Towards Model-driven development of Embedded SoC with UML and SystemC *

Patrizia Scandurra

Dipartimento di Mat. e Inf. – Università di Catania
V.le A. Doria, 6 - 95125 Catania, Italy
scandurra@dmi.unict.it

The increasing complexity of most real-time and embedded systems coupled with requests for more performance and shorter time to market have produced a high interest to investigate advanced design methods and technologies that could simplify and improve the reliability of embedded software design and implementation, while promoting the reuse of software and hardware components for a co-design system development.

From the first versions of UML [11], UML 1.x, designers of real-time and embedded systems have taken to UML with enthusiasm. However, the dream of a complete, model-driven design flow from specification through automated, optimized code generation, has been difficult to realize without some key improvements in UML, specifically targeted to the real-time systems problem. With the enhancements in UML near standardization with UML 2.0, part of these improvements have been made, other are still to be achieved.

This paper aims to describe how in the specific domain of embedded System-on-Chip (SoC), lightweight modeling methods, like UML, and some necessary UML extensions (*UML profiles*), can be exploited in accordance with the *Model-driven architecture* (MDA)[6] – a framework for Model Driven Development (MDD) – design principles to improve the current SoC design flow.

A SoC design methodology (see also [3, 1]) which involves (i) UML 2.0, a UML profile for the SystemC language [8], and some other UML profiles for lower level programming languages is presented. The UML in a platform-independent manner is used as schematic entry to provide a first specification of the system at *functional level*. At this point, the partitioning between hardware and software components – dictated by the sense and experience of expert engineers – is reflected at UML level and two different design flows start for the software and the hardware parts respectively. The UML profile for SystemC is used for the hardware description at several abstraction layers (*functional untimed/timed*, *transactional*, BCA, RTL) before the final *synthesis*, while UML profiles tailored for programming languages like C/C++, Real-time Java, etc. are used, instead, for refining the software part.

Using SystemC to connect system level SoC design flow to consolidated VLSI design flows is a well known issue. What is innovative is the idea to rely on low-cost customized (by a standard profiling technique) UML 2.0 modeling tools

* This work has been partially supported by the project *Tecniche e metodologie di progetto, documentazione, verifica e validazione per i sistemi di IP (Intellectual Property)* at STMicroelectronics.

as front-ends of lower level system design frameworks, and on provided tools which allow through automatic *model transformations* – a key point of MDA – to switch to a new technology while re-using the old designs. The ultimate goal is to raise the level of abstraction, and to develop more complex systems by manipulating models only.

This separation of models, however, demands more support for *model evolution activities* such as code generation, reverse engineering, model refinement, model refactoring, model inconsistency management, etc. Today, only limited support is available in Model-driven development tools for these evolution activities [9].

We believe that a rigorous foundation to specify models and their evolution with respect to the reuse principle could be realized by the synergies of: (i) MDA, whose efforts until now have been focusing on achieving platform independence allowing the definition of domain specific languages and the construction of low-cost tools; (ii) *Generative software development*, which focuses on automating the creation of *system-family members* (also known as *product-line members*) addressing both technical and application-domain variability [2]; (iii) *Aspect-oriented development* [5] technologies, whose some benefits in the real-time domain included more modular code, lower development costs, and better real-time predictability, have been already proven [7, 10, 4].

References

1. S. Bocchio, E. Riccobene, A. Rosti, and P. Scandurra. A SoC design flow based on UML 2.0 and SystemC. In *International DAC Workshop - UML-SoC05*.
2. Krzysztof Czarnecki and Ulrich W. Eisenecker. *Generative Programming: Methods, Tools, and Applications*. Addison-Wesley, 2000.
3. E. Riccobene, P. Scandurra, A. Rosti, 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.
4. Aniruddha Gokhale and Jeff Gray. An Integrated Aspect-oriented Model-driven Development Toolsuite for Distributed Real-time and Embedded Systems. In *Proc. of the AOSD Workshop on Aspect-Oriented Modeling Workshop*, 2005.
5. G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. Lopes, and J.-M. Loingtier. Aspect-oriented programming. In *Proc. of the ECOOP, LNCS 1241*. Springer-Verlag, 1997.
6. OMG. The Model Driven Architecture (MDA). http://www.omg.org/mda/.
7. W. Pree and J. Templ. Aspect-oriented hard real-time programming and tool integration. In *The Monterey Workshop Series (Z. Manna, T Henzinger, eds.)*, 04.
8. T. Gröetker, S. Liao, G. Martin and S. Swan. *System Design with SystemC*. Kluwer Academic Publisher, 2002.
9. S. Ducasse S. Demeyer R. Hirschfeld T. Mens, M. Wermelinger and M. Jazayeri. Challenges in software evolution. In *Proc. of the International Workshop on Software Evolution*. IEEE, 2005.
10. A. Tesanovic, D. Nyström, J. Hansson, and C. Norström. *Aspects and Components in Real-Time System Development: Towards Reconfigurable and Reusable Software*. Cambridge International Science Publishing, 2004.
11. OMG. The Unified Modeling Language (UML). `http://www.uml.org`.