An OCL Formulation of UML2 Template Binding

Olivier Caron, Bernard Carré, Alexis Muller, and Gilles Vanwormhoudt

Laboratoire d’Informatique Fondamentale de Lille
UMR CNRS 8022
Université des Sciences et Technologies de Lille
F-59655 Villeneuve d’Ascq cedex, France

Abstract. After being considered only as documentation for a long time, models are gaining more and more importance in the software development lifecycle, as full software artefacts. The UML standard contributes a lot to this mutation, with the identification and the structuration of models space dimensions and constructs. Models can nowadays be explicitly manipulated through metamodeling techniques, dedicated tools or processes such as model transformation chains. This is "Model Driven Engineering". Once it is clear that models are full software ingredients, we are faced with new problems (needs!) such as the possibility of their reusability and composability. As a consequence, specific constructs are introduced in order to facilitate this, such as the template notion initiated by UML1.3. Applications of this notion are growing more and more so that it was deeply revisited and strengthened in UML2. Though, its specification still lacks precision, particularly concerning the "binding" mechanism that allows to obtain models from templates. We propose a set of OCL constraints which strengthens the definition and helps in verifying the correctness of resulting models. These constraints apply to the UML2 metamodel and were implemented in an OCL verifier that we integrated in the Eclipse environment.

1 Introduction

After being considered only as documentation elements for a long time, models are gaining more and more importance in the software development lifecycle, as full software artefacts. The UML [2] standard contributes a lot to this mutation, with the identification and the structuration of models space dimensions and constructs. Models can nowadays be explicitly manipulated through metamodeling techniques, dedicated tools or processes such as the MDA [1] transformation chains. This is "Model Driven Engineering" [9]. The main motivation is the reduction of delays and costs by the capitalization of design efforts (models) at each stage, and the automation, as far as possible, of transitions between these stages. So it would be possible to separate high level business oriented models from low level architectural and technological ones, but also to reuse these models from one application to another.
Indeed, once it is clear that models are full software ingredients, we are faced with new problems (needs!) such as the possibility of their reusability and composability. As a consequence, models stand more and more as good candidates for the "design for reuse" quest and specific constructs are introduced to make them generic. It is the case of the UML template notion which helps in specifying parameterized models. Applications are patterns formulation [3] [4], modelization of reusable subjects in [5] or frameworks in Catalysis [6]. We also use templates to specify modeling components which capture reusable functional aspects [10].

UML templates applications are numerous and various, with the result that its initial introduction in UML1.3 was deeply revisited and strengthened in the UML2 standard. Though its specification remains much more structural and verbal in [3]. Particularly, constraints lack for the precise definition of the related "binding" relation which allows to obtain models from templates. These constraints are needed to verify the correctness of the resulting models. That is why we propose here a set of OCL constraints [11] which could strengthen the notion of model templates and facilitate the above verification. These constraints apply to the UML2 metamodel.

In the following section, we show the notion of templates through examples of parameterized model elements. Then (section 3), we present the UML metamodels of templates and the template binding as they are specified in [3] and explain them with the help of the preceding examples. We so (section 4) propose a set of OCL constraints which could complete this specification and help in verifying the correctness of resulting models. These constraints were checked in an OCL verifier that we integrate in the Eclipse environment (section 5).

2 The UML 2 Template Notion

In UML standard, a template is a model element which is parameterized by other model elements. Such parameterizable elements can be classes or packages, so called respectively class templates or package templates. To specify its parameterization, a template element owns a signature. A template signature corresponds to a list of formal parameters where each parameter designates an element that is part of the template. Template elements have also a specific notation which consists in superimposing a small dashed rectangle containing the signature on the right-hand corner of the standard symbol.

A template can be used to generate other model elements using template binding relationship. A bind relationship links a "bound" element to the signature of a target template and specifies a set of template parameter substitutions that associate actual elements to formal parameters. The binding of a bound element implies that its contents are based upon the contents of the target template with any element exposed as formal parameter substituted by the actual element specified in the binding.

Figure 1 shows a class template on the left. This class, Stack, is graphically represented as a standard UML class with a dashed rectangle containing its sig-