

# GReCCo: Composing Generic Reusable Concerns<sup>\*</sup>

Aram Hovsepian, Stefan Van Baelen, Yolande Berbers, Wouter Joosen

K.U.Leuven DistriNet, Celestijnenlaan 200A, B-3001 Leuven, Belgium  
{Aram.Hovsepian, Stefan.VanBaelen, Yolande.Berbers,  
Wouter.Joosen}@cs.kuleuven.be

## 1 Introduction

In this paper we give a brief description of GReCCo, an Aspect-Oriented Modeling based framework to promote and enhance the reuse of concerns. GReCCo supports (1) *composition obliviousness* by modelling concerns independently from a concrete context in which they are going to be applied, (2) *composition symmetry* by treating all concerns (including the base concern) uniformly, and (3) *interdependency management* by a coupling to the Concern Interaction Acquisition (CIA) system. We have developed a prototype composition engine implemented in ATL that can be used to compose concern models specified in UML.

## 2 General description of GReCCo

GReCCo enables the composition of oblivious concern models. We represent each composition step as the Greek letter  $\Upsilon$ . The left and the right branches of the  $\Upsilon$  contain two concern models. Our approach is composition symmetric in the sense that we treat *component* and *aspect* concerns uniformly. In order to combine the concern models, we provide a composition model that instructs the model transformation engine how the two models should be composed.

*Concern Models* describe the structure and the behavior of the concerns using respectively UML class and sequence diagrams. The *Composition Model*, which also consists of UML class and sequence diagrams using the GReCCo profile, specifies how the concern models are composed by defining all composition-specific parameters and their bindings. This assures a higher degree of reusability of the two concerns as they can be used in different contexts (*composition obliviousness*). The *Composition Model* describes how the source concern models should be composed, and is therefore by definition depending on these models. This allows to isolate the necessary links between the concerns into the composition model, thereby keeping the concerns oblivious and reusable. Using a generic composition engine, we generate the output *Composed Model* from the input composition and concern models.

---

<sup>\*</sup> The described work is part of the EUREKA-ITEA SPICES project, and is partially funded by the Flemish government institution IWT (Institute for the Promotion of Innovation by Science and Technology in Flanders), by the Interuniversity Attraction Poles Programme Belgian State, Belgian Science Policy, and by the Research Fund K.U.Leuven.

### 3 Composition Specification

In order to compose two concern models, we need to specify the composition by defining the *Composition Model*. Elements that are not directly referenced in the *Composition Model* are copied to the *Composed Model*. Other elements are modified by the composition engine according to the composition specification.

#### 3.1 Structure

We distinguish five structural composition directives in total. From the point of view of a single concern model, we distinguish the following directives that involve one element: (1) we can *add* a new element, (2) we can *modify* the properties of an existing element, and (3) we can *remove* an existing element. When two concern models are composed, there are some additional composition directives that we can specify for the input elements. We can (4) *merge* two elements to obtain a single entity with the combined properties. Some concerns introduce roles and/or template parameters as semantic variation points, which we can (5) *instantiate* by using concrete UML elements.

#### 3.2 Behavior

We use UML sequence diagrams to describe the behavior of concerns, which must be in correspondence with their structural counterpart specified by UML class diagrams. A concern model may contain several sequence diagrams. Each sequence diagram represents a certain scenario. Scenarios from the input concern models that are completely independent of each other, are simply copied to the composed model. We define two scenario's as independent of each other if they can be executed in parallel, meaning that the messages can be freely interleaved. For overlapping behavior scenarios we need to be able to address the following use-cases: (1) specify the sequence of messages between the input behavior scenarios, (2) indicate that a call from one input scenario is the same as a call from the other input scenario, and (3) replace a call or a set of calls from one input scenario by a call or a set of calls from the other input scenario.

To realize the composition, we introduce the UML2 notion of *general ordering*. A general ordering is a binary relation between two interaction fragments to describe that one of the fragments must occur before the other one. Each interaction fragment contains a set of events. The resulting scenario defines a partial ordering of the input events. The interpretation by the composition engine is that the dependency client fragment should immediately precede the dependency supplier fragment. Events that are not involved in any general ordering relation are put in parallel fragment blocks.

### 4 Future Work

In the future we plan to extend the GReCCo engine to support the behavioral composition and integrate it fully with the Concern Interdependency Acquisition (CIA) framework. We are also investigating the possibility to use domain-specific modeling languages for certain concerns. In addition, we plan to formalize the composition mechanisms and evaluate the scalability of our approach.