Test-Driven Agile Simulation for Design of Image Processing Systems

Anna Yupatova, Vitali Schneider, Winfried Dulz, Reinhard German
Friedrich-Alexander-University of Erlangen-Nürnberg, Germany
Department of Computer Science 7
{anna.yupatova, vitali.schneider}@fau.de {dulz, german}@cs.fau.de

***Abstract*—Image processing systems are characterized by very high computational demand caused by large amounts of data, short response times, and the complexity of image processing tasks. For these reasons, specialized hardware solutions based on multiple processing cores, complex interconnects, or custom hardware elements are used for image processing. Due to the complexity of the computational tasks, the complexity of the specialized hardware solutions is continuously increasing. Therefore, new design techniques that reduce the risk of lacks in the system design or of expensive design-to-implementation iterations are desirable. Test-driven Agile Simulation (TAS) is a general-purpose approach that combines novel model-based simulation and testing techniques to achieve an improved overall quality for the development process. In this paper, we present an application and extension of the TAS approach for the efficient design process of image processing systems.**

Keywords–Test-driven agile simulation, model-based engineering, UML, SysML, MARTE, UTP, image processing systems

I. INTRODUCTION

Image processing systems (IPS) play an increasingly important role in our daily life with applications for medical diagnosis, remote sensing, or fingerprint recognition. In general, image processing tasks have very high computational demands. Due to continuously changing requirements, decreasing times and physical restrictions, IPS are becoming more heterogeneous resulting in the distribution and deployment of computational tasks on different processors and programmable logic units ([1][2]). Without the usage of effective design tools and development techniques, the realization of a complex heterogeneous IPS is difficult to carry out. This leads to a decreasing quality of the development process and finally to deficient systems.

Test-drive Agile Simulation (TAS) is an agile approach that improves the overall quality of the development process and reduces the design and development costs, while the reliability of possibly complex implementations increases due to early validation techniques. The main focus of TAS is on constructing the models that allow to detect design errors or inconsistencies in a system specification as early as possible by simulating the given system and executing test cases at the model level. In our work, we deploy the *VeriTAS* [3] framework that supports the automation of consecutive steps of the TAS approach. Thus, the simulation and testing are applied in earlier stages of a development process which also increases it's agility.

To construct models, TAS provides a modeling methodology [4] based on the UML (Unified Modeling Language) and applies multiple extension profiles, such as System Modeling Language (SysML) [5], Modeling and Analysis of Real-time and Embedded systems (MARTE) [6], and UML Testing Profile (UTP) [7]. Due to the utilization of a general-purpose modeling language, the provided modeling methodology is not limited to a specific application domain. However, in the context of heterogeneous image processing systems, one has to deal with typical image processing peculiarities and issues like modeling of image processing pipelines or the distribution of computational tasks on different hardware components. In this paper, we will explain the utilization of the TAS approach in the context of heterogeneous image systems.

II. OVERVIEW OF THE GENERAL DESIGN FLOW

The TAS approach structures the design process into the modeling of requirements, of high-level and refined system specifications as well as of test specifications using UML and a set of extension profiles (see Figure 1). Starting with the

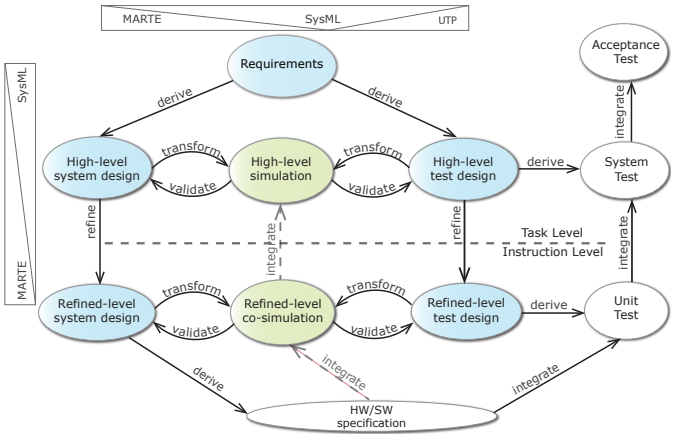

Figure 1: General design flow of the TAS approach.

common requirements specification, our approach is deriving system and test models in the subsequent steps independently from each other in order to ensure their mutual validation. In the high-level specification phase, the modeling is focused on the functional description of a system and corresponding tests. The functional models may be enriched in the subsequent refined specification phase with finer implementation or technical details of the functionality or of additional components for the hardware/software co-design.

Based on the developed models, our approach supports functional verification and performance analysis using simulation-based techniques at different abstraction levels. Prior to the expensive implementation and testing on the real hardware, the simulation of modeled system as well as the simulated execution of test cases allow the validation of the designed system with the specified tests at more abstract levels. Defects found in system specifications during an agile simulation step are easier to correct in the models than in the derived source code and corresponding test scripts.

