



Universiteit
Leiden
The Netherlands

Towards design space exploration of industrial distributed cyber-physical systems

Saadatmand, F.S.

Citation

Saadatmand, F. S. (2026, September 30). *Towards design space exploration of industrial distributed cyber-physical systems*. Retrieved from <https://hdl.handle.net/1887/4310251>

Version: Publisher's Version

License: [Licence agreement concerning inclusion of doctoral thesis in the Institutional Repository of the University of Leiden](#)

Downloaded from: <https://hdl.handle.net/1887/4310251>

Note: To cite this publication please use the final published version (if applicable).

Chapter 7

Summary, Discussion, and Future Directions

7.1 Summary of Challenges and Contributions

This thesis addresses the growing need for a scalable and automated DSE methodology tailored to complex industrial dCPS such as the ASML TWINSKAN machine used as a case study in the thesis. To this end, the proposed CompDSE methodology systematically tackles five core scientific challenges.

First, to overcome the lack of a coherent and scalable framework for efficient DSE focused on performance evaluation in complex industrial dCPS (challenge **CHL1**), we introduce a structured methodology that combines system-level modeling, automated model derivation and transformation, and discrete-event simulation for evaluation.

Second, to support abstract yet accurate system modeling (challenge **CHL2**), we define models for the application workload, the computing hardware platform, and the workload-to-platform mapping. These models are sufficiently abstract to enable efficient exploration while remaining accurate enough to yield meaningful insights.

Third, we identify the necessary execution traces and develop algorithms to automatically derive the aforementioned models from traces collected during real system operation (challenge **CHL3**), thereby eliminating the need for software source code analysis, which is often impractical in industrial contexts.

Fourth, we implement model-to-model transformations and code generators (challenge **CHL4**) to produce simulation-ready entities, thereby enabling rapid performance analysis in a target simulation environment. In this thesis, the OMNeT++ environment has been targeted because it supports a hierarchical module structure, event scheduling, and message passing between components, making it suitable for simulation of large-scale distributed systems with complex interactions.

Finally, to capture intra-process parallelism alongside inter-process parallelism

(challenge **CHL5**), we propose a refined application workload model by grouping cohesive computation and communication events into semantically meaningful execution blocks as well as a dependency analysis approach to detect which blocks within a process can execute safely in parallel.

Through experiments on real trace data collected from the ASML TWINSKAN machine, we demonstrate the accuracy and efficiency of the proposed CompDSE methodology. It enables efficient performance evaluation of large-scale workloads and supports insightful system architecture exploration for realistic industrial settings.

7.2 Discussion and Future Directions

While the CompDSE methodology offers a practical and generalizable framework for the DSE of large-scale industrial dCPS, several limitations and open directions have emerged during its development and evaluation. These can be framed as key challenges for future work.

Partial Traceability of Subsystems

Due to system constraints, we may be unable to trace all subsystems to automatically derive models. Some subsystems are bare-metal and lack an OS or another infrastructure with tracing facilities. Thus, such untraceable subsystems have been grouped into the ENV model. In addition, some embedded subsystems, although they are traceable, they cannot store large traces locally, similar to most resource-constrained embedded systems. As a result, trace data has to be streamed over the communication network, which introduces bandwidth bottlenecks. In our ASML TWINSKAN case study, it is not feasible to trace more than seven subsystems without disrupting the system operation. Even tracing these seven subsystems requires careful configuration of local buffer sizes, streaming rates, and other parameters.

One possible direction to solve the aforementioned tracing problem could be to repeat the same system operation multiple times under the same configuration and operation mode, and then aggregate the resulting partial traces to automatically derive a more complete application workload model. However, such an approach poses challenges because according to ASML engineers, the system's operational determinism, i.e., executing the exact same workload order across several runs, cannot always be guaranteed. As a result, aggregating traces across executions may introduce inconsistencies. In practice, however, these variations are typically limited in magnitude

and appear as small execution-level differences, which may fall below the resolution of the abstracted workload model and therefore do not prevent meaningful comparative analysis across design alternatives within the defined design space.

Unmodeled Shared-Memory Communication

While the majority of inter-process communication in the ASML TWINSKAN machine is captured through message passing, a small subset of interactions rely on shared-memory communication. These interactions are not traced and modeled in this work. Although their impact on the system model accuracy is minimal in our case, other dCPS in which shared memory communication plays a more significant role may require explicit modeling of such communication to ensure accurate performance analysis.

Events Signatures in Application Workload Model

In our CompDSE methodology, we estimate the workload signature of the computation and communication events using CPU cycle counts as an approximation of execution cost. This abstraction captures only the computational demand of events on a CPU and omits important factors such as cache behavior, memory access latency, and other architectural interactions that may significantly affect actual execution time of events.

This simplification also limits the DSE to hardware platforms with CPU-based computing architectures. As a result, the CompDSE methodology cannot be directly applied to evaluate systems that include alternative computing architectures and components such as GPUs or FPGAs for example. A promising future direction is to develop a mapping approach based on empirical measurements that enables the translation of execution costs (signatures) of events across heterogeneous platforms. For example, by deriving mathematical functions or learning ML models that approximate how execution signatures of events change between different architecture components, CompDSE could be extended to explore a broader hardware platform design space.

