Co-Simulation of Hardware and Software in Palladio

Sebastian Weber
sebastian.weber@fzi.de
FZI Forschungszentrum Informatik

Jörg Henss
henss@fzi.de
FZI Forschungszentrum Informatik

Ralf Reussner
reussner@kit.edu
Karlsruhe Institute of Technology

Abstract

To ensure the fulfilment of quality requirements, for example performance, at design time, the software architect can model the software and simulate it with Palladio. The accuracy of the model depends on the estimation of resource demands, which is difficult and error prone. Therefore, in the later stages of development, values should be evaluated based on available information instead of being estimated. An unavailable component implementation or delayed hardware access should not force the software architect to stick with the estimation, but be compensated otherwise, for example with prototypes or hardware simulation. This paper presents six approaches to adapt the evaluation of hardware resource usage in the Palladio software architecture simulation by incorporating co-simulation techniques to compensate different shortcomings. Possible usage scenarios of the approaches are presented and put in relation to the estimated development effort. We present our initial results of implementing one of the approaches, the integration of a hardware-simulation-based resource demand estimation in Palladio. It can compensate unavailable hardware, despite hardware specifications and software being available. The results show that the parameterization options of the hardware simulation were not sufficient to achieve the desired accuracy and the simulation time increases significantly.

1 Introduction

The Palladio approach [4] is used to predict quality requirements of component-based software systems. For this, it uses a model of the software components, how they are connected, a hardware description and a model of the system workload, which form a Palladio Component Model (PCM). These models need to be parameterized, which requires estimated guesses or data from previous projects at design time. Later in the development process, when different artefacts like software components or deployment hardware are available, these estimated guesses can be replaced with measured values.

Usually, a software architect needs all involved parts of the system at hand, which means software, hardware, and workload. If one part is not available, measuring is not possible unless the absence of it is compensated. For example, unavailable hardware can be compensated by using a hardware simulation, which allows for more accurate estimations. Complex and heterogeneous hardware architectures, like embedded controllers in the automotive industry, cannot be modelled accurately in the current PCM due to their highly specialized components. A possible compensation would be using a hardware simulation, connecting the real hardware to the simulation, or extending the PCM.

The contributions of this paper are six approaches and initial results of implementing a hardware-simulation-based resource demand estimation [6]. For every approach we discuss the shortcoming it is intended to overcome and how this can be achieved. Additionally, we present possible scenarios where they would be helpful and what levels of complexity, development effort, parameterization and simulation time we expect. For the implemented approach we present how the chosen hardware simulation was integrated and show initial results, regarding the achieved accuracy and the impact on simulation execution time.

In Section 2, we present the six approaches in detail, followed by the implemented approach in Section 3. We discuss how the approaches and the implementation can be used in Section 4 and end this paper with the conclusion in Section 5.

2 Approaches

The following paragraphs present the six approaches to adapt the evaluation of hardware resource usage in Palladio. We discuss the estimated development costs, which consist of complexity and effort. The first describes how much of the potentially complex concepts of involved software has to be understood and adapted, and the second how many changes are required. The first two approaches use code simulation, the two following approaches propose extensions to the simulation based on the PCM and the last two approaches use code execution and benchmarking.

§1 Replacing the Hardware Simulation The coarse-grained resource-demand-based approach of hardware resource usage evaluation in Palladio is valid for early stages of development, when a more precise estimation and simulation of the resource usage is not possible. In later stages of development, a more accurate simulation should be available if needed. Thus, this approach proposes to replace the hardware resource usage evaluation of Palladio with a fine grained hardware simulation. This can either be done by implementing a hardware simulation for Palladio or by integrating an existing one into it, based on co-simulation standards. This approach is useful in scenarios, where the software is available, but the hardware has yet to be decided. If the needed accuracy is high, estimation and simulation with the current PCM is not sufficient. Complexity and required effort of this approach are high, but could be simplified by using standards for co-simulation (e.g., high level architecture (HLA) [1]). The complex parameterization and high simulation time restrict the applicability of this approach to scenarios with no time limitations.

