Towards a UML Profile for Software Product Lines

Tewfik Ziadi¹, Loïc Hélouët¹, and Jean-Marc Jézéquel²

¹ IRISA-INRIA, Campus de Beaulieu 35042 Rennes Cedex, France
{tziadi,lhelouet}@irisa.fr
² IRISA-Rennes¹ University, Campus de Beaulieu 35042 Rennes Cedex, France
jezequel@irisa.fr

Abstract. This paper proposes a UML profile for software product lines. This profile includes stereotypes, tagged values, and structural constraints and it makes possible to define PL models with variabilities. Product derivation consists in generating product models from PL models. The derivation should preserve and ensure a set of constraints which are specified using the OCL.

1 Introduction

The Unified Modeling Language (UML) [5] is a standard for object-oriented analysis and design. It defines a set of notations (gathered in diagrams) to describe different aspects of a system: use cases, sequence diagrams, class diagrams, component diagrams and statecharts are examples of these notations. A UML Profile contains stereotypes, tagged values and constraints that can be used to extend the UML metamodel.

Software Product Line engineering aims at improving productivity and decrease realization times by gathering the analysis, design and implementation activities of a family of systems. Variabilities are characteristics that may vary from a product to another. The main challenge in the context of software Product Lines (PL) approach is to model and implement these variabilities. Even if the product line approach is a new paradigm, managing variability in software systems is not a new problem and some design and programming techniques allows to handle variability; however outside the Product Line context, variability concerns a single product, i.e variability is inherent part of a single software and is resolved after the product is delivered to customers and loaded into its final execution environment. In the product line context, variability should explicitly be specified and is a part of the product line. Contrarily to the single product variability, PL variability is resolved before the software product is delivered to customers. [1] calls the variability included in a single product “the run time variability”, and the PL variability is called “the development time variability”. UML includes some techniques such as inheritance, cardinality range, and class template that allow the description of variability in a single product i.e variability is specified in the product models and resolved at run time. Furthermore, it is interesting to use UML to specify and to model not only one product but a set of products. In this case the UML models should be considered as reference models from which product models can be derived and created. This variability corresponds to the product line variability. In this paper we consider this type of

* This work has been partially supported by the FAMILIES European project. Eureka Σ! 2023 Program, ITEA project ip 02009.
variability and we use UML extension mechanisms to specify product line variability in UML class diagrams and sequence diagrams. A set of stereotypes, tagged values and structural constraints are defined and gathered in a UML profile for PL.

The paper is organized as follows: Section 2 presents the profile for PL in terms of stereotypes, tagged values and constraints, Section 3 presents the use of this profile to derive product models from the PL, Section 4 presents related work, and Section 5 concludes this work.

2 A UML Profile for Product Lines

The extensions proposed here for PL are defined on the UML 2.0 [5] and they only concern the UML class diagrams and sequence diagrams. We use an ad-hoc example to illustrate our extensions. The example concerns a digital camera PL. A digital camera comports an interface, a memory, a sensor, a display and may comport a compressor. The main variability in this example concerns the presence of the compressor, the format of images supported by the memory, which can be parameterized and the interface supported. We distinguish three types of interfaces: Interface 1, Interface 2, and Interface 3.

2.1 Extensions for Class Diagrams

UML class diagrams are used to describe the structure of the system in terms of classes and their relationships. In the context of Product Lines, two types of variability are introduced and modeled using stereotypes.

Stereotypes

– **Optionality.** Optionality in PLs means that some features are optional for the PL members. i.e: they can be omitted in some products. The stereotype `<optional>` is used to specify optionality in UML class diagrams. The optionality can concern classes, packages, attributes or operations. So The `<optional>` stereotype is applied to Classifier, Package and Feature metaclasses.

– **Variation.** We model variation point using UML inheritance and stereotypes: each variation point will be defined by an abstract class and a set of subclasses. The abstract class will be defined with the stereotype `<variation>` and each subclass will be stereotyped `<variant>`. A specific product can choose one or more variants. These two stereotypes extend the metaclass Classifier. The alternative variability especially defined in feature driven approaches is a particular case of our variation variability type where each product should choose one and only one variant. This can be modeled using OCL (Object Constraint Language) [10] as a mutual exclusion constraint between variants. The mutual exclusion constraint will be presented in Section 3.

Constraints. A UML profile also includes constraints that can be used to define structural rules for all models specified by the defined stereotypes. An example of such profile