Revising the Comparison Criteria for Composition

Omar Alam, Matthias Schöttle, Jörg Kienzle

School of Computer Science, McGill University, Montreal, Canada
Omar.Alam@mail.mcgill.ca, mschoettle@cs.mcgill.ca,
Joerg.Kienzle@mcgill.ca

Abstract The Comparing Modeling Approaches (CMA) workshop proposed in 2011 a set of criteria that allows modellers to understand, analyze, classify and compare various modelling approaches. Based on feedback gained from applying the criteria, the criteria and questionnaire document were revised and extended multiple times. In this paper, we suggest a change to the criteria that is aimed at improving the assessment of model composition. We argue that the definition of *composition rule* is ambiguous in the current document, and suggest to replace it with an easier and more useful criterion – the *input specification*. We propose an updated questionnaire and show how to apply it to three modelling approaches: AoURN, RAM, and UML.

1 Introduction

The Bellairs modelling workshop and the follow-up Comparing Modeling Approaches workshop at MODELS 2011 proposed a comparison criteria [4] document for understanding, analyzing, classifying and comparing modelling approaches. The authors continuously revised the document over the last two years to further improve and clarify the criteria and extended it with additional comparison criteria from the "parking lot" [3,1]. Especially the section on composition in the criteria document has undergone several changes during the revisions. However, as authors of the original criteria document, we still find some terms and questions misleading. In particular, we find that the term "composition rule" is not clearly defined. The latest revision of the criteria document (April 2013) used for the CMA workshop at RE 2013 [1] acknowledges that the difference between "composition rule" and "composition operator" is not very clear. This became apparent when the authors of this paper individually applied the criteria to the Reusable Aspect Models (RAM) approach [2] and had different interpretations for these terms, which led to a long discussion on the essence of composition and a proposal on how to improve the composition section of the document, which is presented in this paper.

The rest of the paper is structured as follows. Section 2 discusses the issues with the term "composition rule" and proposes an alternative criterion: the "input specification". Section 3 presents the updated questionnaire and section 4 applies the updated questionnaire to some of the important composition operators of UML, RAM and AoURN [5]. Finally, the paper concludes in section 5.

2 Discussion on Composition Criteria

In section 2.2 of the most recent CMA comparison criteria [1], composition rules and operators are defined as follows: "A composition rule provides the specification of a composition, but does not actually perform the composition. A composition operator, on the other hand, results in a composed model". We had two possible interpretation of this definition:

- The composition operator is a function that executes the composition, but to do so it needs a composition rule that specifies how this task should be done. However, this appears not to be always the case as an example later on in the document states that the Control Flow Construct of UCM is "not a composition rule with a synonymous composition operator but rather just a composition operator".
- The composition operator specifies how a composition is to be performed and knows how to execute the composition, and the composition rule only specifies the input model elements to which the composition is to be applied. This also does not seem to be the case, as the signature of the example rules shown in the document (e.g., the UML association) use the implication operator, which suggests that the rule itself produces some output.

We were further confused by the following:

- A composition specification is defined in the document as "consisting of a composition rule and operator". Then, how can a rule "provide the specification of a composition" as stated in the definition of the composition rule?
- What does "the specification of a composition" mean? Does it mean the specification of the type of the inputs, the number of input parameters, or something else? From the signatures given in the examples in the document, it seems that it means the type and number of input. However, the composition operator signatures already specify the type and number of inputs, which makes the rule redundant.
- The example signatures for composition rules and composition operators given in the document did not help us to clarify the notions of rule and operator. For example, the rule and operator signatures of UML generalization and association look very similar.
- The document states that UML Association and Control Flow Construct are different, because the former is "not implemented when the "line" is drawn", whereas the latter "immediately results in the composition being performed". For the authors of this paper, UML Generalization, Association and Control Flow Construct compositions are in essence the same: when the operator is applied, the inputs are specified and the model is updated.

On top of that, the criteria document also mentions that the distinction between rules and operators is not clear. We believe that especially the notion of composition rule is vaguely defined, and in the end not useful. At least so far we failed to understand how composition rules would help in comparing

different modelling approaches, in particular since some example approaches in the document do not have rules for their operators.