Simplified Scheduling for Processes

In this work, we employ a FCFS scheduling policy to schedule the processes mapped onto a given host, while the real ASML TWINSKAN system employs a more complex, priority-based scheduling. Although this simplification does not significantly impact the accuracy of our performance analysis for this system, the same may not hold for other dCPS that operate under hard real-time constraints for example. In such

systems, accurately modeling the scheduling policy, potentially replicating the real-time scheduler used in deployment, may be essential to ensure reliable performance analysis.

Communication Network Model Simplification

Our communication network model within the hardware platform model relies on approximated communication delays associated with communication events. While this high level of abstraction is sufficient for our case study to explore computing hardware platforms and workload-to-platform mappings, it restricts the fidelity with which alternative network architectures can be evaluated. Although the design space defined in Section 3.6 allows modifications to the network topology, such variations are assessed at the abstraction level of the adopted communication model. Detailed protocol behavior, routing dynamics, and fine-grained contention effects are not explicitly represented. Consequently, the impact of topology changes on communication load and overall system performance can only be analyzed in terms of abstract latency and bandwidth parameters rather than detailed network-level effects.

Scope and Limitations of the Explored Design Space

The design space defined in this thesis is intentionally scoped to what can be evaluated efficiently using abstract models. This scoping reflects a trade-off between accuracy and scalability, i.e., increasing model details may improve fidelity for specific effects, but it also increases design-point evaluation time and can make DSE impractical for large and complex industrial distributed systems.

Some limitations arise directly from the chosen abstraction level and cannot be removed within the current methodology. In particular, design dimensions that require instruction-level or micro-architectural visibility are fundamentally outside the scope of the current methodology. This includes exploration of cache hierarchies and cache policies, pipeline-related effects, fine-grained memory-access behavior, and other instruction-level mechanisms. Supporting such variations would require instruction-level traces or cycle-accurate models, which conflict with the intended scalability constraints in large industrial environments.

There are additional limitations in this thesis that do not stem from the proposed methodology itself, but from the current instantiation of the models. For example, as discussed earlier, a more detailed communication network model would enable the network architecture and topology to be treated as explicit design dimensions, thereby allowing a broader exploration of design points. Similarly, heterogeneous

compute resources (e.g., GPUs or FPGAs) could be considered at an abstract level if an appropriate equivalence or mapping of computational capability can be established as discussed.

Memory-related design dimensions occupy an intermediate position with respect to these limitations. While detailed memory behavior is strongly influenced by micro-architectural mechanisms, memory effects can, in principle, be incorporated at a higher level of abstraction. For example, memory can be modeled as an abstract shared service characterized by parameters such as bandwidth, and latency, and application workloads can be augmented with abstract memory-demand attributes. Such representations do not aim to capture precise memory access timing, but rather to reflect the relative impact of memory pressure on execution behavior across alternative design points. However, introducing these abstractions requires additional assumptions regarding overlap between computation and memory access, as well as contention effects.

Algorithmic or functional changes in the application software are not treated as direct variation/design points within the defined design space. However, such changes can still be evaluated within the overall workflow by collecting new execution traces and re-deriving the workload model automatically for the modified software, after which the resulting model can be assessed under alternative platform and mapping configurations.

Generality vs. Accuracy in Parallelism Analysis

The refined application workload model enables the identification of inter-process parallelism, i.e., identifying dependencies between execution blocks within a process to reveal potential parallel execution of blocks. To identify such dependencies, we adopt a conservative dependency analysis approach in order to maintain generality, deadlock-free model execution, and avoid reliance on system domain knowledge. However, incorporating system-specific knowledge in the dependency analysis could make it less conservative, thereby revealing more accurately the potential parallelism between execution blocks. For example, in the ASML TWINSCAN machine, the measurement and exposure phases belong to two distinct and independent functional categories. Recognizing this separation could help distinguish independent execution blocks within a process more effectively.

Additionally, in the refined model, the ordering of events within each execution block is kept fixed to ensure correctness. With sufficient domain knowledge, however, this constraint could also be relaxed, allowing events within a block to be reordered or

executed in parallel, thereby revealing further opportunities for parallelism.

Evolving the Δ Parameter Model into Design Variables

The Δ -parameter approach is introduced to capture the timing impact of untraced embedded hosts that interact with the physical environment via sensors and actuators. Two questions arise: Can individual Δ parameters serve as independent design variables in DSE rather than scaling all Δ parameters, as done in the experiments presented in Chapter 6? If so, can the model be reformulated such that the Δ parameter values capture only delays due to the physical environment (sensors and actuators), while excluding execution and communication latency inside untraced subsystems?

In practice, for a complex dCPS, our current approach generates thousands of Δ parameters because each communication with an untraced host is represented individually. The very large number of Δ parameters creates a huge design space that is costly and infeasible to explore. In addition, many of these parameters are dependent, so varying them independently can yield unrealistic or infeasible design points.

