Simulation Modelling Practice and Theory 38 (2013) 115–131

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Modeling and simulation-driven development of embedded real-time systems

Mohammad Moallemi *, Gabriel Wainer

Systems and Computer Engineering Department, Carleton University, 1125 Colonel by Dr., Ottawa K1S 5B6, Canada

ARTICLE INFO

Article history:
Received 5 September 2012
Received in revised form 22 July 2013
Accepted 28 July 2013
Available online 30 August 2013

Keywords:
Discrete-event simulation
Embedded systems
Real-time simulation and control
Model-based approach

ABSTRACT

The design and development of embedded hard real-time (RT) systems is one of the complex development practices, because of the requirements of criticality and timeliness of these systems. One critical aspect of RT systems is the production of output before specified deadline. Formal methods are promising in dealing with the design issues of these applications, although they do not scale well for complex systems. Instead, Modeling and Simulation (M&S) provides a cost-effective approach to verify the design and implementation details of very Complex RT applications. M&S methods provide dynamic and risk-free testing environments to verify different scenarios, and they are used for feasibility analysis and verification of such systems. Nevertheless, the simulation models are usually discarded in the later phases of the development.

We present the application of an M&S-based method referred to as DEVSRT (Discrete EVent System Specifications in Real-Time) to solve the discontinuity between the simulation models and the final embedded application, in this paper. DEVSRT defines explicit deadline notation for DEVS transitions, draws a clear mapping between DEVS transitions and real-time tasks and provides a formal method and tool for integration of simulation models with the associated hardware components.

1. Introduction

Real-time (RT) and embedded systems are employed in various applications ranging from telecommunications, customer electronics, transportation, medical equipment, and automated systems. An RT system is formally defined by Liu as "a system that is required to complete its work and deliver its services on a timely basis" [1]. The system in which all timing constrains must be met are considered "hard real-time systems". Real-time application development has evolved rapidly because of the growing use of these applications. Many of these systems are deployed in embedded controllers working in hardware computing platforms with special configurations and interfaces. An "embedded system" is described by Steve Heath as "a microprocessor-based system that is built to control a function or a range of functions and is not designed to be programmed by the end user in the same way that a PC is" [2]. In the case of embedded systems with hard real-time constraints, the design decisions can lead to catastrophic consequences for infrastructures or lives [3]. The size, variety, and criticality of the computations carried out on these systems have attracted creative and visual design methods that increase their complexity, reliability and performance. The architecture of these systems usually integrates different types of hardware components such as processors, analog and digital components, as well as mechanical (e.g. sensors and actuators) and visual components, which demands increasingly challenging multidisciplinary design and development efforts [4].

* Corresponding author. Tel.: +1 3862267972.
E-mail addresses: moallemi@sce.carleton.ca (M. Moallemi), gwainer@sce.carleton.ca (G. Wainer).

1569-190X/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.simpat.2013.07.004

Nevertheless, because of the heterogeneity of these systems and their constraints (such as cost, time to market, and performance), their development cycle is still time consuming, error prone and expensive.

A solution that provides a correct framework for designing these systems is the adoption of formal methods [5], described as special cases of mathematical-based techniques for designing, development and verification of software and hardware systems [4]. They allow for appropriate mathematical specification and analysis of the designs, which can contribute to the reliability of the final product, yet they add to the complexity of the design by applying mathematical-based approaches, which makes it hard to prove, hence increasing the overhead of the development. Adding this overhead might increase the overall cost, nevertheless in some cases, spending these resources would result in an overall gain in the quality and cost of the final product. They are appropriate for systems with critical applications where safety and robustness is a foremost aspect. Nevertheless, these methods do not scale up well, as most formal proving mechanisms cannot provide formal proofs of correctness when the complexity of the system grows.

Instead, Modeling and Simulation (M&S) provides a practical solution in solving the above-mentioned difficulties. M&S is a useful tool for efficient analysis, design, verification and optimization of general dynamic systems. The use of M&S in software engineering reduces costs and risks, and allows for exploring different aspects of the system in a risk-free environment.

Formal M&S uses mathematical models to define the specifications of a system. This approach has shown promising results in making multidisciplinary system development tasks manageable [6]. Model-based design is a computational approach that provides a hierarchical design scheme in which higher abstract levels are branched into levels that contain more details; the system specifications are defined in a formal way at which new and distinguishing functionalities are conceived. Other advantages of Model-based development are applying formal model checking techniques at design time [7,8], the incremental refinement of the initial simulation models, the simulation-based validation and reuse of the existing models, the risk free testing of critical RT applications, and the increased reliability in the design. As discussed earlier, formal approaches as model checking add to the complexity of the design, and on the other hand, it increases the reliability and robustness of the final product. Formal M&S complements and enhances the pure formal method-based approaches, providing an alternative for the design of real-time and embedded systems.

The use of these techniques is often used in the early stages of the development of real-time and embedded systems. However, when the scope of the development moves towards the final architecture to be deployed on the actual target hardware, the early simulation models are abandoned, and the final system is redeveloped from scratch based on the results obtained during the simulation phase. Currently, existing development tools and methods do not support a simulation-based approach or they lack features providing model continuity from the requirement analysis stage of software development up to deployment into the target hardware [9]. Model continuity shortens the development process and speeds up the implementation phase. In the approach, we will show how M&S is a foremost component in the RT application development, and it goes further by utilizing the simulation models on the target platform. Although commercial tools like MATLAB/Simulink [10] and LabVIEW [11] provide good resources for these activities, they are mostly limited to simulation only and they do not directly support model continuity or model checking. On the other hand, approaches like UML-RT [12] can be used to develop software design models, or they are not suitable to be used as simulation models [13]. Also, they do not provide any means for modeling the environment surrounding the RT embedded application, while M&S-based methods do.

We introduce the ideas and applications of DEVSRT (Discrete-Event Systems Specifications in Real-Time), a domain extension to DEVS theory [6] for embedded real-time application development. DEVS provides a formal foundation to M&S that proved to be successful in different complex applications [3]. DEVSRT takes advantage of well-defined M&S properties and constructs of DEVS to design and interface embedded systems with the hardware and the environment under study. The DEVSRT approach provides the following advantages:

- The notion of a deadline is added to the DEVS formalism, making it appropriate for real-time system modeling and design. Based on DEVS computational properties, a set of assumptions is defined to be used in the design of a real-time application. DEVSRT uses DEVS formal outputs as the output signals of the real-time system, therefore a relative deadline is associated with each output produced at the end of each state.
- An efficient interfacing mechanism is added to DEVS theory, in order to satisfy our research motivations: (1) Model continuity from the simulation stage, up to the deployment in the target hardware. (2) The entire system is designed in a hardware-software co-design approach, in which the models represent different hardware and software components and are tested together as an integrated DEVS model. (3) Provides a hardware-in-the-loop simulation platform where some of the models act as the simulated plant components (in which the driver interfaces provide the electrical emulator signals) and are tested with the embedded system (controller) model to be deployed on the hardware.
- The Embedded CD++ tool (E-CD++) [14] is extended to implement the proposed DEVSRT framework for the formal development of embedded real-time applications. The new version of E-CD++ is implemented on a real-time kernel, incorporating real-time services.

We show how DEVSRT satisfies the motivations and present the real-time specifications added to the DEVS formalism to develop real-time applications. DEVSRT has been used to develop various real-time embedded systems on a variety of