An Executable Semantics of the SystemC UML Profile

Elvinia Riccobene and Patrizia Scandurra

1 DTI Dept., Università degli Studi di Milano, Italy
elvinia.riccobene@unimi.it
2 DIIMM Dept., Università degli Studi di Bergamo, Italy
patrizia.scandurra@unibg.it

Abstract. The SystemC UML profile is a modeling language designed to lift features and abstractions of the SystemC/C++ class library to the UML level with the aim of improving the current industrial System-on-Chip design methodology. Its graphical syntax and static semantics were defined following the “profile” extension mechanism of the UML metamodel, while its behavioral semantics was given in natural language. This paper provides a precise and executable semantics of the SystemC Process State Machines that are an extension of the UML state machines and are part of the SystemC UML profile to model the reactive behavior of the SystemC processes. To this purpose, we used the meta-hooking approach of the ASM-based semantic framework, which allows the definition of the dynamic semantics of metamodel-based languages and of UML profiles.

1 Introduction

The SystemC UML profile [28,24] is a modeling language developed to improve the conventional industrial Systems-on-Chip (SoC) design methodology with a model-driven approach [25,26,27]. It is a consistent set of modeling constructs designed to lift both structural and behavioral features (including events and time features) of SystemC [32] to the UML [33] level. It was defined by exploiting the UML profile mechanism that requires the specification of UML extension elements (stereotypes and tagged values) and of new constraints as Object Constraint Language (OCL) [22] rules.

The profile, while providing a complete description of the modeling syntax and static semantics, suffers from the lack of a precise behavioral semantics that is given in natural language. Indeed, in the OMG framework used to define the profile, as well as in other metamodeling environments (like Eclipse/Ecore, GME/MetaGME, AMMA/KM3, XMF-Mosaic/Xcore, etc.), the way to define the language abstract syntax in terms of a metamodel and its static semantics as OCL rules is well established, while no standard and rigorous support is given to provide the dynamic semantics that is usually expressed in natural language. This lack has negative consequences, as often remarked in the past since the first UML version. Moreover, defining a precise semantics of UML extensions is widely felt, especially now that UML is turning into a “family of languages” (see the OMG standardization activities of UML profiles in [33]).

* This work is supported in part by the PRIN project D-ASAP (Dependable Adaptable Software Architecture for Pervasive computing).
The definition of a means for specifying rigorously the semantics of UML profiles, as well as of metamodel-based languages, is therefore an open and crucial issue in the model-driven context.

In [11], a formal semantic framework based on the ASM (Abstract State Machine) formal method [2] is presented, which allows us to express a precise and executable semantics of metamodel-based languages using different techniques. We here adapt one of the techniques in [11], the meta-hooking, for UML profiles, and we show its application to the SystemC UML profile. This implies to provide a rigorous semantics of the SystemC Process (SCP) state machines formalism of the SystemC UML profile used to model the reactive behavior of SystemC processes.

This paper is organized as follows. Some background on the SystemC UML profile is given in Sect. 2. Sect. 3 presents the meta-hooking technique of the ASM-based semantic framework. Sect. 4 shows the application of the meta-hooking technique to the OMG metamodeling framework for the semantics specification of the SCP state machines. Some related work is presented in Sect. 5, while Sect. 6 concludes the paper.

2 The SystemC UML Profile

SystemC [32] is an open standard in the EDA (Electronic Design Automation) industry. Built as C++ library, SystemC is a language providing abstractions for the description and simulation of SoCs. Typically, the design of a system is specified as a hierarchical structure of modules and channels. A module is a container class able to encapsulate structure and functionality of hardware/software blocks, while a channel (primitive or hierarchical) serves as a container to encapsulate the communication functionality of blocks. Each module may contain attributes as simple data members, ports (proxy objects) for communication with the surrounding environment and processes for executing module’s functionality and expressing concurrency in the system. Fig. 1 shows a module example, count_stim, containing a thread process stimgen, two input ports dout and clock, and two output ports load and din, in the SystemC UML profile.

We here skip the details concerning the structural modeling constructs, as the focus is on the behavioral aspects of the profile. Some basic concepts underlying the SCP state machines are reported below as defined in the SystemC UML profile [28]. This formalism is to be considered a conservative extension of the UML method state machine defined through the UML extension mechanism of “profiles” (i.e., stereotypes, tags, and constraints) [33].

SystemC Process State Machines: Processes are the basic unit of execution within SystemC and provide the mechanism for simulating concurrent behavior. Three kinds of processes are available: sc_method, sc_thread and sc_cthread. Each kind of process has a slight different behavior, but in general (i) a process is declared within the scope of a class (a module or a hierarchical channel) as a stereotyped operation with no return type and no arguments (see, for example, Fig. 1); (ii) all processes run

1 A UML “method” state machine specifies the algorithm or procedure for a behavioral feature (such as a class’s operation).