Exploiting the ASM method within the Model-driven Engineering paradigm

Angelo Gargantini[1] Elvinia Riccobene[2] Patrizia Scandurra[3]

[1] Dipartimento di Ingegneria Gestionale e dell'Informazione
Università di Bergamo, Italy
angelo.gargantini@unibg.it
[2] Dipartimento di Tecnologie dell'Informazione
Università di Milano, Italy
riccobene@dti.unimi.it
[3] Dipartimento di Matematica e Informatica
Università di Catania, Italy
scandurra@dmi.unict.it

Model-driven Engineering (MDE) [3] is an emerging approach for software development and analysis where models play the fundamental role of first-class artifacts. Metamodelling is a key concept of the MDE paradigm and it is intended as a modular and layered way to endow a language or a formalism with an abstract notation, so separating the abstract syntax and semantics of the language constructs from their different concrete notations. Although the foundation constituents of the MDE approach are still evolving, some implementations of the MDE principles can be found in environments like the OMG MDA (Model Driven Architecture) [6], Model-integrated Computing (MIC) [7], Software Factories and their Microsoft DSL Tools [1], etc.

We believe that the MDE paradigm can gain rigour and preciseness, necessary expecially for model analysys purposes, from the integration with formal approaches. We, in particular, try to address here the problem of defining the operational semantics of metamodel-based languages in the MDE approach.

Currently, metamodelling frameworks lack of a way to specify the semantics of languages. The OMG metamodelling framework, for example, provides two main standard techniques to define languages: *metamodels* and *UML profiles*. Building metamodels is a first-class extension mechanism which requires the definition of a new metamodel or an extension of an existing metamodel based on the MOF (Meta Object Facility) metalanguage - i.e static class diagrams and well-formedness rules written in the OCL (Object Constraint Language)[8]. The UML profile mechanism is a non-first-class extension mechanism, i.e. it does not allow for modifying the existing UML metamodel, but it requires the specification of UML extension elements (stereotypes and tagged values) and the definition of new constraints (well-formedness OCL rules) specific to a target domain. For both extension techniques, the way to define the abstract syntax (i.e. the MOF-compliant metamodel) and static semantics (the OCL constraints) of a new language are well established. However, the OMG framework does not have any standard and rigorous support to provide the dynamic (operational) semantics, which is usually given in natural language.

Dagstuhl Seminar Proceedings 06191
Rigorous Methods for Software Construction and Analysis
http://drops.dagstuhl.de/opus/volltexte/2006/638

This lack has several negative consequences, as often signaled in the past for the UML metamodel [12] since its first version, and confirmed by several works existing in literature which aim at formalizing the UML semantics. Integrating formal approaches into metamodelling frameworks could result in a unified methodology for reusable syntax and semantics definitions of metamodels.

We explain how the formal framework of ASMs [4] and the ASM metamodel definition (AsmM) [2] can be smoothly integrated with the OMG framework for MDE to rigorously define the operational semantics of metamodel-based languages and, in particular, of UML extensions (UML profiles), in a way which permits us to uniformly link abstract syntax (expressed in the MOF meta-language) and detailed semantics (expressed in the ASM meta-language) of languages. We apply the proposed methodology to a UML profile for the SystemC language [11] - involved in a model-based SoC (System-on-chip) design flow for embedded systems development [5] - to define the operational semantics of the SystemC Process State Machines [10, 9], an extension of the UML statechart formalism used to model the reactive behaviour of the SystemC thread processes. We want to remark that, although the approach was first identified for the OMG's MDA framework, it could be easily extended and applied to other metamodelling frameworks.

References

1. Microsoft DSL Tools in Visual Studio 2005. `http://lab.msdn.microsoft.com/teamsystem/workshop/dsltools/`.
2. The Abstract State Machines Metamodel (AsmM) website. `http://www.dmi.unict.it/~scandurra/AsmM/`.
3. Jean Bézivin. On the Unification Power of Models. *Software and System Modeling (SoSym)*, 4(2):171–188, 2005.
4. E. Börger and R. Stärk. *Abstract State Machines: A Method for High-Level System Design and Analysis*. Springer Verlag, 2003.
5. E. Riccobene and P. Scandurra and A. Rosti and S. Bocchio. A SoC Design Methodology Based on a UML 2.0 Profile for SystemC. In *Proc. of Design Automation and Test in Europe (DATE05)*. IEEE Computer Society, 2005.
6. OMG. The Model Driven Architecture (MDA). `http://www.omg.org/mda/`.
7. Model Integrated Computing (MIC). `http://www.isis.vanderbilt.edu/Research/mic.html#MIC`.
8. OMG. UML 2.0 OCL Specification, ptc/03-10-14.
9. E. Riccobene and P. Scandurra. Modelling SystemC Process Behaviour by the UML Method State Machines. In *Proc. of Rapid Integration of Software Engineering Techniques (RISE'04)*, volume 3475 of *LNCS*. Springer, 2005.
10. E. Riccobene, P. Scandurra, A. Rosti, and S. Bocchio. A UML 2.0 profile for SystemC: toward high-level SoC design. In *EMSOFT '05: Proceedings of the 5th ACM international conference on Embedded software*, pages 138–141. ACM Press, 2005.
11. T. Gröetker and S. Liao and G. Martin and S. Swan. *System Design with SystemC*. Kluwer Academic Publisher, 2002.
12. OMG. The Unified Modeling Language (UML). `http://www.uml.org`.