Validation challenges in model composition: The case of adaptive systems

Freddy Munoz, Benoit Baudry

▶ **To cite this version:**

Freddy Munoz, Benoit Baudry. Validation challenges in model composition: The case of adaptive systems. In Proceedings of ChaMDE 2000 - Workshop on Challenges in Model Driven Engineering in conjounction with MODELS'08, 2008, Toulouse, France, France. inria-00456505

HAL Id: inria-00456505
https://inria.hal.science/inria-00456505

Submitted on 15 Feb 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Validation challenges in model composition: The case of adaptive systems*

Freddy Munoz, Benoit Baudry

INRIA Bretagne Atlantique
Campus de Beaulieu F-35042, Rennes cedex
{fmunoz,bbaudry}@irisa.fr

Abstract. Model Driven Engineering helps dealing with complexity by promoting models as abstraction units. Aspect Oriented Modeling helps separating concerns that crosscut across different models. MDE and AOM have well identified challenges that need to be addressed. However, there are new challenges that appear when combining both techniques. In this paper we present the challenges that appear when validating the model composition in the context of MDE and AOM applied to adaptive systems.

Keywords: Model Driven Engineering, Aspect Oriented Modeling, Model Composition Validation, Model Validation, Model Driven Engineering for Adaptive Systems.

1 Introduction

Model Driven Engineering (MDE) promotes abstraction as a basis for managing complexity. MDE proposes the systematic use of models as primary engineering artifacts. Such models can have a variety of natures and range in abstraction and complexity. Models from higher abstraction level are refined (transformed) into lower levels until the implementation. Besides, models can be transformed from one domain to another in order to ease the resolution of a defined problem.

Aspect Oriented Modeling (AOM) helps separating crosscutting concerns at model level by encapsulating them into different modeling dimensions referred as *aspect*. AOM enables a clear modularization of the different concerns constituting a design model. It also allows designers to reason about each concern separately, and later composed into a global model.

Research challenges have been widely identified for AOM and MDE [11]. Model transformation testing[5], model verification and validation [9], and AOM weaving mechanism definition [2] are just a few examples of the challenges faced by these technologies.

Nevertheless, not all the challenges have been identified as far as validation in MDE and AOM is concerned. Challenges regarding the composition and refinement of aspect models need to be identified. This is critical to ensure that composed

* This work was partially funded by the DiVA project (EU FP7 STREP)

models will perform as expected, and therefore, their refined implementation will do so [4, 8]. This is a fundamental issue for the adoption of AOM as a mechanism for separation of concerns and MDE as a complexity coping mechanism.

In this paper we present the challenges that arise from the validation of model composition. We specially address the challenges that arise in the case of adaptive systems, where models are used to abstract from the executing platform and aspects represent the dynamic variability of the system. Such challenges range from the combinatorial explosion produced by the composition order of different aspects, to the specification of the model resulting from the composition.

The remainder of this paper is organized as follows. Section 2 presents MDE and AOM applied to adaptive systems. Section 3 presents the challenges that arise when validating AOM and MDE in the context of adaptive systems. Finally, section 4 concludes.

2 MDE and AOM for adaptive systems

Designing, developing, maintaining and executing adaptive systems is very complex and error prone. Model Driven and Aspect Oriented techniques can help dealing with this complexity. In this section we present the contribution of MDE and AOM to handle the complexity when dealing with adaptive systems.

Adaptive systems are software systems capable of change their internal structure and behavior in response to changes in their environment [1]. They are typically deployed in heterogeneous computing devices ranging from mobile devices such as phones or PDAs to large computer systems. Generally, several variation points are defined in order to develop an adaptive system. Each variation point represents a different option in the system implementation that might be chosen to adapt the system. The selection of different variation points to derive the adapted system leads to a huge number of possible configurations. Reasoning over that huge set of configurations to choose the best possible configuration to adapt is too time consuming because of the large number of evaluations needed. Moreover, the adaptation logic relies on reconfiguration policies that are generally complex low-level and hand-written in the application producing large and complex reconfiguration files. These factors make the construction, execution and maintenance of adaptive systems highly complex. MDE and AOM help dealing with this complexity by the meaning of abstraction and separation of concerns [19].

