ProMARTES Toolkit for Cycle-accurate Performance Analysis of Real-Time Distributed Systems

Konstantinos Triantafyllidis, Egor Bondarev, Peter H.N. de With
Eindhoven University of Technology, 5600 MB, Eindhoven, The Netherlands
Email: {k.triantafyllidis, e.bondarev, p.h.n.de.with}@tue.nl

***Abstract*—This paper presents the ProMARTES toolkit for profiling, modeling and performance analysis of component-based real-time distributed systems (CB-RTDS). The ProMARTES toolkit features multiple benefits: (a) cycle-accurate profiling of individual components and automated generation of MARTE-compatible SW component performance models, (b) guided composition of a system architecture from available SW and HW components based on the UML-MARTE profile, (c) automated generation of a performance model, (d) comprehensive performance analysis (scheduling, simulation and network analysis) of the composed system. In the ProMARTES toolkit, available as open source [1], the individual tools are integrated into the toolkit pipeline, providing a complete design and analysis life-cycle.**

I. Introduction

The development of component-based real-time distributed systems (CB-RTDS) has become an adopted practice, enabling system prototyping at the early design phase based on existing HW and SW components. However, the composed system should meet predefined performance requirements, such as throughput, latency and robustness. Thereby, accurate assessment methods predicting the performance at early design phases are required. Moreover, the assessment methods should perform a fast analysis, reducing the processing delay of the Design Space Exploration (DSE).

The ProMARTES toolkit has been developed to achieve sufficient performance-prediction accuracy and speed. For the component-development phase, ProMARTES offers profiling of the SW components, at cycle-accurate level, and automated generation of their performance models. Moreover, the toolkit facilitates graphical composition of a system architecture from available SW and HW components, and, then automatically translates the graphs into a system performance model. Finally, the toolkit provides both scheduling and simulation-based analysis, and applies both techniques for predicting the network and computation delays.

ProMARTES toolkit (see Fig.1) contains three principal modules: (a) profiling and modeling, (b) modeling of system architecture and automated generation of system models, and (c) performance analysis, treating both computation and network delays as first-class citizens.

II. Profiling and Modeling

The first module (ProMo tool) enables profiling of individual SW components [2]. The ProMo tool allows profiling of SW components at cycle-accurate level and automatically generates performance models based on the obtained measurements. For every function of a provided interface, ProMo profiles the following performance metrics: (a) *Instructions* and *cycles* used, (b) *Cache-miss-rate* for L2 and L3 cache levels, (c) amount of memory *claimed* and *released*, and (d) *bus load* due to read/write instructions.

The ProMo performance model of an individual SW component is further transformed to a MARTE-compatible resource model by the Promo2Marte meta-modeling tool (see Fig. 1). We have extended the MARTE profile to support the cycle-accurate performance metrics obtained by the ProMo tool. The generated resource models are then placed, alongside with the corresponding SW components, into a repository, and can be loaded by the Architecture Composition module, once an architect selects the component for design purposes.

III. Architecture Composition

In the second module, an architect defines the system composition by GUI-supported tools of ProMARTES [3]. First, using the Papyrus plugin of the Eclipse IDE, the architect loads the SW and HW component performance models that may potentially satisfy the functional requirements. Second, the architect specifies the component composition instantiating and binding the selected components. Third, the architect maps the SW components onto a specific HW platform. The instantiations, bindings and mappings represent the SW/HW architecture of the composed system. Finally, the architect defines execution scenarios for the system, by specifying the triggers of the system and the functions that are invoked by those triggers. The scenarios are reconstructed by the Marte2Mast tool [4] into a system model, which defines the workload and the end-to-end functionality of the system. The generated system model is an executable structure and therefore can be applied to performance analysis.

IV. System Model Analysis

The third module provides various analysis algorithms for the generation of performance metrics for predicting the system performance. In order to successfully predict the system's performance, it is better to combine two types of analysis: scheduling and simulation-based. Scheduling analysis is being performed by the MAST scheduling analysis tool [5], providing best- and worst-case response latencies for each task instance scenario associated with a real-time deadline. This formal analysis provides guaranteed worst-case boundary conditions and fast execution. However, it does not provide the detailed time-line data of the task execution.

In order to obtain these time-line data, a simulation-based analysis can be performed by the JSimMAST tool [6]. The