A straightforward way to reduce the number of Δ parameters is to increase trace coverage so that fewer interactions remain untraced. However, as explained earlier, tracing additional subsystems has its own challenges, and moreover, some subsystems will remain untraceable anyway. In such cases, a more effective approach can be to incorporate system domain knowledge. The key idea is to identify a small subset of independent Δ parameters as **representative** parameters and to express the remaining parameters as functions of these representatives. Using only the representative parameters as design variables in DSE can reduce the size of the design space significantly and can prevent infeasible or unrealistic design points from being considered. This can decrease the number of Δ parameters and improve the effectiveness and efficiency of DSE when exploring changes in untraced hosts and in sensors and actuators.

Domain knowledge also helps separate Δ parameters that are tied directly to the physical environment from those that arise from execution or communication within untraced subsystems. For parameters that correspond to repeated interactions with the same sensor or actuator within an operation mode, the observed timings are often constant or nearly constant. In such cases a single characteristic value can serve as the representative for that component. If a different sensor or actuator is selected during DSE, only the corresponding representative value needs to change. Consequently, the portion of the Δ parameters that reflects the physical environment becomes a set of well-defined design variables to explore.

Simulation Speed vs. Model Detail Trade-off

By refining the application workload model, we improve the model accuracy but increase the model simulation time by approximately 10 times, with each design point taking around 10 minutes to evaluate. This fact highlights the need to carefully balance between the model accuracy and model simulation efficiency. Future work aiming for higher model accuracy, such as adding more detail to the ENV model, must consider the impact of such model refinement on the efficiency of DSE, especially the time needed for a single design point evaluation.

OMNet++ Scalability Limits

The OMNeT++ discrete-event simulation environment has been chosen in our work due to its strong capabilities in modeling large-scale networks and distributed systems. However, our practical experiments reveal that it has scalability limits in terms of the number of modules that can be simulated. This means that even if we are able to trace all subsystems in a dCPS, it may not be feasible to include all process entities as separate modules in the simulation environment. Therefore, a systematic study is needed to evaluate the suitability of different discrete-event simulation tools for large-scale workloads in complex dCPS, and to identify alternatives or extensions that can scale better.

7.3 Concluding Remark

This thesis concludes a very focused and industry-driven research effort towards enabling DSE for complex industrial dCPS. While this work marks the end of our contribution to this specific line of research, it lays a foundation that can be extended and applied beyond the system studied here.

The proposed CompDSE methodology is not limited to the ASML TWINSKAN machine. Its trace-based modeling approach, algorithms and tools, simulation infrastructure, and refinement techniques are relatively general and can be adapted to other large-scale systems. In this sense, the work takes a first step towards making DSE more practical and accessible in industrial contexts, supporting system architects in evaluating alternative designs early in the development process through efficient system simulation.

Bibliography

- [1] Sezen Acur and Teun Hendriks. Vision and Outlook for Systems Architecting and Systems Engineering in the High-Tech Equipment Industry. Report TNO-2024-R10542, TNO-ESI, 2024.
- [2] Shreya Adyanthaya, Hadi Alizadeh Ara, João Bastos, Amir Behrouzian, Róbinson Medina Sánchez, Joost van Pinxten, Bram van der Sanden, Umar Waqas, Twan Basten, Henk Corporaal, Raymond Frijns, Marc Geilen, Dip Goswami, Martijn Hendriks, Sander Stuijk, Michel Reniers, and Jeroen Voeten. xCPS: A tool to eXplore Cyber Physical Systems. *ACM SIGBED Review*, 14(1):81–95, 2017.
- [3] Ilge Akkaya, Patricia Derler, Shuhei Emoto, and Edward A. Lee. Systems Engineering A Formal Approach to Power Optimization in CPSs with Delay-Workload Dependence Awareness. *Proceedings of the IEEE*, 104(5):997–1012, 2016.
- [4] John R Allen and Ken Kennedy. Automatic Loop Interchange. In *SIGPLAN symposium on Compiler construction*, pages 233–246, 1984.
- [5] Randy Allen and Ken Kennedy. Automatic Translation of FORTRAN Programs to Vector Form. *Transactions on Programming Languages and Systems (TOPLAS)*, 9(4):491–542, October 1987.
- [6] Rajeev Alur. *Principles of Cyber-Physical Systems*. MIT Press, 2015.
- [7] Maral Amir and Tony Givargis. Pareto Optimal Design Space Exploration of Cyber-Physical Systems. *Internet of Things*, 12:100308, 2020.
- [8] Hyung-Chan An, Hoeseok Yang, and Soonhoi Ha. A Formal Approach to Power Optimization in CPSs With Delay-Workload Dependence Awareness. *Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 35(5):750–763, 2016.
- [9] Emad Arasteh Malekzadeh and Rainer Dömer. Improving Parallelism in System Level Models by Assessing PDES Performance. In *Forum on Specification and Design Languages (FDL)*, pages 1–7. IEEE, 2021.
- [10] David F. Bacon, Susan L. Graham, and Oliver J. Sharp. Compiler Transformations for High-Performance Computing. *ACM Computing Surveys (CSUR)*, 26(4):345–420, December 1994.