To remedy this situation, we propose the following changes:

1. Revise the definition of *composition*: We suggest to change the definition of composition from "the act of creating new building blocks by using existing ones" to "the act of creating new or modifying existing modelling entities based on existing ones". Be believe that this change is necessary in order to not exclude approaches that do not produce new entities, but modify the entities provided as input to the composition.
2. Use the following definition for *composition operator*: A composition operator is an operational model transformation that combines existing first class entities from one or several models to produce a composed model that contains the composed entities. The composed model may be one of the input models (in which case the operator might only modify existing entities), or it may be a new model (in which case all entities are created anew). The signature of a composition operator defines the interface for the model transformation. It clearly states the number and type of input model elements that are required to apply the operator, as well as the number and type of output model elements.
3. Replace the *composition rule* used in the document with *input specification*: *An input specification specifies the input model elements to be used for a specific application of one or several composition operators.*

From 2 and 3 above it immediately follows that a composition operator can only be executed if the inputs are specified. We believe that input specification is a good comparison criteria because compositional modelling approaches widely differ in how they specify the inputs to their operators. Some approaches specify the inputs separately from the operators, while others do not. There are approaches that have the input specification as part of their language / in their models, and other approaches specify the inputs elsewhere, for example when applying an operator in a tool. In addition, it is possible in some approaches to specify the input without performing the operator. The next section explains how we redesigned the questionnaire to incorporate the suggested changes.

3 Updated Questionnaire

Based on the above, we updated the part of the questionnaire relating to composition. The new questions are aimed at determining the pronounced differences between modelling approaches supporting composition based on their composition operators and corresponding input specifications. The updated questionnaire is shown in Fig. 1. Questions A and K-U from the original questionnaire were omitted, since they weren't modified and are not relevant to our proposed changes. The resulting questionnaire contains questions A-T, of which C-T are to be repeated for all composition operators identified in question B. Comments on the questions are given below:

[Question A from original questionnaire]

B. List the composition operator(s) supported by the modelling approach:
C. To which language does the composition operator belong?
D. What is the signature of the operator?
E. Provide a brief textual description of the composition operator (i.e. its semantics).
F. Does the operator appear within a model (i.e., is the operator part of your language/metamodel)?
[ ] Yes [ ] No
G. Is the input specification for applying this operator part of a model (i.e., is the input specification for applying the operator part of your language/metamodel)? How is it specified?
[ ] Yes, how: _____ [ ] No, where/how: _____
H. If you answered "yes" for G, then answer the following: In your approach, is it possible to specify the inputs of the composition operator in a model without applying the operator? If yes, when is the operator applied?
[ ] Yes, when applied: _____ [ ] No
I. Does the result of the composition contain a modelling element that does not exist in the source models (i.e., does the composition add new model element(s) to the source models)?
[ ] Yes [ ] No

[Questions J-T are the same as questions K-U from original questionnaire]

Figure 1. Updated Section 2.2 of the Questionnaire

- D. The signature of the operator specifies the types and the numbers of inputs – the *formal parameters* – that need to be provided for the operator to be applicable. The type of output should also be specified. At this point it is also possible to indicate if the operator can accept input model elements from different models or not.
- E. The textual description should explain the semantics of the operator and mention any application constraints.
- F. In some modelling notations, the operator itself is represented in the model. For example, in UML class diagrams, an association (i.e. the association operator) is represented in a model by a line. From a metamodel perspective, an association shown in a model is an instance of the Association metaclass. In Protocol Modelling (PM) on the other hand, composition of concurrent state machines using the CSP (Communicating Sequential Processes) parallel composition operator (///) is implicit. In other words, nothing in a PM model represents the // operator. Likewise, the PM metamodel does not define the || operator.
- G. Before a composition operator can be applied, the input model elements must be specified. In other words, an actual parameter has to be provided for each formal parameter specified in the signature of the composition operator. Most of the time, an input specification for applying a composition operator can be specified in the model itself. For example, in UML class diagrams, an association line is drawn between 2 (or more) classes, which designate the inputs for the association operator. On the other hand one can also imagine a modelling approach that provides a tool that merges the properties (i.e. attributes and operations) of two classes to combine them into one class. In this case, the input specification for applying the "merge class" operator would be done by selecting the two classes that are to be merged.
- H. In some approaches that allow to specify the inputs for an operator in the model, it is not necessary to apply the operator immediately when specifying the inputs, whereas in other approaches, the input specification is inseparable from applying the operator. For example, drawing an association line

between two classes in UML specifies the inputs and immediately connects them with an association. In RAM on the other hand, it is possible to specify in an aspect model A that it instantiates an other aspect B without actually applying structural merge to combine the two models. The "aspect merge" operator can be applied, if the modeller wishes to visualize the combined structure of the two models, by hitting the "weave" button in the RAM tool.

4 Applying the Composition Criteria

In order to evaluate our proposed new definition, we applied the new composition criteria to the RAM composition operators, UML Association, UML Generalization and two of the AoUCM composition operators.

4.1 Description of Operators

In RAM we identified four composition operators: *aspect merge* (RAM AM) merges two aspect models by merging the structural view and copying the message and state views from the lower-level to the higher-level aspect; *message view inlining* (RAM MVI) merges the message views of all operation invocations inside a message view into the former; *message view advising* (RAM MVA) integrates all aspect message views that a message view is affected by into that message view; *state view composition* (RAM SVC) composes all state machines belonging to the same class (i.e., a state view including all state machines) using CSP || composition semantics.

