Dimension Templates: Multi-dimensional separation of concerns in UML

Marcus Fontoura
Computer Science Department, Princeton University
35 Olden Street, Princeton, NJ 08544-2087, U.S.A.
e-mail: mfontoura@acm.org

This paper introduces the concept of Dimension Templates to support the instantiation of the Hyperspaces model for UML design diagrams. Dimension Templates represent composition rules through UML parameterized collaborations. An environment that supports these concepts is also described.

Introduction

The Hyperspace model [3] provides a way of achieving multi-dimensional separation of concerns through the software development lifecycle. On the other hand, the recent standardization efforts of the Unified Modeling Language (UML) [2] provide a unique chance to harness UML as notational basis for software development projects. However, UML does not support separation of concerns in dimensions other than the object dimension. This paper shows how multi-dimensional separation capabilities may be added to UML. The concept Dimension Templates is introduced to achieve this goal. Dimension Templates represent composition rules through UML parameterized collaboration diagrams.

The rest of this paper shows how Dimension Templates may be successfully applied to the Software Engineering Environment (SEE) example proposed in [3], proposes a supporting environment (not yet implemented), and presents our conclusions and future research directions.

Modeling example

Figure 1 models the SEE example in UML. If there are two dimensions of separation (object and feature), additive system evolution should be achieved either through the addition of new slices in the object dimension or through the addition of new slices in the feature dimension. Figures 2 and 3 shows Dimension Templates that model the addition of new objects and features, respectively. The dashed relationships represent the UML biding stereotype, which represents the instantiation the parameterized collaborations (modeled by the dashed circles).

In the object dimension, the new slices are classes modeled by the *NewObject* role. Figure 2 exemplifies the addition of a new object slice: *Multiply*.

In the feature dimension, the new slices are collaborations modeled by the *Expression, Number, UOperator, BOperator, Plus, Minus, UPlus,* and *UMinus* roles. In each of these roles only the method *feature(),* which implements the new feature being added, must be implemented. Figure3 illustrates the definition of the *Persistence* feature as a new slice.

Every time a new slice is added to a given dimension, the Dimension Template diagrams for the other dimensions must be updated. Figures 4 and 5 illustrate this interdependency for the object and feature dimensions. The Dimension Template diagrams (Figures 2 and 3) and their evolution through the addition or exclusion of new slices (Figures 4 and 5) implicitly represent the Hyperspace composition rules.

Support environment

An environment that supports the instantiation of the Hyperspaces model for UML artifacts should allow the definition of Dimension Templates and the definition of rules for the Dimension Templates evolution. The designer should be able to evolve the system by adding and removing slices in a given dimension. To achieve that, the environment has to maintain a list of dimensions and, for each dimension, a list of slices.

To add a new slice the designer has to select a desired dimension and to instantiate the appropriate Dimension Template. After the change the environment has to update all (or some) of the other Dimension Templates using the evolution rules. This last step may be automatically performed or may require human interaction. Figure 6 illustrates the process of adding a new object slice. Note that the evolution of the feature Dimension Template cannot be automatically performed in this case.

To remove a slice the designer has to select the proper dimension and remove the slice from the list of slices in that dimension. After the exclusion the Dimension Templates must be updated using the evolution rules, which may require human interaction. Figure 7 illustrates the process of removing object slices. In this case the Dimension Template evolution can be automatically performed: the class that models the object is removed from the collaboration.