- [11] J. Bakakeu, Jonathan Fuchs, Tallal Javied, Matthias Brossog, Jörg Franke, H. Klos, Werner Eberlein, Schirin Tolksdorf, Jörn Peschke, and Lars Jahn. Multi-Objective Design Space Exploration for the Integration of Advanced Analytics in Cyber-Physical Production Systems. In *Industrial Engineering and Engineering Management (IEEM)*, pages 1866–1873. IEEE, 2018.
- [12] Felice Balarin, Yosinori Watanabe, Harry Hsieh, Luciano Lavagno, Claudio Passerone, and Alberto Sangiovanni-Vincentelli. Metropolis: An Integrated Electronic System Design Environment. *Computer*, 36(4):45–52, 2003.
- [13] Simonetta Balsamo, Antiniscia Di Marco, Paola Inverardi, and Marta Simeoni. Model-based Performance Prediction in Software Development: a survey. *Transactions on Software Engineering*, 30(5):295–310, 2004.
- [14] Utpal Banerjee. Dependence. In *Dependence Analysis for Supercomputing*, volume 60, pages 27–43. Springer, 1988.
- [15] Twan Basten, Emiel van Benthum, Marc Geilen, Martijn Hendriks, Fred Houben, Georgeta Igna, Frans Reckers, Sebastian de Smet, Lou Somers, Egbert Teeselink, Nikola Trčka, Frits Vaandrager, Jacques Verriet, Marc Voorhoeve, and Yang Yang. Model-Driven Design-Space Exploration for Embedded Systems: The Octopus Toolset. In Tiziana Margaria and Bernhard Steffen, editors, *Leveraging Applications of Formal Methods, Verification, and Validation (ISoLA)*, volume 6415 of *Lecture Notes in Computer Science*, pages 90–105. Springer, 2010.
- [16] Fabrice Bellard. QEMU, a Fast and Portable Dynamic Translator. In *USENIX Annual Technical Conference (USENIX ATC)*, pages 41–46, June 2005.
- [17] Nathan Binkert, Bradford Beckmann, Gabriel Black, Steven K. Reinhardt, Ali Saidi, Arkaprava Basu, Joel Hestness, Derek R. Hower, Tushar Krishna, Somayeh Sardashti, Rathijit Sen, Korey Sewell, Muhammad Shoaib, Nilay Vaish, Mark D. Hill, and David A. Wood. The gem5 Simulator. *SIGARCH computer architecture news*, 39(2):1–7, 2011.
- [18] William Lloyd Bircher and Lizy K John. Complete System Power Estimation Using Processor Performance Events. *Transactions on Computers*, 61(4):563–577, 2011.
- [19] William Blume, Ramon Doallo, Rudolf Eigenmann, John Grout, Jay Hoeflinger, and Thomas Lawrence. Parallel Programming with Polaris. *Computer*, 29(12):78–82, December 1996.
- [20] Paul Bogdan and Radu Marculescu. Towards a Science of Cyber-Physical Systems Design. In *International Conference on Cyber-Physical Systems (ICCPS)*, pages 99–108. IEEE, 2011.
- [21] Alex Boten and Charity Majors. *Cloud-Native Observability with OpenTelemetry: Learn to Gain Visibility into Systems by Combining Tracing, Metrics, and Logging with OpenTelemetry*. Packt Publishing Ltd, 2022.

- [22] Sami Boukhechem and El-Bay Bourennane. TLM Platform Based on SystemC for STARSoC Design Space Exploration. In *Adaptive Hardware and Systems (AHS)*, pages 354–361, July 2008.
- [23] Arquimedes Canedo and Jan H. Richter. Architectural Design Space Exploration of Cyber-Physical Systems using the Functional Modeling Compiler. *Procedia CIRP*, 21:46–51, 2014. 24th CIRP Design Conference 2014.
- [24] Bryan M. Cantrill, Michael W. Shapiro, and Adam H. Leventhal. Dynamic Instrumentation of Production Systems. In *USENIX Annual Technical Conference (USENIX ATC)*, pages 15–28, June 2004.
- [25] Jianjiang Ceng, Jerónimo Castrillón, Weihua Sheng, Hanno Scharwächter, Rainer Leupers, Gerd Ascheid, Heinrich Meyr, Tsuyoshi Isshiki, and Hiroaki Kunieda. MAPS: an Integrated Framework for MPSoC Application Parallelization. In *Design Automation Conference (DAC)*, pages 754–759, 2008.
- [26] Samarjit Chakraborty, Simon Künzli, and Lothar Thiele. A General Framework for Analysing System Properties in Platform-based Embedded System Designs. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, volume 3, pages 190–195, 2003.
- [27] Anirban DasGupta. The Trimmed Mean. In *Asymptotic Theory of Statistics and Probability*, pages 271–278. Springer New York, 2008.
- [28] John II Davis, Christopher Hylands, Jörn Janneck, Edward A. Lee, Jie Liu, Xiaojun Liu, Steve Neuendorffer, Sonia Sachs, Mary Stewart, Kees Vissers, Paul Whitaker, and Yuhong Xiong. Overview of the Ptolemy Project. Technical Memorandum UCB/ERL M01/11, Department of Electrical Engineering and Computer Science, University of California, Berkeley, March 2001.
- [29] Patricia Derler, Thomas Huining Feng, Edward A. Lee, Slobodan Matic, Hiren D. Patel, Yang Zhao, and Jia Zou. PTIDES: A Programming Model for Distributed Real-Time Embedded Systems. Technical Report UCB/EECS-2008-72, EECS Department, University of California, Berkeley, May 2008.
- [30] Mathieu Desnoyers and Michel R. Dagenais. The LTTng tracer: A Low Impact Performance and Behavior Monitor for GNU/Linux. In *Linux Symposium*, volume 1, pages 209–224, 2006.
- [31] Rainer Dömer, Andreas Gerstlauer, Junyu Peng, Dongwan Shin, Lukai Cai, Haobo Yu, Samar Abdi, and Daniel D. Gajski. System-on-Chip Environment: A SpecC-Based Framework for Heterogeneous MPSoC Design. *EURASIP Journal on Embedded Systems*, 2008:1001–1015, 2008.
- [32] Sven Efftinge, Moritz Eysholdt, Jan Köhnlein, Sebastian Zarnekow, Robert von Massow, Wilhelm Hasselbring, and Michael Hanus. Xbase: Implementing Domain-Specific Languages for Java. *SIGPLAN Not.*, 48(3):112–121, 2012.