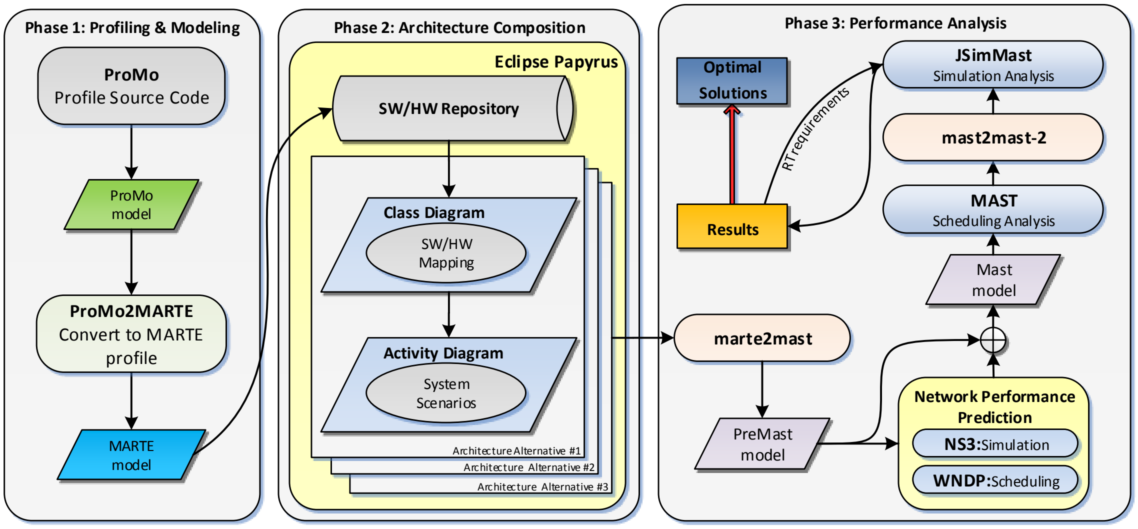

Fig. 1: Overview of the ProMARTES toolkit.

simulation outputs the average-case execution time for each task (scenario) and detailed behaviour time-line data. Simulation cannot guarantee reachability of the worst-case executions, therefore the data obtained by both formal and simulation tools form a synergy in the prediction results.

In CB-RTDS, network delays play a definitive role on the overall performance, e.g. a lost packet and the consequent retransmission of may lead to a missed real-time deadline. Especially in wireless communication schemes, the protocol as well as the signal strength plays an important role in the communication delays. Therefore, an accurate communication delay prediction method is required to be integrated into the performance analysis toolkit. Despite the fact that both MAST and JSimMast tools support network communication schemes, they do not support wireless links and related protocols. Moreover, these tools lack accuracy, when compared to network analysis tools supporting the low OSI layers. To solve this problem, we have integrated our WNDP and the NS3 network analysis tools into the ProMARTES toolkit [1] [7]. The WNDP tool is based on formal methods, while the NS3 is a simulation-based performance prediction tool. Both tools can be independently combined with the MAST and JSimMast tools, depending on the desired type of analysis.

The combination of formal and simulation-based analysis techniques enhances the advantages of each technique and eliminates the limitations of those techniques. For instance, the worst-case performance predictions can be rapidly obtained from the scheduling analysis and can be further used as guideline for next design iterations. Moreover, the architecture alternatives, labeled as ”most performing” by the formal technique, can be further analyzed through simulation techniques, providing both the average-case latencies and the detailed behaviour time-line data, thereby completing the metric set required for architecture verification.

With respect to visualization, the ProMARTES toolkit outputs diagrams with the resource utilization of each CPU, network media as well as a cumulative diagram, which depicts the deadlines of the defined systems scenarios, with the associated predicted end-to-end delays.

V. Conclusion

This paper has introduced the ProMARTES toolkit, which features multiple benefits. First, it guides an architect through all design and analysis phases. Second, the effort for the architect is limited because the toolkit performs all the modeling and computation processes, while the architect only makes the design decisions. Third, the tool bridges and interfaces the design and analysis phases, supporting the conversions from profiling metrics to performance models, then the translation of the performance models to computation delays, and finally, the performance analysis of the composed system. Fourth, the ProMARTES toolkit allows the combination of different analysis tools, depending on the required type of analysis. In conclusion, the toolkit provides cycle-accurate and reliable performance predictions on the system model execution of both network and computation means.

References

[1] K. Triantafyllidis, “Promartes: Profiling, modeling,analysis of real-time embedded systems,” 2013, http://vca.ele.tue.nl/demos/ProMARTES.

[2] K. Triantafyllidis, E. Bondarev, and P. de With, “Low-level profiling and marte-compatible modeling of software components for real-time systems,” in *Software Engineering and Advanced Applications*.

[3] K. Triantafyllidis, E. Bondarev, and P. de With, “Promartes: Performance analysis method and toolkit for real-time systems,” in *Languages, Design Methods, and Tools for Electronic System Design*.

[4] J. Medina and A. Garcia Cuesta, “Model-based analysis and design of real-time distributed systems with ada and the uml profile for marte.”

[5] M. Gonzalez, J. Gutierrez, and J. Drake, “Mast: Modeling and analysis suite for real time applications.”

[6] C. Cuesta, “Jsimmast: The performance analysis simulator for real-time,” 2010, ”http://mast.unican.es/jsimmast/index.html”.

[7] K. Triantafyllidis, W. Aslam, E. Bondarev, J. Lukkien, and P. de With, “Promartes: Accurate network and computation delay prediction for component-based distributed systems,” *submitted to ACM Transactions on Modeling and Performance Evalution of Computing Systems*.