MDE techniques provide the means to automate and optimize the creation of reconfiguration scripts. Besides, MDE helps abstracting from the target platform by defining models independent of target devices and technologies. Models representing the system in execution (models at runtime) help to manage the execution at more abstract level; therefore, they enable designers to reason about the system properties and adaptation logic at higher level.

Aspect-Oriented modeling techniques [10, 13, 15, 18] help encapsulating distinct variation points into aspects separated from the base model functionalities. Different aspects might be composed with the base model in order to obtain different

configurations. This reduces the reasoning space to a limited number of aspects, therefore avoiding the combinatorial explosion due to different variants.

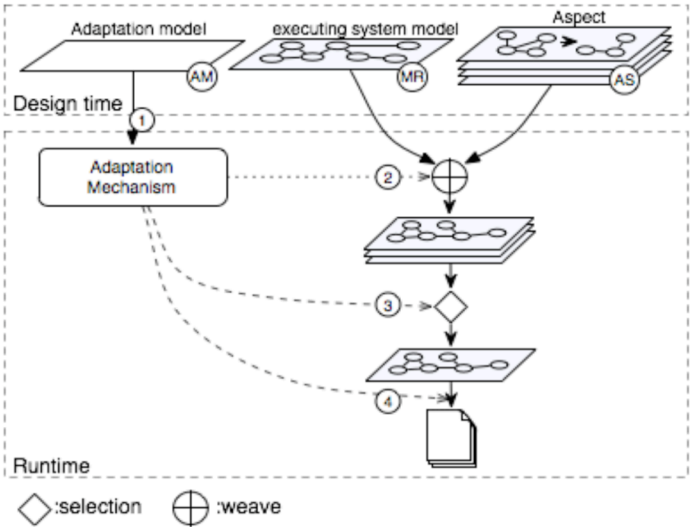

Fig. 1. Overall MDE/AOM approach for adaptive systems

Figure 1, presents an overall approach for adapting systems by using MDE and AOM techniques. At design-time, the application base (AM) and variant architecture (AS) models are designed. At this time, the adaptation model, which states when, and how to adapt is built. At runtime, the adaptation mechanism processes the adaptation model in order to adapt when needed (1). When an adaptation is required, the adaptation mechanism chooses (driven by the adaptation model) a set of aspects (variants) and weaves them into the base (2). This weaving results in multiple models that could be used to adapt the executing system. The adaptation mechanism chooses only one model (3) and then automatically generates the reconfiguration scripts used to adapt the executing system (4).

3 Challenges

MDE and AOM can help dealing with the complexity involved in the life cycle of adaptive systems. However, their usage raises new challenges regarding the validation of model composition. In the following we summarize the challenges and research questions related to the validation in the context of adaptive system.

Validation of composed models (3 in figure 1): The selection of the best possible configuration is critical for adaptive systems. The aspects modifying the base configuration must produce configurations that will not break down the system and that response in the best possible way to environmental changes . Therefore, it is crucial to ensure that the composed models fit the adaptation requirements. A model

that is valid with respect to the adaptation requirements will lead to a correct adaptation. Testing techniques such as combinatorial testing [12, 24] and search based testing [17] may help validating that the composed model fits the adaptation requirements. Such testing techniques provide the means to explore a huge adaptation space and test whether the chosen configurations are fitted adaptations. Formal behavioral specification techniques [25] may also be useful to verify that the composed models will fit the adaptation invariants.

It is an intuition to think that, if the chosen aspects are valid and the base model is valid, then the composed model will be valid. Is this always true? How can it be ensured? These questions are fundamental because they may allow to reason about the aspects and model validity separately and then compose a valid model. However, event if this is true, it is still an issue how to validate the aspect models and how to validate the base model. Moreover, the composition engine must also be valid in order to produce valid compositions.

Combinatorial explosion of composed models (2 in figure 1): The weaving of different aspects leads to different composed models. Likewise, the weaving order of aspects may also generate different models. Therefore, the rate of models resulting from the composition of different aspects and composition orders grows exponentially with the amount of aspects to weave. Moreover, there is no assurance that the composed models will be valid or fits the adaptation requirements. This is a serious issue because it is necessary to validate a huge amount of composed models. Such validation may consume an unrealistic amount of time.