- [33] Cagkan Erbas, Andy D Pimentel, Mark Thompson, and Simon Polstra. A Framework for System-Level Modeling and Simulation of Embedded Systems Architectures. *EURASIP Journal on Embedded Systems*, 2007(1):82123, 2007.
- [34] Thomas Huining Feng, Lynn Wang, Wei Zheng, Sri Kanajan, and Sanjit A. Seshia. Automatic Model Generation for Black Box Real-Time Systems. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6. IEEE, 2007.
- [35] Paul Feutrier. Dataflow Analysis of Scalar and Array References. *International Journal of Parallel Programming*, 20(1):23–53, February 1991.
- [36] John Fitzgerald, Carl Gamble, Richard Payne, and Benjamin Lam. Exploring the Cyber-Physical Design Space. In *INCOSE International Symposium (IS)*, volume 27, pages 371–385. Wiley, July 2017.
- [37] Peter Fritzson, Adrian Pop, Karim Abdelhak, Adeel Ashgar, Bernhard Bachmann, Willi Braun, Daniel Bouskela, Robert Braun, Lena Buffoni, Francesco Casella, Rodrigo Castro, Rüdiger Franke, Dag Fritzson, Mahder Gebremedhin, Andreas Heuermann, Bernt Lie, Alachew Mengist, Lars Mikelsons, Kannan Moudgalya, Lennart Ochel, Arunkumar Palanisamy, Vitalij Ruge, Wladimir Schamai, Martin Sjölund, Bernhard Thiele, John Tinnerholm, and Per Östlund. The OpenModelica Integrated Environment for Modeling, Simulation, and Model-Based Development. *Modeling, Identification and Control*, 41(4):241–295, 2020.
- [38] Johannes Geismann and Eric Bodden. A Systematic Literature Review of Model-Driven Security Engineering for Cyber-Physical Systems. *Journal of Systems and Software*, 169:110697, 2020.
- [39] Daniela Genius, Ilias Bournias, Ludovic Apvrille, and Roselyne Chotin. High-Level Partitioning and Design Space Exploration for Cyber-Physical Systems. In *Model-Driven Engineering and Software Development (MODELSWARD)*, pages 84–91. SCITEPRESS, 2020.
- [40] Andreas Gerstlauer, Christian Haubelt, Andy D. Pimentel, Todor Stefanov, Daniel D. Gajski, and Jürgen Teich. Electronic System-Level Synthesis Methodologies. *Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 28(10):1517–1530, October 2009.
- [41] Frank Ghenassia, editor. *Transaction-Level Modeling with SystemC: TLM Concepts and Applications for Embedded Systems*. Springer, New York, USA, 1 edition, 2005.
- [42] Gina Goff, Ken Kennedy, and Chau-Wen Tseng. Practical Dependence Testing. In *SIGPLAN Programming Language Design and Implementation (PLDI)*, pages 15–29. ACM, 1991.

