of features and cause a kind of complex and implicit inter-relationships among the set of features. It is difficult to discover these complex dependencies without modelling the system concerns. However, current feature modelling approaches did not have sufficient support for system concern modelling. Consequently, the generated feature models based on these approaches may be incomplete and invalid as complex dependencies cannot be appropriately identified and represented in feature models.

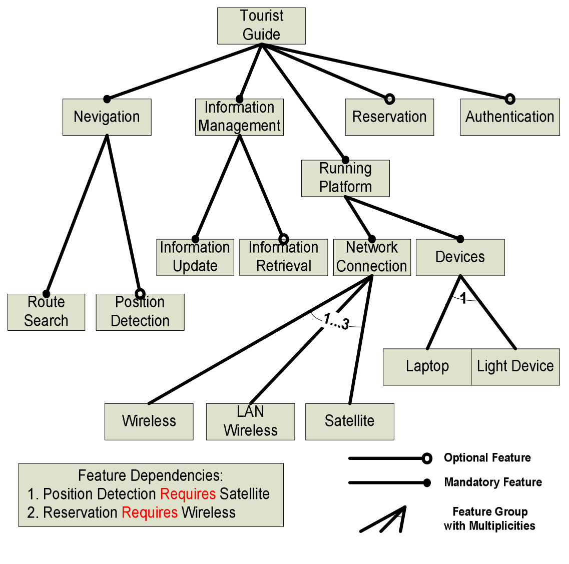

Figure 1 A traditional feature model

An example is shown in Fig. 1 and Fig. 2 to contrast two different feature models for the same "Tourist Guide" software product line. In the traditional feature model (see Fig 1), two simple requires-dependencies are identified based on the functionalities of features without considering system concerns. Based on the domain knowledge and product requirements, two system concerns, mobility and security are considered important to this software product line. For each of them, a set of features that are crosscut by this system concern are identified. For example, "Devices", "Network Connection" and "Position Detection" are identified as the features closely related with mobility of the system. A complex dependency can be discovered based on the relationships between the system mobility and these three features. This complex dependency constrains the selections on these three variabilities as follows: the high mobility will imply the selection of "Light Device" on "Devices", "Wireless" on "Network Connection" and "Position Detection"; the low mobility will imply the selection of "Laptop" on "Devices" and "LAN Wireless" on "Network Connection" and prohibit the selection of "Position Detection" and "Satellite". Therefore, after the complex dependencies have been identified based on system concerns we found, the original feature model shown in Fig. 1 is invalid and incomplete because some important dependencies were missed. The updated feature model including the identified additional dependencies is shown in Fig. 2.

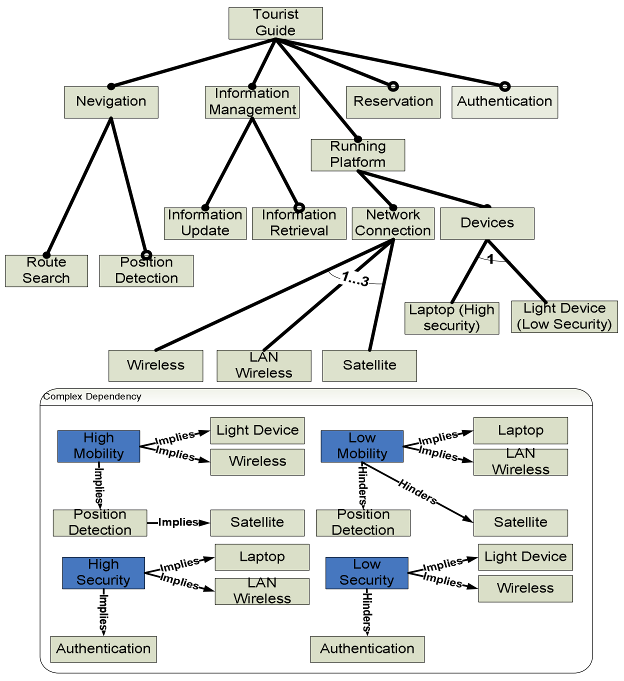

Figure 2 A feature model with complex dependencies

In fig. 2, two system concerns can be modelled as features. There newly introduced features representing system concerns may also have dependent relationships between them, e.g. one system concern may conflict with another system concern. In a product configuration, the conflict must be resolved by making trade-offs between two system concerns. This kind of trade-off may further constrain the selections of variabilities. In fig. 3, the conflict between two system concerns, security and mobility has been identified-high security and high mobility cannot be selected in a single product. If high security is required only low mobility can be accommodated, and vice versa. For example, the selection of "Light Device" on "Devices" and "Wireless" on "Network Connection" will imply the high system mobility. The high mobility will lead to the low security because of the conflict between the two system concerns. The low security will further constrain the selections on four variabilities: “Devices”, “Network Connection”, “Authentication” and “Information Retrieval”.