Aspects effects and interactions (2, 3 in figure 1): Different aspects have different effects on the base model. Some aspects may add new system properties whereas others may remove them. The effect of aspects may depend on the order in which they are weaved. For instance, consider two aspects, one relating the communication (C) and another relating the security concerns (S). When C is woven first, the system network response is very short, whereas when S is woven first, the system network response is slower but more secure. Some weaving orders or combinations of aspects could add or remove system properties unexpectedly. Controlling the emergence of properties introduced/ removed by aspects is very important to ensure that the composition result will be valid and aspects will perform as expected. Similar problems have been studied at code level. Solutions such as the specification of the aspect behavior [3, 6, 20] are used to increase the maintainability of aspect-oriented programs. The properties added/ removed by the aspects are controlled by the specifications.

Aspects adapting a system will interact in a variety of ways. Some interactions may include/ exclude the weaving of some aspects; other interactions may interfere or partially invalidate the effect of aspects over the system. Moreover, interactions and the effect of aspects may change according to the target model they are weaved into. Therefore, detecting aspects interactions in advance is very important to avoid composition conflicts and know before hand the aspects dependencies [22, 23]. This issue is related to critical pair analysis [7, 16, 21] in which the conflicts between different interacting features are detected via graph analysis. Critical pair analysis detects functional inclusive/ exclusive aspects configurations. However, interaction

issues can be beyond functional interactions. Aspects can have a qualitative impact over the system, for instance making the quality of service better or worst. At code level, the characterization of interactions could be used to determine patterns of interactions for instance to detect aspect interferences [14].

Runtime / Design time validation: Since adaptation happens at runtime, adaptive systems have to respond to hard time and hardware constraints when adapting, a fundamental question is how much of the validation and analysis can be done statically? The ideal will be to calculate at design time *all* the possible interactions and effect of aspects and their possible weaving orders. However, this may not be possible due to the huge amount of possible weaving orders

An idea to cope with these issues is defining contracts on the aspect models. These contracts may be an abstract specification of the effect of the aspects over the system and the interactions between aspects. For instance, they can explicitly declare that an aspect will increase the overall system security but making it slower. They could also allow us to calculate optimal and valid weaving orders. By specifying include/ exclude relations between aspects. Moreover, they may be helpful to detect interactions conflicts at design time, thus saving some computation time when adapting at runtime. The abstract description of the aspects' effect will help determining whether aspects may be valid or not in relation to a base model.

4 Conclusions

In this paper we have identified the challenges that appear when validating the model composition in the context of adaptive systems. Issues such as the weaving order of aspect models, interaction issues and the validation of the composed model are not trivial. Tackling these issues is fundamental to assess the usage of MDE and AOM.

We have pointed out possible ways to address the validation challenges presented here. We specially suggest the definition of contracts to tackle several challenges and ease the solution of others. In future work we will explore how contracts must look like, which information they must contain and how to successfully use them at design time and runtime.

References

1. Aksit, M. and Z. Choukair, *Dynamic, Adaptive, and Reconfigurable Systems Overview and Prospective Vision*, in *23rd Int'l Conf. Distributed Computing Systems Workshops (ICDCSW)*. 2003: Providence, Rhode Island USA.
2. Aldawud, O., et al. *AOM: 11th International Workshop on Aspect-Oriented Modeling*. in *10th International Conference On Model Driven Engineering Languages And Systems*. 2007. Nashville, TN, USA.
3. Aldrich, J., *Open Modules: Modular Reasoning About Advice*, in *ECOOP 2005: 19th European Conference on Object-Oriented Programming*. 2005: Glasgow, UK.