- [43] Imen Graja, Aicha Mechim, Slim Kallel, Nawal Guermouche, and Ahmed Hadj Kacem. Demonstrating BPMN4CPS: Modeling and Verification of Cyber-Physical Systems. In *Consumer Communications & Networking Conference (CCNC)*, page 593. IEEE, 2017.
- [44] Jeff Gray and Bernhard Rumpe. Reflections on the Standardization of SysML 2. *Software and Systems Modeling*, 20(2):287–289, 2021.
- [45] Lukas Gressl, Christian Steger, and Ulrich Neffe. Design Space Exploration for Secure IoT Devices and Cyber-Physical Systems. *ACM Transactions on Embedded Computing Systems (TECS)*, 20(4):1–24, 2021.
- [46] Matthias Gries. Methods for Evaluating and Covering the Design Space During Early Design Development. *Integration, the VLSI Journal*, 38(2):131–183, 2004.
- [47] Thomas Gross, David R O’Hallaron, and Jaspal Subhlok. Task Parallelism in a High Performance Fortran Framework. *Parallel and Distributed Technology*, 2(3):16–26, 1994.
- [48] Soonhoi Ha, Sungchan Kim, Choonseung Lee, Youngmin Yi, Seongnam Kwon, and Young-Pyo Joo. PeaCE: A Hardware-Software Codesign Environment for Multimedia Embedded Systems. *Transactions on Design Automation of Electronic Systems (TODAES)*, 12(3):1–25, 2008.
- [49] Christian Haubelt, Thomas Schlichter, Joachim Keinert, and Mike Meredith. SystemCoDesigner: Automatic Design Space Exploration and Rapid Prototyping from Behavioral Models. In *Design Automation Conference (DAC)*, pages 580–585, 2008.
- [50] Marius Herget, Faezeh Sadat Saadatmand, Martin Bor, Ignacio González Alonso, Todor Stefanov, Benny Akesson, and Andy D. Pimentel. Design Space Exploration for Distributed Cyber-Physical Systems: State-of-the-art, Challenges, and Directions. In *Euromicro Conference on Digital System Design (DSD)*, pages 632–640. IEEE, 2022.
- [51] Steven A Hofmeyr, Stephanie Forrest, and Anil Somayaji. Intrusion Detection using Sequences of System Calls. *Journal of computer security*, 6(3):151–180, 1998.
- [52] Liang Huang and Kenny Wong. Assisting Failure Diagnosis through Filesystem Instrumentation. In *Center for Advanced Studies on Collaborative Research (CASCON)*, pages 160–174, November 2011.
- [53] Tsuyoshi Isshiki, Dongju Li, Hiroaki Kunieda, Toshio Isomura, and Kazuo Satou. Trace-driven Workload Simulation Method for Multiprocessor System-On-Chips. In *Design Automation Conference (DAC)*, pages 232–237, July 2009.

- [54] Ruben Jonk, Jeroen Voeten, Marc Geilen, Rolf Theunissen, Yuri Blankenstein, Twan Basten, and Ramon Schiffelers. Inferring Timed Message Sequence Charts from Execution Traces of Large-scale Component-based Software Systems. (ESR-2019-01), 2019.
- [55] Tero Kangas, Petri Kukkala, Heikki Orsila, Erno Salminen, Marko Hännikäinen, Timo D. Hämäläinen, Jouni Riihimäki, and Kimmo Kuusilinna. UML-based Multiprocessor SoC Design Framework. *Transactions on Embedded Computing Systems (TECS)*, 5(2):281–320, 2006.
- [56] Joachim Keinert, Martin Streubühr, Thomas Schlichter, Joachim Falk, Jens Gladigau, Christian Haubelt, Jürgen Teich, and Michael Meredith. SystemCoDesigner—an Automatic ESL Synthesis Approach by Design Space Exploration and Behavioral Synthesis for Streaming applications. *Transactions on Design Automation of Electronic Systems (TODAES)*, 14(1):1:1–1:23, 2009.
- [57] Bart Kienhuis, Ed F. Deprettere, Pieter van der Wolf, and Kees Vissers. A Methodology to Design Programmable Embedded Systems: The Y-Chart Approach. In *Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS)*, volume 2268 of *Lecture Notes in Computer Science*, pages 18–37. Springer, 2002.
- [58] Dmitrii Kirov, Pierluigi Nuzzo, Roberto Passerone, and Alberto Sangiovanni-Vincentelli. ArchEx: An Extensible Framework for the Exploration of Cyber-Physical System Architectures. In *Design Automation Conference (DAC)*, pages 1–6. ACM, 2017.
- [59] Tim Kogel, Neeraj Goel, and Andreas Wieferink. Method for Generating Workload Models from Execution Traces. US Patent Application US20160357516A1, December 2016.
- [60] Akash Kumar, Shakith Fernando, Yajun Ha, Bart Mesman, and Henk Corporaal. Multi-Processor System-Level Synthesis for Multiple Applications on Platform FPGA. In *Field Programmable Logic and Applications (FPL)*, pages 92–97, 2007.
- [61] Leslie Lamport. The Parallel Execution of DO Loops. *Communications of the ACM*, 17(2):83–93, 1974.
- [62] Leslie Lamport. Time, Clocks, and the Ordering of Events in a Distributed System. In *Concurrency: The Works of Leslie Lamport*, pages 179–196. ACM, 2019.
- [63] Bowen Li, Xin Peng, Qilin Xiang, Hanzhang Wang, Tao Xie, Jun Sun, and Xuanzhe Liu. Enjoy Your Observability: an Industrial Survey of Microservice Tracing and Analysis. *Empirical Software Engineering*, 27, 2022.
- [64] Davide Lorenzoli, Leonardo Mariani, and Mauro Pezzè. Automatic Generation of Software Behavioral Models. In *International Conference on Software Engineering (ICSE)*, pages 501–510. ACM, 2008.