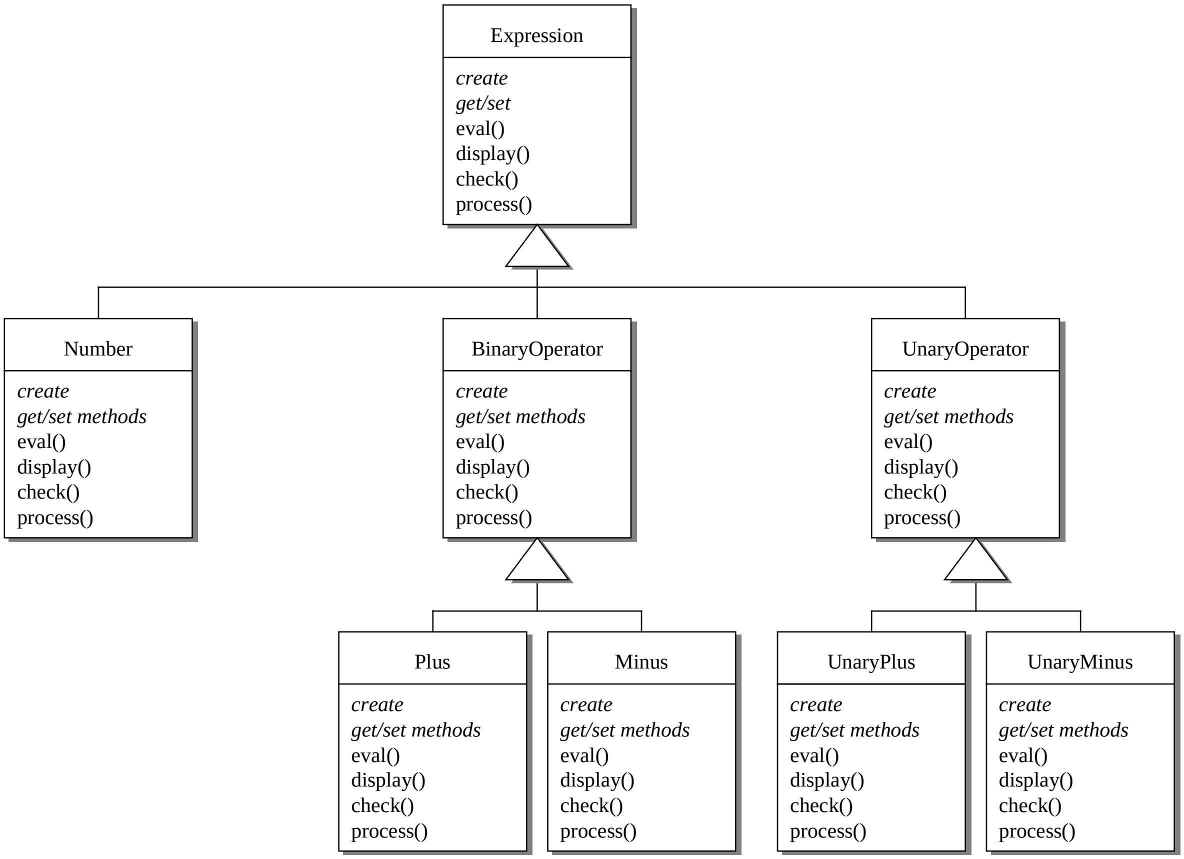


Figure 1. SEE class hierarchy

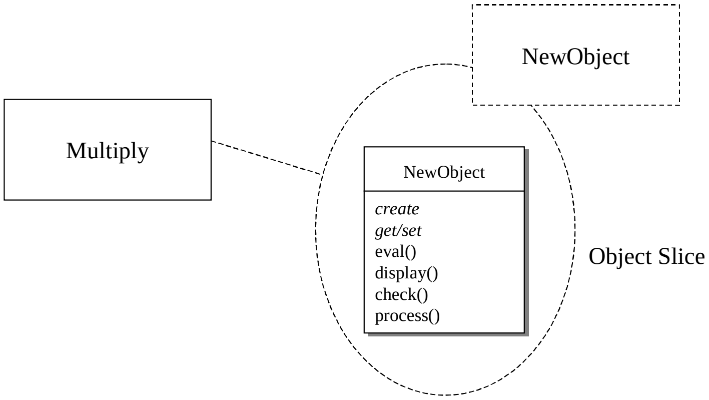


Figure 2. Object dimension template

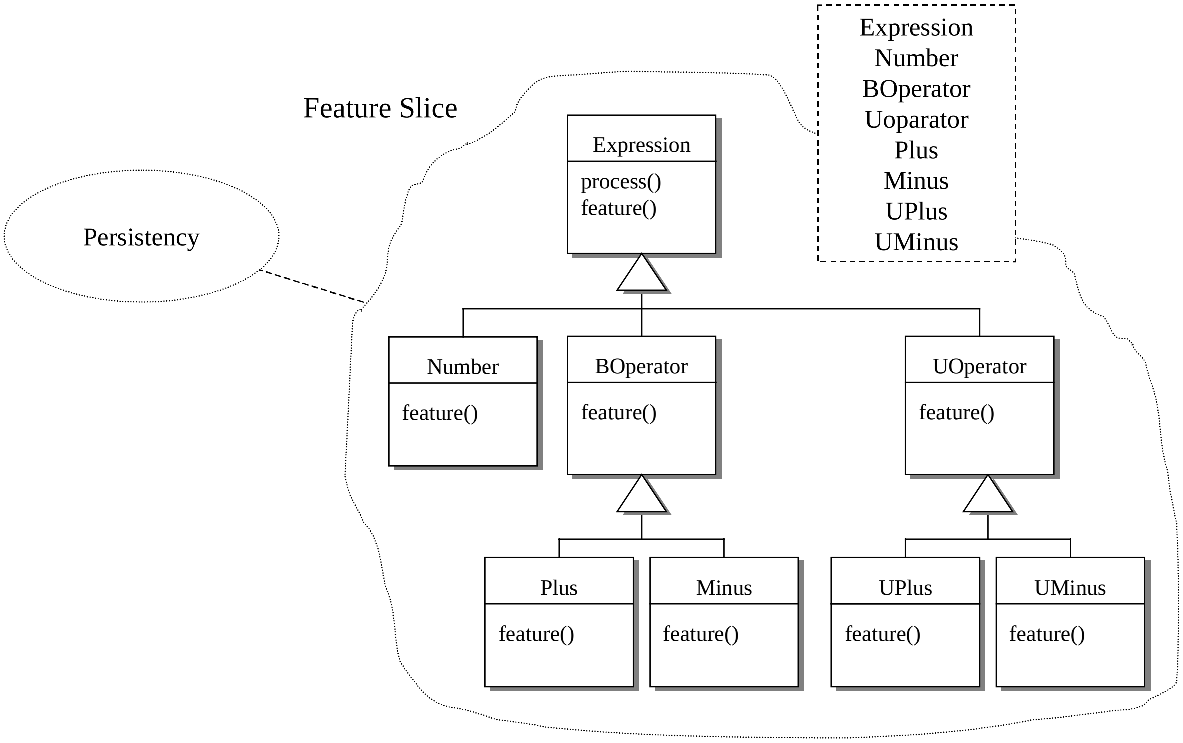


Figure 3. Feature dimension template