§2 Hardware Simulation based Resource Demand Estimation Like the first approach, this one uses a hardware simulation to enable software architects to compensate unavailable hardware. The key difference is that the hardware simulation only evaluates the resource demands for the hardware resource usage evaluation in Palladio, which remains unchanged. The evaluation of resource demands can be done before running Palladio or during a Palladio simulation run, which was chosen for the evaluation in Section 3. Possible scenarios revolve around hardware not being available or testing different hardware specifications, for which real hardware access would not be feasible. The complexity and required effort are low, because the resource usage evaluation in Palladio is paused until the hardware simulation evaluated the resource demand. This independence makes the parameterization easier and reduces the simulation time drastically, compared to the first approach.

§3 Adaptive Algorithms If algorithms use hardware utilization or other properties to adapt their behaviour to the system state, e.g. deactivate tracing on high system load, simulating these algorithms requires the same capability. In the simulation of Palladio this could be achieved by adding an additional step to the resource usage evaluation. In the PCM variables have to be specified in the resource demands, which are replaced by the actual values of the hardware properties when evaluating the resource usage. These values can either be measured at the time the resource demand should be evaluated or constantly by an extension of Palladio. Complexity and required effort are medium, because the parsing of resource demands must be adapted and the hardware properties have to be measured at the correct point in time. The parameterization is easy, but the simulation time might increase drastically, because the state space of the simulation grows, depending on how complex the adaptive behaviour of the algorithm is.

§4 Explicit Modelling of Computing Resources Hardware modelling in Palladio is currently limited to the coarse-grained concepts of Central Processing Unit (CPU), Hard Disk Drive (HDD) and network with corresponding properties, like frequency, size and bandwidth. Complex and heterogeneous hardware architectures, e.g. highly specialized controllers or embedded hardware, cannot be modelled as precisely as required for real time systems. Therefore, more fine grained concepts, e.g. processors for cryptography or graphical calculations, should be added to Palladio as components to model the hardware. Estimated effort and complexity are both quite low due to the extensibility of Palladio. The complexity of the parameterization depends on the modelling granularity of the computing resources. Like the adaptive algorithms, the state space of this approach increases with the complexity of the modelled behaviour and thus needed execution time for reaching a steady state.

§5 Hardware-in-the-loop If the hardware is accessible, but the specification is unknown, or if the hardware cannot be modelled, it can be connected directly to the simulation of Palladio by co-simulation. Complexity and required effort of this approach are high because the hardware has to be added as hardware usage evaluation back-end, despite lacking the flexibility of simulations, like pausing and resuming. Using co-simulation standards can reduce the effort necessary here too. The generation of representative input data for the parameterization may prove to be difficult, because inside the simulation of Palladio this data is represented much more abstract. The simulation time is bound to the internal time of the included hardware, which usually should be real time.

§6 Performance-Prototype If the hardware is available, but only parts of the software, a software prototype can enable measurements. To improve the accuracy of the prototype, this approach proposes to include these available parts of the software in the prototype instead of an estimated demand. The drawbacks are creating a dependency between the programming language of the available software and the prototype and increasing the necessary resources to execute the prototype. Complexity and required effort are low, if the same programming language is used and the available code has clearly defined interfaces. Because the prototype recreates the execution behaviour of the real software, the required time is quite high compared to simulation-based approaches. The generation of representative input data might prove difficult here too, because the available code needs more detailed data than the rest of the prototype.