- [65] Federico Maggi, Matteo Matteucci, and Stefano Zanero. Detecting Intrusions through System Call Sequence and Argument Analysis. *Transactions on Dependable and Secure Computing*, 7(4):381–395, 2008.
- [66] Peter S Magnusson, Magnus Christensson, Jesper Eskilson, Daniel Forsgren, Gustav Hallberg, Johan Hogberg, Fredrik Larsson, Andreas Moestedt, and Bengt Werner. Simics: A Full System Simulation Platform. *Computer*, 35(2):50–58, 2002.
- [67] Radu Marculescu and Paul Bogdan. Cyberphysical Systems: Workload Modeling and Design Optimization. *Design & Test of Computers*, 28(4):78–87, 2011.
- [68] Brit Meier, Mladen Skelin, Frans Beenker, and Wouter Leibbrandt. HTSM Systems Engineering Roadmap. Technical Report v1.0, Topsector HTSM, July 2020.
- [69] Hugo Meyer, Uraz Odyurt, Simon Polstra, Evangelos Paradas, Ignacio Gonzalez Alonso, and Andy D. Pimentel. On the Effectiveness of Communication-Centric Modelling of Complex Embedded Systems. In *International Conference on Big Data and Cloud Computing (BDCloud)*, pages 979–986. IEEE, 2018.
- [70] Ahmad Mizan and Greg Franks. An Automatic Trace Based Performance Evaluation Model Building for Parallel Distributed Systems. In *International Conference on Performance Engineering*, page 61–72. ACM, 2011.
- [71] Mahtab Modaber, Martijn Hendriks, Marc Geilen, Twan Basten, and Jeroen Voeten. A Method for Building Trustworthy Hybrid Performance Models for Cyber-Physical Systems of Systems. *IEEE Access*, 12:92733–92752, 2024.
- [72] Steven S. Muchnick. *Advanced Compiler Design Implementation*. Morgan Kaufmann Publishers, 1997.
- [73] Nina Mühleis, Michael Glaß, Liyuan Zhang, and Jürgen Teich. A Co-Simulation Approach for Control Performance Analysis during Design Space Exploration of Cyber-Physical Systems. *ACM SIGBED Review*, 8(2):23–26, 2011.
- [74] Sanjai Narain, Emily Mak, Todd Huster, Jeremy Cohen, Kishore Pochiraju, Brendan Englot, and Niraf Jha. Design Space Exploration for Cyber Physical Systems. Technical report, Perspecta Labs, Basking Ridge, NJ, USA, 2019.
- [75] Kai Neubauer, Philipp Wanko, Torsten Schaub, and Christian Haubelt. Exact Multi-Objective Design Space Exploration using ASPmT. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 257–260. IEEE, 2018.
- [76] Hristo Nikolov, Todor Stefanov, and Ed Deprettere. Multi-Processor System Design with ESPAM. In *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, pages 211–216, 2006.

- [77] Pierluigi Nuzzo, Alberto L. Sangiovanni-Vincentelli, and Richard M. Murray. Methodology and Tools for Next Generation Cyber-Physical Systems: The iCyPhy Approach. In *INCOSE International Symposium*, volume 25, pages 235–249. Wiley, 2015.
- [78] Andy D Pimentel. Exploring Exploration: A Tutorial Introduction to Embedded Systems Design Space Exploration. *IEEE Design & Test*, 34(1):77–90, 2016.
- [79] Andy D. Pimentel. Methodologies for Design Space Exploration. In *Handbook of Computer Architecture*, pages 1–31. Springer Nature Singapore, 2022.
- [80] Andy D. Pimentel, Simon Polstra, Frank Terpstra, A. W. van Halderen, Joseph E. Coffland, and Louis O. Hertzberger. Towards Efficient Design Space Exploration of Heterogeneous Embedded Media Systems. In *Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS)*, volume 2268 of *Lecture Notes in Computer Science*, pages 57–73. Springer Berlin Heidelberg, 2002.
- [81] Riccardo Pincirolì, Connie U. Smith, and Catia Trubiani. Modeling More Software Performance Antipatterns in Cyber-Physical Systems. *Software and Systems Modeling*, 23(4):1003–1023, 2024.
- [82] Claudius Ptolemaeus, editor. *System Design, Modeling, and Simulation using Ptolemy II*. Ptolemy.org, 2014.
- [83] William Pugh. The Omega Test: A Fast and Practical Integer Programming Algorithm for Dependence Analysis. In *ACM/IEEE Conference on Supercomputing*, pages 4–13, 1991.
- [84] Víctor Reyes, Tomás Bautista, Gustavo Marrero, Pedro P Carballo, and Wido Kruijtzer. CASSE: a System-level Modeling and Design-space Exploration Tool for Multiprocessor Systems-on-chip. In *Euromicro Symposium on Digital System Design (DSD)*, pages 476–483, January 2004.
- [85] Valentina Richthammer, Fabian Fassnacht, and Michael Glaß. Search-Space Decomposition for System-Level Design Space Exploration of Embedded Systems. *Transactions on Design Automation of Electronic Systems (TODAES)*, 25(2):1–32, 2020.
- [86] Faezeh Sadat Saadatmand, Todor Stefanov, Ignacio González Alonso, Andy D. Pimentel, and Benny Akesson. CompDSE: A Methodology for Design Space Exploration of Computing Subsystems Within Complex Cyber-Physical Systems. *IET Cyber-Physical Systems: Theory & Applications*, 10(1), 2025.
- [87] Faezeh Sadat Saadatmand, Todor Stefanov, Ignacio González Alonso, Andy D. Pimentel, Benny Akesson, Marius Herget, and Martin Bor. Automated Derivation of Application Workload Models for Design Space Exploration of Industrial Distributed Cyber-Physical Systems. In *Industrial Cyber-Physical Systems (ICPS)*, pages 1–8. IEEE, 2024.