UML Association and UML Generalization are examples explained in the questionnaire. The operator of *UML Association* (UML A) creates a relationship between two classes, visually by drawing a line but also by modifying the affected classes adding information to them. Similarly, the *UML Generalization* (UML G) operator creates a relationship between two classes, visually by drawing an arrow but also modifying the sub-class.

In the Aspect-Oriented Use Case Maps (AoUCM) approach, we evaluated the generic control flow constructs (which represent any UCM control flow) and the pointcut-advice-mechanism. In that context we identified the *control flow construct insertion* operator (AoUCM CFI) and the *aspect marker insertion* (AoUCM AMI) operator. *Control flow construct insertion* inserts a control flow construct into the model and connects it to those workflow model elements that are used as input for the control flow construct. *Aspect marker insertion* inserts aspect markers into a workflow at the points where the pointcut pattern was matched, and binds the in and out paths of the marker to the start and end points of the associated advice aspect map.

4.2 Questionnaire

Table 1 contains the answers to questions B-I (with the exception of E, since this question was already answered above) of our proposed updated questionnaire from section 3.

Table 1. The completed updated questionnaire for RAM, UML and UCM.

B.	**RAM AM**	**RAM MVI**	**RAM MVA**	**RAM SVC**	**UML A**	**UML G**	**AoUCM CFI**	**AoUCM AMI**
C.	all RAM	Message View	Message View	State View	UML Class Diagram	UML Class Diagram	AoUCM	AoUCM
D.	**RAM Aspect Merge**: $Aspect_1$ x $Aspect_2$ x $Aspect_2$Instantiation → $Aspect_1$' **RAM Message View Inlining**: $Message\ View_1$ x $Message\ View_2$ → $Message\ View_1$' (all within same model) **RAM Message View Advising**: Message View x Aspect Message View → Message View' (all within same model) **RAM State View Composition**: $State\ View_1$ x $State\ View_2$ → $State\ View_1$' (all within same model) **UML Association**: M x $Class_1$ x $Class_2$ → M' **UML Generalization**: M x $Class_1$ x $Class_2$ → M' **UCM Control Flow Construct Insertion**: M x Map Element1 x M x Map Element2 → M' **UCM Aspect Marker Insertion**: M x Map Element x **Before/After x Start Point/out-path of pointcut stub x End Point/in-path of pointcut stub** → M'							
F.	No	No	No	No	Yes	Yes	Yes	Yes
G.	Yes, instantiation	Yes, calls	Yes, affected by	No, **	No, *	No, *	No, *	No, *
H.	Yes, ***	Yes, ***	Yes, ***					
I.	No	No	No	No	Yes	Yes	Yes	Yes

* defined when operator applied
** user selection in the tool
*** when weaving requested by user

5 Summary

This paper proposes changes to the parts of the CMA comparison criteria document and questionnaire that address composition. The work was motivated by the fact that when we applied the existing criteria to our approach we found the part on composition confusing. Some definitions were vague and hence we each interpreted them differently. Particularly, we found that definition of composition rule was not clear and the criteria document acknowledged the ambiguity. Additionally, the examples given in the paper did not help, as they sometimes even contradict each other. To remedy the situation, we suggest a more detailed definition of composition operator, propose to remove the composition rule, and to use input specification instead. We redesigned the questionnaire to reflect this change and applied our proposed criteria to three modelling approaches, AoURN, RAM and UML. Future work could address additional questions to address relationships between operators, i.e., operators that use other operators, which could be either independent or dependent operators (sub-operators).

References

1. Modeling Approach Comparison Criteria for the CMA@RE Workshop at MODELS 2013, `http://cserg0.site.uottawa.ca/cma2013models/ComparisonCriteria.pdf`
2. Kienzle, J., Al Abed, W., Klein, J.: Aspect-Oriented Multi-View Modeling. In: AOSD 2009. pp. 87 – 98. ACM Press (March 2009)
3. Mussbacher, et al.: Assessing composition in modeling approaches. CMA, ACM (2012)
4. Mussbacher, et al.: Comparing six modeling approaches. MODELS, Springer (2012)
5. Mussbacher, G., Amyot, D., Araújo, J., Moreira, A.: Requirements Modeling with the Aspect-oriented User Requirements Notation (AoURN): A Case Study. In: Katz, S., Mezini, M., Kienzle, J. (eds.) Transactions on Aspect-Oriented Software Development VII, Lecture Notes in Computer Science, vol. 6210, pp. 23–68. Springer Berlin / Heidelberg (2010)