Work-in-Progress: Communication-Centric Analysis of Complex Embedded Computing Systems

Odyurt, U.; Meyer, H.; Polstra, S.; Paradas, E.; Gonzalez Alonso, I.; Pimentel, A.D.

DOI
10.1109/EMSOFT.2018.8537189

Publication date
2018

Document Version
Final published version

Published in
2018 proceedings of the International Conference on Embedded Software (EMSOFT)

Work-in-Progress: Communication-Centric Analysis of Complex Embedded Computing Systems

Uraz Odyurt
Informatics Institute (IvI)
University of Amsterdam
Amsterdam, The Netherlands
u.odyurt@uva.nl

Hugo Meyer
Informatics Institute (IvI)
University of Amsterdam
Amsterdam, The Netherlands
h.d.meyer@uva.nl

Simon Polstra
Informatics Institute (IvI)
University of Amsterdam
Amsterdam, The Netherlands
s.polstra@uva.nl

Evangelos Paradas
ASML Netherlands B.V.
Veldhoven, The Netherlands
evangelos.paradas@asml.com

Ignacio Gonzalez Alonso
ASML Netherlands B.V.
Veldhoven, The Netherlands
ignacio.alonso@asml.com

Andy D. Pimentel
Informatics Institute (IvI)
University of Amsterdam
Amsterdam, The Netherlands
a.d.pimentel@uva.nl

***Abstract*—We show experimental evidence and argue that communication-centric modelling of complex embedded computing systems provides predictive power over the workload dependent behaviour of these systems. System and external observables included in this behaviour can be utilised in the system's analysis. We provide the preliminary results from our *detection* (monitoring) and *imitation* (simulation) phases, both part of a larger workflow in development.**

***Index Terms*—System of systems, Communication-centric modelling, Performance monitoring, Distributed Cyber-Physical Systems (DCPS)**

I. Introduction

Multi-core processor usage has been on the rise throughout the full spectrum of embedded computing systems and their modern applications depend on it. Software components running on these systems also have grown and incorporate complex functionality. Such advancements have been adding to the overall complexity of stand-alone embedded systems. In addition, modern industrial systems are made up of many such stand-alone embedded computing systems, creating a heterogeneous, distributed, networked real-time system, involving many nodes. Such systems have highly dynamic behaviour throughout their operational cycle. Accordingly, the challenge of improving and optimising such behaviour is a profound one [1]. In this paper, we present and discuss an approach towards the aforementioned conundrum, based on high-level modelling and simulation, simplifying analysis (and later on prediction) of the system's performance behaviour. This could make efficient anomaly detection, anomaly prediction and design space exploration feasible, in a complex computing system's domain of possibilities. Such anomaly analyses will result in improved reliability and availability for software components of complex embedded computing systems, in turn improving the reliability and availability of the system itself. Our eventual high-level workflow of such a solution will involve phases such as, *detection*, *imitation*, *prediction* and *enhancement* of the system under scrutiny.

One structural characteristic shared amongst complex embedded computing systems is the presence of one, or more underlying communication subsystems. This is a necessity, as such systems involve many nodes working in tandem and they need to intercommunicate. Communication subsystems predominantely act as a proxy, i.e., a *broker* between *producers* and *consumers* of data. As for the software engineering aspects, different flavours of the broker software pattern can be considered, especially *publish-subscribe* software pattern [2].

As the main contribution, we show that communication-centric modelling of complex embedded computing systems (i.e., cyber-physical systems) can grasp the performance behaviour of the whole system. That is, to understand the behaviour of a complete complex computing system, studying the behaviour from the perspective of its underlying communication subsystem would suffice. This statement means that the process of understanding complex computing system behaviour, performing analysis based on it, and providing improvements to the system, is facilitated. To demonstrate the potential of our communication-centric modelling methodology, we have performed initial experiments on production industrial systems, introduced in the following section.

II. Industrial use-case

ASML's photolithography machines are typical examples of heterogeneous, distributed and networked real-time cyber-physical systems. These machines are used in the micro-fabrication process of semiconductor manufacturing. As a result of extreme complexity of these systems, looking into their functional behaviour for deriving high-level models and exploratory purposes is a futile effort. To be able to create an abstraction, sufficiently reflecting the signature behaviour of such a complex system, we have employed our analysis

This paper is composed as part of the research project 14208, titled *"Interactive DSL for Composable EFB Adaptation using Bi-simulation and Extrinsic Coordination (iDAPT)"*, funded by The Netherlands Organisation for Scientific Research (NWO).

978-1-5386-5560-3/18/$31.00

methodology. We have substituted full system behaviour with the behaviour observed from the system's communication subsystem. As depicted in Fig. 1, it is important to mention that this approach will result in missing parts of the behaviour originating from those processes that are independent of the communication subsystem.

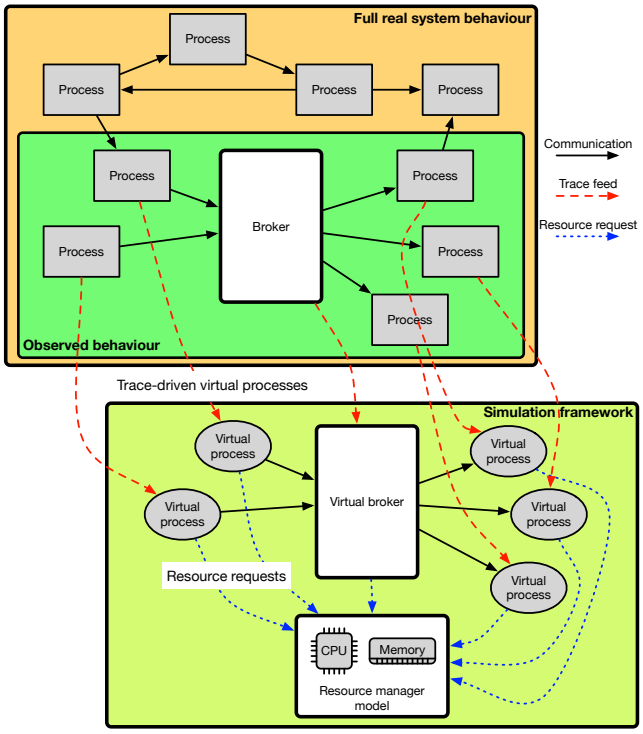

Fig. 1. Capturing and replaying the part of the real system behaviour, represented by the communication subsystem