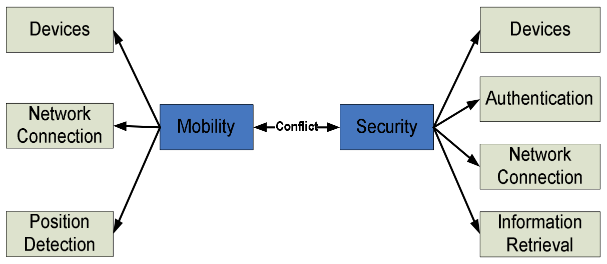

Figure 3 A conflict between two system concerns in feature models

Based on the previous introduction on complex dependencies, we formally define complex dependencies as follows:

- It may be a one-to-many feature relationship: a set of variable features in a feature model may be related with each other, because they contribute to one system concern.
- It may be a many-to-many feature relationship: a set of variable features (contributing to one system concern) may be related with another set of variable features (contributing to another system concern) because of the conflict between two system concerns.

The validation of a feature model is the prerequisite for configuring valid products from a feature model. For the traditional feature models, the validation aims at detecting the conflicts among a set of simple feature dependencies, such as “requires” and “excludes”. The validation of feature models in our work is more complex, because the system concerns, which cause complex dependencies, are introduced into feature models. Besides the conflicts among simple dependencies, the conflicts among system concerns also need to be identified. In fig. 3, there exists a conflict between security and mobility. This conflict is identified from two variabilities, “Devices” and “Network Connection”, which are crosscut by both mobility and security. For example, the high mobility will imply the selection of “Light Device” and “Wireless” while the high security will imply the selection of “Laptop” and “LAN Wireless”. However, “Laptop” will conflict with “Light Device”; and further “Wireless” and “LAN Wireless” cannot be selected in a single product. Therefore, the high security conflicts with the high mobility in this feature model. The products, requiring both high security and high mobility, cannot be configured from this feature model. If this feature model needs to support this kind of products, the feature dependencies must be modified.

One research question is identified from the research problem: How to identify and represent the complex dependencies explicitly in a feature model? This research question has two critical issues: the identification method for detecting the complex dependencies and the explicit representation schemas for representing the complex dependencies in a feature model.

## III. Proposed Approach

An aspect-oriented approach (also refers to aspect-oriented feature modelling) is proposed to identify and represent the complex dependencies explicitly in a feature model. It introduces aspect-oriented methodology into feature modelling and generates an aspect-oriented feature model (AO-FM).

Aspect-oriented methodology is used to identify and represent crosscutting concerns at different abstract-level models [5]. At requirement level, a crosscutting concern crosscuts several other concerns by affecting their requirements. In feature models, a system concern also crosscuts a set of variable features by affecting their requirements to constrain their selections. In this sense, the system concerns that cause complex dependencies in a feature model can also be regarded as crosscutting concerns. Aspect-oriented modelling method is used to represent the crosscutting concerns and the crosscutting relationships in high abstract-level models, especially requirement models [6]. Feature models are also coarse-grained requirement models. Therefore, the aspect-oriented modelling method is a potential approach for representing the complex dependencies which are caused by the system concerns in a feature model.

The proposed approach consists of three sections: a conceptual framework, a prototype tool and an evaluation method.

### A. *Conceptual Framework*

The conceptual framework provides the methodology that is used to identify and represent complex dependencies in feature models. To identify complex dependencies, we develop a concern modelling method to model functional concerns, system concerns and concern relationships. The mapping between the functional concerns in a concern model and the functional features in a feature model will be established by relating their requirements. Next, the features crosscut by a system concern can be discovered by the concern relationships and the mapping relationships between a feature model and a concern model. Finally, a complex dependency among the identified features can be established based on the domain knowledge on the system concerns and the crosscut features. To represent the complex dependencies, we adopt the philosophy of aspect-oriented modelling methods and introduce a new kind of features (aspectual features) to encapsulate the system concerns and represent the complex dependencies. Further, we propose a contribution analysis method to detect the conflict between two system concerns.

