Preface

The design of modern software systems requires support capable of properly dealing with their ever-increasing complexity. In order to account for such a complexity, the whole software engineering process needs to be rethought and, in particular, the traditional division among development phases to be revisited, hence moving some activities from design time to deployment and runtime. Model-Driven Engineering (MDE) and Component-Based Software Engineering (CBSE) can be considered as two orthogonal ways of reducing development complexity: the former shifts the focus of application development from source code to models in order to bring system reasoning closer to domain-specific concepts; the latter aims to organize software into encapsulated independent components with well-defined interfaces, from which complex applications can be built and incrementally enhanced.

When exploiting these development approaches, numerous different modelling notations and consequently several software models are involved during the software life cycle. On the one hand, effectively dealing with all the involved models and heterogeneous modelling notations that describe software systems needs to bring component-based principles at the level of the software model landscape hence supporting, e.g., the specification of model interdependencies, and their retrieval, as well as enabling interoperability between the different notations used for specifying the software. On the other hand, MDE techniques must become part of the CBSE process to enable the effective reuse of third-party software entities and their integration as well as, generally, to boost automation in the development process.

An effective interplay of CBSE and MDE approaches could help in handling the intricacy of modern software systems and thus reducing costs and risks by: (i) enabling efficient modelling and analysis of extra-functional properties, (ii) improving reusability through the definition and implementation of components loosely coupled into assemblies, (iii) providing automation where applicable (and favourable) in the development process. In the last fifteen years, such a cooperation has been recognized as extremely promising; tools and frameworks have been developed for supporting this kind of integrated development process. Nevertheless, when exploiting interplay of MDE and CBSE, clashes arise due to misalignments in the related terminology but also, and more importantly, due to differences in some of their basic assumptions and focal points.

The goal of the workshop on Interplay of Model-Driven and Component-Based Software Engineering 2016 (ModComp'16) was to gather researchers and practitioners to share opinions, propose solutions to open challenges and generally explore the frontiers of collaboration between MDE and CBSE. ModComp'16 aimed at attracting contributions related to the subject at different levels, from modelling to analysis, from componentization to composition, from consistency to versioning; foundational contributions as well as concrete application experiments were sought.

The workshop was co-located with ACM/IEEE 19th International Conference on Model Driven Engineering Languages & Systems, and represented a forum for practitioners and researchers. We received twelve papers out of which six papers were selected for inclusion in the proceedings. The accepted papers covers many different forms of intertwining of MDE and CBSE including, but not limited to:

- model integration;
- model transformations for analysis and code generation;
- modeling component interactions for quality assessment;
- concern-oriented modeling of components;

– modeling for self-adaptive systems
– modeling languages for components.

This was the third edition of the workshop and the high attention received once again in terms of submissions proves that the topics are relevant both in practice and in theory of model-driven engineering of component-based software systems. Thus, we would like to thank the authors – without them the workshop simply would not have taken place – and the program committee for their hard and precious work.

September 2016 Federico Ciccozzi and Ivano Malavolta