4. Baleani, M., et al. *Correct-by-construction transformations across design environments for model-based embedded software development*. in *Design, Automation and Test in Europe, 2005. Proceedings*. 2005.
5. Baudry, B., et al. *Model Transformation Testing Challenges*. in *IMDT workshop in conjunction with ECMDA-FA 06*. 2006. Bilbao, Spain.
6. Clifton, C. and G.T. Leavens. *Observers and Assistants: A Proposal for Modular Aspect-Oriented Reasoning*. in *FOAL 2002: Foundations of Aspect-Oriented Languages (AOSD-2002)*. 2002. Enschede, The Netherlands.
7. Detlef, P., *Hypergraph rewriting: critical pairs and undecidability of confluence*, in *Term graph rewriting: theory and practice*. 1993, John Wiley and Sons Ltd. p. 201-213.
8. Dion, B., *Correct-by-Construction Methods for the Development of Safety-Critical Applications*. SAE transactions, 2004. **SAE Paper # 4AE-129**(SAE World Congress).
9. Faivre, A., S. Ghosh, and A. Pretschner. *MoDeVVA: 5th workshop on Model Driven Engineering, Verification, And Validation: Integrating Verification And Validation In MDE*. in *1st International Conference on Software Testing, ICST 2008*. 2008. Lillehammer, Norway.
10. France, R., et al., *Providing Support for Model Composition in Metamodels*, in *EDOC'07: 11th Int. Enterprise Computing Conference*. 2007: Annapolis, Maryland, USA.
11. France, R. and B. Rumpe, *Model-driven Development of Complex Software: A Research Roadmap*, in *2007 Future of Software Engineering*. 2007, IEEE Computer Society.
12. Grindal, M., *Handling Combinatorial Explosion in Software Testing*, in *Computer Science*. 2007, University of Skövde and Enea: Skövde, Sweden.
13. Jayaraman, P., et al., *Model Composition in Product Lines and Feature Interaction Detection Using Critical Pair Analysis*, in *Model Driven Engineering Languages and Systems*. 2007. p. 151-165.
14. Katz, S., *Diagnosis of Harmful Aspects Using Regression Verification*, in *FOAL: Foundations Of Aspect-Oriented Languages*, C.C.a.R.L.a.G.T. Leavens, Editor. 2004: Lancaster, UK.
15. Lahire, P., et al., *Introducing Variability into Aspect-Oriented Modeling Approaches*, in *Model Driven Engineering Languages and Systems*. 2007. p. 498-513.
16. Leen, L., E. Hartmut, and O. Fernando, *Efficient Conflict Detection in Graph Transformation Systems by Essential Critical Pairs*. Electron. Notes Theor. Comput. Sci., 2008. **211**: p. 17-26.
17. McMinn, P., *Search-based software test data generation: a survey*. Software Testing Verification Reliability, 2004. **14**: p. 105 -- 156.
18. Morin, B., O. Barais, and J.-M. Jézéquel. *Weaving Aspect Configurations for Managing System Variability*. in *Second International Workshop on Variability Modelling of Software-intensive Systems*. 2008. Essen, Germany.
19. Morin, B., et al. *An Aspect-Oriented and Model-Driven Approach for Managing Dynamic Variability*. in *11th International Conference on Model Driven Engineering Languages and Systems (MODELS)*. 2008. Toulouse, France
20. Munoz, F., B. Baudry, and O. Barais, *Improving Maintenance in AOP Through an Interaction Specification Framework*, in *ICSM08: 24th IEEE International Conference on Software Maintenance*. 2008, IEEE Computer Society: Beijing, China.
21. Praveen, J., et al., *Model Composition in Product Lines and Feature Interaction Detection Using Critical Pair Analysis*. Model Driven Engineering Languages and Systems, 2007: p. 151--165.
22. Sanen, F., et al. *Classifying and documenting aspect interactions*. in *ACP4IS06 Proceedings of the Fifth AOSD Workshop on Aspects, Components, and Patterns for Infrastructure Software*. 2006. Bonn, Germany.
23. Truyen, E., et al., *Support for distributed adaptations in aspect-oriented middleware*. Proceedings of the 7th international conference on Aspect-oriented software development, 2008: p. 120-131.
24. Yilmaz, C., M.B. Cohen, and A.A. Porter, *Covering Arrays for Efficient Fault Characterization in Complex Configuration Spaces*. IEEE Transactions on Software Engineering, 2006. **32**: p. 20--34.
25. Zhang, J. and B.H.C. Cheng, *Model-based development of dynamically adaptive software*, in *ICSE '06: Proceedings of the 28th international conference on Software engineering*. 2006, ACM: Shanghai, China. p. 371--380.