III. Analysis Phases

Here, we present the first two phases of our workflow, *detection* and *imitation* phases.

A. Monitoring the complex computing system

Our detection phase, i.e., system monitoring, involves capturing system parameters, as well as message-passing-specific communication and computation events, e.g., writing and reading data. The motivation is that by capturing the behaviour reflected at the broker, we will be capturing a high portion of the whole system behaviour, enough to base further analyses on. Benefiting from the fact that all users of the communication broker will utilise its various functionalities, much like a software library, the monitoring is performed in an invasive fashion. This is done by instrumenting the broker's code and capturing intended parameters, e.g., event initiation timestamps, message sizes, etc., as seen in Fig. 1.

B. Trace-driven simulation

At the imitation phase, which encapsulates *automated* high-level model generation and its execution as a simulation, a model is auto-inferred from our collected monitoring data. This model includes virtual processes, representing every detected real process using the broker (about 1/6th of the total application processes for our use-case), a virtual process representing the broker itself, as well as communication links, representing different subscriptions and the communication topology, connecting producers and consumers (shown in Fig. 1). The model is executed as a simulation, using the OMNEST simulation framework [3].

As an initial step, we have opted for a discrete-event simulation, replaying communication subsystem specific events from the trace of the observed real system execution. However, we have also developed alternative, more flexible simulation policies, allowing us to assess different *what-if* scenarios.

C. Preliminary results

Fig. 2 shows one of the metrics we have considered so far to evaluate system behaviour matching, CPU utilisation. The experiment has been performed with a variety of workloads on a production grade ASML system. Workloads consist of different numbers of wafers to be exposed by the machine. Some workloads involve a queue of different wafer batches, e.g., 2 wafers plus 10 wafers. The CPU utilisation trend of the total system is closely matched with the combined CPU utilisation of processes using the communication subsystem specifically. We have compared both accumulated utilisation values captured via the UNIX '`top`' command and from our tracing events (collected via resource usage library, '`getrusage`'), with total system utilisation values from '`top`'. The absolute difference between CPU utilisation of the processes involved with the communication subsystem and total CPU utilisation of the full system represents the amount of undetected behaviour. Fig. 2 depicts this undetected behaviour with small amounts of dispersion (of which, between 0.7% to 8.7% are outliers), indicating a matching behaviour throughout the execution time. Actual utilisation values are confidential and cannot be presented.

The blue box plot is based on a graph resulting from the absolute difference between full system CPU utilisation and observed CPU utilisation for every point in time, all from system parameters ('`top`'). It shows median values of 10.10%, 11.35%, 11.35% and 11.30%. The red box plot is generated similarly, with the only difference that observed CPU utilisations are based on recorded communication events data ('`getrusage`') and shows median values of 10.50%, 11.94%, 12.01% and 11.91%.

IV. Conclusion and Future Steps

We have presented our communication-centric modelling methodology for complex embedded computing systems. It is our intention to complement our claim with more experimental evidence, showing that a communication-centric view encompasses all kinds of observables resulting from a workload. We also aim to deploy our more flexible simulation policies, allowing us to explore the change in behaviour of the system, given different operational parameters, e.g., different process

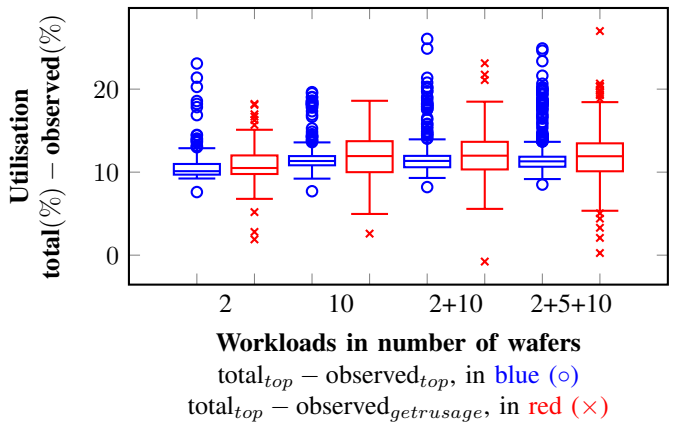

Fig. 2. CPU utilisation differences for multiple workloads, total system vs. communication-centric view

scheduling policies. Following that, we will complete other phases of our methodology.

REFERENCES

[1] J. W. Fowler and O. Rose, "Grand challenges in modeling and simulation of complex manufacturing systems," *SIMULATION*, vol. 80, no. 9, pp. 469–476, 2004.

[2] P. T. Eugster, P. A. Felber, R. Guerraoui, and A. Kermarrec, "The many faces of publish/subscribe," *ACM Comput. Surv.*, vol. 35, no. 2, pp. 114–131, Jun. 2003.

[3] A. Varga and R. Hornig, "An overview of the omnet++ simulation environment," in *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, ser. Simutools '08, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, 60:1–60:10.