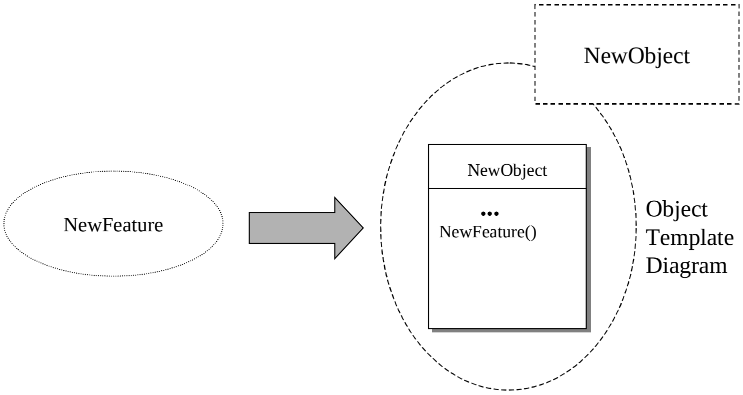


Figure 4. Adding a new feature: modifications in the object dimension template

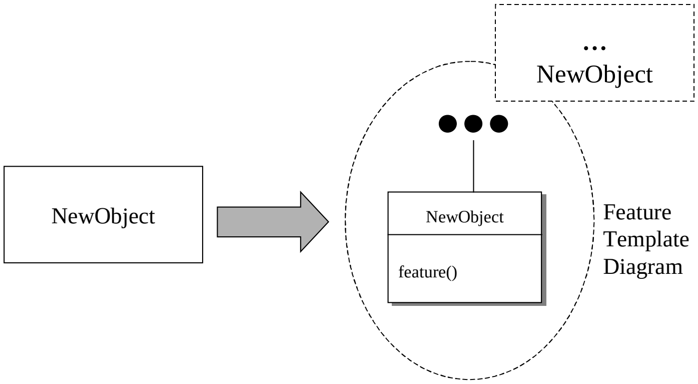
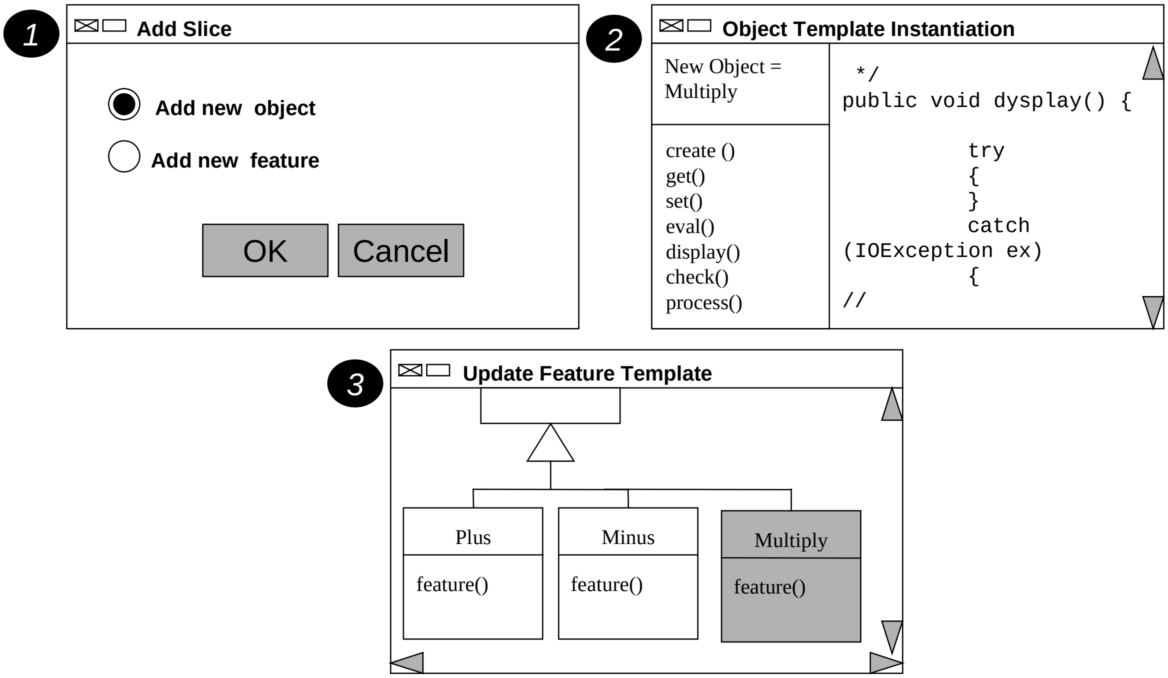