1) *Concern Modelling:* Concern modelling aims at identifying concerns and concern relationships for a software product line and organizing these concerns in a concern model. We establish a mapping between a concern model and a

feature model by relating their requirements. The features crosscut by a system concern can be discovered through the concern relationships and the mapping between a concern model and a feature model. A complex dependency can be established among the discovered features. There are four critical issues in the concern modelling:

- ***A general concern space*** consists of the most general concerns. It provides a framework for identifying and specifying the concerns and concern relationships for a specific system or a specific software product line.
- ***A meta-model for concern models*** defines the basic elements of a concern model and the relationships between these elements.
- ***Representation schemas for concerns and concern relationships*** provide the structures to specify the concerns and concern relationships.
- ***Mapping between functional concerns and functional features*** bridges the concern model and the feature model and provides a way to discover the features crosscut by a system concern.

2) *Aspectual Features:* The system concerns identified in concern modelling are represented as aspectual features in feature models. The set of features crosscut by a system concern are modelled as joinpoint features. To represent the relationship between an aspectual feature and its associated joinpoint features, the representation schemas for aspectual features need include a set of elements:

- ***Role***: Each aspectual feature has a property "aspectual" and each joinpoint feature has a property "joinpoint" to indicate their roles.
- ***Target***: Each aspectual feature has a mapping target when transforming a feature model to an architecture model. The mapping target may be an aspectual element in architecture models or a design decision when designing the architecture.
- ***Composition Mechanism***: Each aspectual feature has a composition mechanism to represent the relationship between an aspectual feature and its associated joinpoint features. The composition mechanism includes a quantification method to match an aspectual feature to a set of joinpoint features.
- ***Advice***: Each aspectual feature has an advice to encapsulate the rules for describing how the aspectual feature constrains the selections on its associated joinpoint features.

3) *Contribution Analysis:* The contribution analysis method aims at identifying the conflict between two aspectual features. This identification is based on the contribution of one aspectual feature to another aspectual feature. The contribution may be positive, negative or none. If no features are crosscut by both of the two aspectual features, the contribution is none. If the system concern level of one aspectual feature has some positive or (negative) influence on the system concern level of another aspectual feature through the variabilities crosscut by both of the two aspectual features, the contribution is positive or negative. Therefore, the negative contribution between two aspectual features indicates a conflict between them. The conflicts must be resolved by making trade-off decisions.

### B. *Prototype and Evaluation*

A prototype tool will be developed to support the conceptual framework. It will provide the following functions:

- ***Modelling:*** the prototype tool will provide the users with an interface to input the elements of their feature models and provide a window for visualizing the generated feature models. The generated feature models must conform to the meta-model of the aspect-oriented feature models.
- ***Product Configuration:*** the prototype tool will provide an interface to make trade-offs between two conflicting aspectual features. The system concern levels in the aspectual features can be selected based on the trade-offs or the customer requirements. Further, the prototype tool will support to select the desired features from feature models. The validity of the configured product will be tested by checking whether the selected features conflict with the feature dependencies.

This approach will be validated on two issues: complex dependency identification and complex dependency modelling. Two medium-size case studies (tourist guide and smart home software product line) and one industry-size case study (library system product line) will be performed on existing feature modelling approaches (such as FODA and FORM) and aspect-oriented feature modelling approach to validate the applicability of the proposed approach from two aspects: the quality requirements satisfaction in the derived applications and the inconsistencies existing in the derived applications because of quality concerns.

## IV. CONCLUSIONS

This paper proposes an approach to identify and represent complex dependencies in a feature model. It is expected that the successful completion of this work will improve the completeness and adaptability of traditional feature models. In the future, we will focus on the detailed implementation of the proposed approach.

## REFERENCES

[1] K. Czarnecki, S. Helsen, and U. Eisenecker, "Formalizing Cardinality-based Feature Models and their Specialization," in Software Process Improvement and Practice, vol. 10(1), 2005, pp. 7-29.

[2] Kyo C. Kang, S. Cohen, J. Hess, W. Novak, and A. Petersem, "Feature-Oriented Domain Analysis (FODA) Feasibility Study," Tech. Rep. CMU/SEI 90-TR-21, 1990.

[3] S. Buehne, G. Halmans, and K. Pohl, "Modelling Dependencies between Variation Points in Use Case Diagrams," in Proceedings of 9th International Workshop on Requirements Engineering, 2003.

[4] A. Schauerhuber, W. S., E. Kapsammer, W. Retschizegger, M. Wimmer, and G. Kappel, "A survey on Aspect-Oriented Modeling Approaches," 2006.

[5] A. Rashid, P. Sawyer, A. M.D. Moreira, and J. Araujo, "Early Aspects: A Model for Aspect-Oriented Requirement Engineering," in Proceedings of the 10th Anniversary IEEE Joint International Conference.

[6] http://en.wikipedia.org/wiki/Cross-cutting_concern