 ISBN: 978-1-61208-370-4

III. Overview of the Design Flow for Image Processing Systems

The general modeling methodology for TAS is based on a combined subset of SysML, MARTE and UTP. However, the standard UML profiles do not provide enough semantics for the effective design process in the image processing domain. In order to introduce the required semantics in a specific application domain, the definition of a Domain-Specific Language (DSL) offers a feasible solution. However, the usage of DSL involves the additional effort to learn a new modeling language as well as to design and to maintain domain-specific model editors. Therefore, we are working on an UML-based library for image processing systems modeling named Lib4IPS. The library provides a pre-characterization of usable components for building models in the image processing domain. Starting with a detailed analysis of the IPS domain, we extract the most important characteristics and major tasks of image processing applications. The common procedure scheme in such applications is called image processing pipeline and is based on a similar workflow, namely: (1) image acquisition, (2) image pre-processing using local operators, (3) image processing using global operators, and finally (4) image post-processing using complex operators. For example, local pre-processing operators are used to perform an image filtering like the smoothing or edge detection [8]. In addition, the operators can be divided into groups depending on specific criteria like linear or non-linear filtering. According to this structure Lib4IPS performs the classification and characterization of typical image operations by means of UML. For each operator in the image processing pipeline, we define metrics that are important for application design and for performance analysis issues in the later steps of the development process. These metrics include computation cycles, number of instructions, and memory usage. Furthermore, the library holds specific image processing elements, i.e.: bitmap, vector graphics, pixels or image resolution.

Early validation and testing of system properties may have a deep impact on the performance of the final implementation. Hence, we utilize the general design flow of the TAS approach (see Section II) and adjust it for the IPS design and development process. Based on the requirements, a designer needs to devise system design at a high abstraction level, as illustrated in Figure 2. This level is independent of any hardware platform and application details. Its sole purpose is to describe the systems's functional behavior and to provide an initial performance validation step. The refined-level system design distinguishes between the application, the hardware architecture, and a possible mapping step. The hardware architecture model defines platform resources and captures resource acquisitions, performance and timing constraints. The application model describes the functional behavior of the image processing application in an architecture-independent manner. The allocation relationships involve the mapping of the application model onto the hardware architecture model, after which the system model is able to be validated quantitatively. For validation and testing purposes, we combine two simulation frameworks - *OMNeT++* [9] and *SystemC* [10]. The first simulation framework facilitates the system simulation at a high-level of abstraction with the focus on component's communication interfaces. The second simulation framework enables the simulation of image processing application on hardware components. Iterative simulation and test on the model level supports the planning and customization of system specification in order to achieve a desired quality. Furthermore, the approach helps identifying architectural bottlenecks, taking well-founded design decisions as well as finding ways for optimization of the application's execution efficiency.

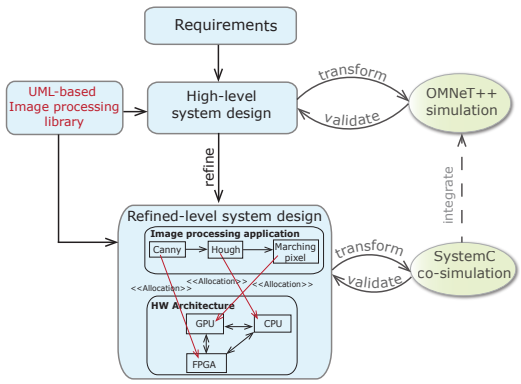

Figure 2: Design flow for image processing systems.

IV. Conclusion and Future Work

In this paper, we explain the application of the TAS approach for the image processing domain that assists a designer in the development of models for the system specification and analysis purposes. We extend the TAS approach by implementing the specific UML-based library which provides required semantics encountered in the image processing domain. In one of our next research steps, we are going to design a result back-tracing strategy for our approach, that will help developers to get simulation results directly into their designed models.

References

[1] D. G.Bailey, "Design for embedded image processing on FPGAs." John Wiley&Sons (Asia), 2011.

[2] I. K. Park, N. Singhal, M. H. Lee, S. Cho, and C. W. Kim, "Design and Performance Evaluation of Image Processing Algorithms on GPUs," IEEE Transactions on Parallel and Distributed Systems, 2011, pp. 91–104.

[3] A. Djanatliev, W. Dulz, R. German, and V. Schneider, "VeriTAS - A Versatile Modeling Environment for Test-driven Agile Simulation," in Proc. of the 2014 Winter Simulation Conference, Phoenix, AZ, USA, December 2011, pp. 3657–3666.

[4] V. Schneider, A. Yupatova, W. Dulz, and R. German, "How to Avoid Model Interferences for Test-driven Agile Simulation based on Standardized UML Profiles," in Proc. of the Symposium on Theory of Modeling and Simulation, Tampa, FL, USA, April 2014, pp. 540–545.

[5] F. Sanford, M. Alan, and S. Rick, "A practical guide to SysML." Morgan Kaufmann, 2012.

[6] S. Bran and S. Gerard, "Modeling and Analysis of Real-Time and Embedded Systems with UML and MARTE." Morgan Kaufmann, 2014.

[7] Object Management Group (OMG), "UTP: UML Testing Profile 1.2." [Online]. Available: http://omg.org/spec/UTP (retrieved October 2014)

[8] M. Petrou and C. Petrou, "Image processing: The Fundamentals," John Wiley&Sons Ltd, 2011.

[9] "OMNeT++ Network Simulation Framework." [Online]. Available: http://www.omnetpp.org/ (retrieved October 2014)

[10] T. Grotker, S. Liao, G. Martin, and S. Swan, "System Design with SystemC." Kluwer Acad. Publ., 2004.

 ISBN: 978-1-61208-370-4