3 Approach Evaluation

We implemented the approach §2, hardware-simulation-based resource demand estimation, to evaluate the usage of hardware simulations for resource demand estimation. It extends different parts of Palladio. The first part is the analyzer, which is used by Palladio to estimate the quality properties of software modelled with the PCM. Palladio supports multiple analyzers, from which we used the analyzer SimuLizar [3]. SimuLizar creates resources like CPU or HDD and corresponding schedulers to control the execution order of demands. In this approach, a new scheduler is added for every existing scheduler of SimuLizar. These new schedulers inherit from their corresponding scheduler and add the resource demand estimation step before actually scheduling the resource demand. Apart from that the scheduling itself remains unchanged.

The hardware simulation used is gem5 [2] due to its high configurability and permissive license. The different Palladio roles have to supply additional input data, like hardware description (system deployer), code to execute (component developer) and input parameters (domain expert). These have to be specified in the corresponding model instance of the PCM to be gathered and forwarded to the hardware simulation. The entirety of input data for the hardware simulation forms a configuration. To reduce the number of time-consuming hardware simulation runs, the results of a simulation per configuration are cached, because the hardware simulation used is deterministic.

The evaluation of the implemented approach shows for different small example applications (e.g. audio encoding, fibonacci numbers) that the ratio of execution time measured to simulated converges from about 5 for small inputs to 0.75, when increasing the input size. The simulation time is about 150 times higher than the simulation of a comparable unmodified PCM, when the hardware simulation is involved [6]. These results are not transferable to larger applications but show that the out of the box usage of the hardware simulation in this implementation is not sufficient.

4 Discussion

The approaches presented in Section 2 highly differ in expected effort and benefit. The second approach §2 was implemented during the course of a master's thesis [6], because it could reasonably be implemented and the usage scenarios are not too specific. Approach §1 and §5 were not feasible for implementation during a master's thesis due to the extensive changes required in Palladio. The approaches §3 and §4 offer too selective benefits. Scaling is usually done by increasing hardware capacity and modelling proprietary hardware is only beneficial if reuse is possible. Approach §6 is too restrictive due to the dependency between prototype and code parts.

Approach §2 shows, that using a hardware simulation in Palladio increases the parameterization needed, which the PCM can only partially support. Therefore, the PCM should be extended and tools may be needed to simplify collecting the necessary parameters. Also, different hardware simulations require different hardware descriptions and code formats (e.g. source or binary code). A similar approach [5] uses another hardware simulation in a prototype but does not achieve the desired accuracy too. The main problems across all approaches are the necessity for more detailed input data and the partially drastic increase in simulation time up to real time or even higher. Therefore, implementing the approaches should include extensive optimizations of PCM in regard to the simulation execution.

5 Conclusion

In this paper, we discussed six different scenarios, where Palladio in its current form is not able to utilize partially available information. We presented approaches to adapt the evaluation of hardware resource usage in Palladio to partly compensate this. The implemented approach of hardware-simulation-based resource demand estimation shows the general applicability but needs to be parameterized more extensive. Implementing the remaining approaches and evaluating the already implemented approach with more extensive parameterization and complex examples is planned as future work, if the corresponding scenarios prove to be viable.

References

[1] "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)– Framework and Rules". In: *IEEE Std 1516-2010 (Revision of IEEE Std 1516-2000)* (2010), pp. 1–38.

[2] N. Binkert et al. "The gem5 simulator". In: *ACM SIGARCH computer architecture news* 39.2 (2011), pp. 1–7.

[3] M. Becker, S. Becker, and J. Meyer. "SimuLizar: Design-Time Modeling and Performance Analysis of Self-Adaptive Systems". In: *Software Engineering 2013*. Ed. by S. Kowalewski and B. Rumpe. Bonn: Gesellschaft für Informatik e.V., 2013, pp. 71–84.

[4] R. H. Reussner et al. *Modeling and simulating software architectures: The Palladio approach.* MIT Press, 2016.

[5] S. Graef. "Connecting Palladio with multicore CPU simulators". B.S. thesis. 2018.

[6] S. Weber. "Co-Simulation of Hardware and Software in the Palladio Component Model". MA thesis. Karlsruhe Institute of Technology (KIT), 2022.