Figure 5. Adding a new object: modifications in the feature dimension template



● Add new object
○ Add new feature

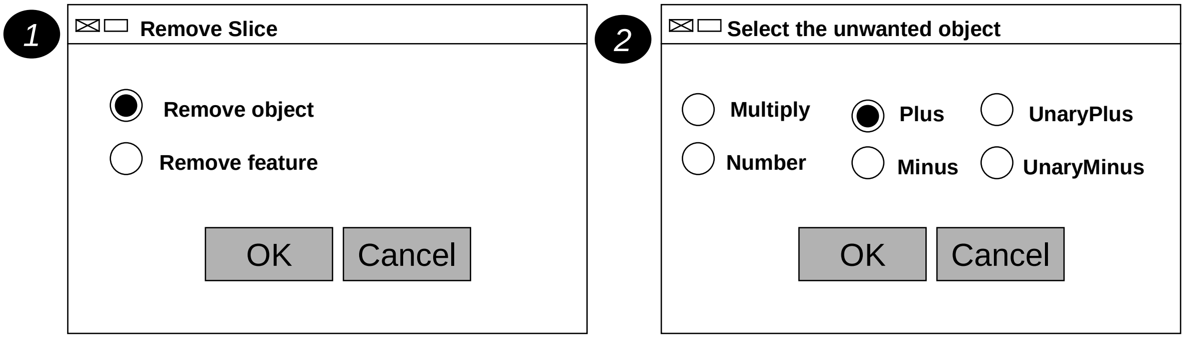
New Object = Multiply

create ()
get()
set()
eval()
display()
check()
process()

```
 */
public void dysplay() {

         try
         {
         }
         catch
(IOException ex)
         {
//
```

Figure 6. Environment support for adding a new object



● Remove object
○ Remove feature

○ Multiply ● Plus ○ UnaryPlus
○ Number ○ Minus ○ UnaryMinus

Figure 7. Environment support for removing objects

Conclusions and future work

This paper has shown how Dimension Templates may be defined to add multi-dimensional separation capabilities to UML. Dimension Templates build on the sound basis of UML standardization and propose no radically new path. The main goal of this position paper is to disseminate this idea and to discuss its usability with the community.

We are now developing an environment to support framework development based on the extension of an UML case tool [1]. We would like to add the support for Dimension Templates in this environment. Another area of interest is the discovery and experimentation with other dimensions (other than object and feature).

References

1. M. Fontoura, “A Systematic Approach for Framework Development“, Ph.D. Thesis, Computer Science Department, PUC-Rio, 1999 (http://www.les.inf.puc-rio.br/~mafe).

2. J. Rumbaugh, I. Jacobson, and G. Booch, *The Unified Modeling Language Reference Manual*, Addison-Wesley, 1998.

3. P. Tarr, H. Ossher, W. Harrison, and S.M. Sutton, Jr., “N Degrees of Separation: Multi-Dimensional Separation of Concerns”, Proceedings of the International Conference on Software Engineering (ICSE'99), 1999.