- [88] Faezeh Sadat Saadatmand, Todor Stefanov, Andy D. Pimentel, Benny Akesson, and Ignacio González Alonso. Unraveling Parallelism in Automated Workload Modeling for Distributed Cyber-Physical Systems. In *Euromicro Conference on Digital System Design (DSD)*, pages 24–33. IEEE, September 2025.
- [89] Alberto Sangiovanni-Vincentelli. Quo Vadis, SLD? Reasoning About the Trends and Challenges of System Level Design. *Proceedings of the IEEE*, 95(3):467–506, 2007.
- [90] Donghwan Shin, Domenico Bianculli, and Lionel Briand. PRINS: Scalable Model Inference for Component-based System Logs. *Empirical Software Engineering*, 27(4):87, 2022.
- [91] Zhaogang Shu, Jiafu Wan, Daqiang Zhang, and Di Li. Cloud-Integrated Cyber-Physical Systems for Complex Industrial Applications. *Mobile Networks and Applications*, 21(5):865–878, 2016.
- [92] Todor Stefanov, Andy Pimentel, and Hristo Nikolov. DAEDALUS: System-Level Design Methodology for Streaming Multiprocessor Embedded Systems on Chips. In *Handbook of Hardware/Software Codesign*, pages 1–36. Springer Netherlands, 2017.
- [93] Dave Steinberg, Frank Budinsky, Marcelo Paternostro, and Ed Merks. *EMF: Eclipse Modeling Framework*. Eclipse Series. Addison-Wesley, 2nd edition, 2008.
- [94] Weiqi Sun and Wenbin Dai. An Automatic Software Behavior Model Generation Method for Industrial Cyber-Physical System. In *International Conference on Industrial Informatics (INDIN)*, pages 897–902. IEEE, 2020.
- [95] Janos Sztipanovits, Ted Bapty, Sandeep Neema, Larry Howard, and Ethan Jackson. OpenMETA: A Model- and Component-Based Design Tool Chain for Cyber-Physical Systems. In Saddek Bensalem, Yassine Lakhnech, and Axel Legay, editors, *From Programs to Systems: The Systems Perspective in Computing*, volume 8415 of *Lecture Notes in Computer Science*, pages 235–248. Springer, 2014.
- [96] El-Ghazali Talbi. *Metaheuristics: From Design to Implementation*. John Wiley & Sons, 2009.
- [97] Giacomo Tanganelli, Luca Cassano, Antonio Miele, and Carlo Vallati. A Methodology for the Design and Deployment of Distributed Cyber-Physical Systems for Smart Environments. *Future Generation Computer Systems*, 109:420–430, 2020.
- [98] Prerit Terway, Kenza Hamidouche, and Niraj K. Jha. DISPATCH: Design Space Exploration of Cyber-Physical Systems. *arXiv preprint arXiv:2009.10214*, 2020.
- [99] Lothar Thiele, Iuliana Bacivarov, Wolfgang Haid, and Kai Huang. Mapping Applications to Tiled Multiprocessor Embedded Systems. In *Application of Concurrency to System Design (ACSD)*, pages 29–40, 2007.

- [100] Lothar Thiele, Samarjit Chakraborty, and Martin Naedele. Real-time Calculus for Scheduling Hard Real-time Systems. In *International Symposium on Circuits And Systems (ISCAS)*, volume 4, pages 101–104. IEEE, 2000.
- [101] Mark Thompson, Hristo Nikolov, Todor Stefanov, Andy D. Pimentel, Cagkan Erbas, Simon Polstra, and Ed F. Deprettere. A Framework for Rapid System-Level Exploration, Synthesis, and Programming of Multimedia MP-SoCs. In *International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, pages 9–14, 2007.
- [102] Mark Thompson and Andy D. Pimentel. Exploiting Domain Knowledge in System-Level MPSoC Design Space Exploration. *Journal of Systems Architecture*, 59(7):351–360, 2013.
- [103] Casper Thule, Kenneth Lausdahl, Cláudio Gomes, Gerd Meisl, and Peter Gorm Larsen. Maestro: The INTO-CPS co-simulation framework. *Simulation Modelling Practice and Theory*, 92:45–61, 2019.
- [104] Trace Compass Developers. Trace Compass, 2024. [Software]. Available at: <https://tracecompass.org/>.
- [105] Konstantinos Triantafyllidis, Yuri Blankenstein, Sobhan Niknam, and Jos Hegge. Platform Performance Suite (PPS): A Framework for Performance Analysis & Diagnosis of Complex Cyber-physical Systems. In *International Conference on Performance Engineering (ICPE)*, pages 168–180, May 2025.
- [106] Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings. MCDS: AI Augmented Workflow Scheduling in Mobile Edge Cloud Computing Systems. *Transactions on Parallel and Distributed Systems (TPDS)*, 33(11):2794–2807, 2021.
- [107] Bram van der Sanden, Yonghui Li, Joris van den Aker, Benny Akesson, Tjerk Bijlsma, Martijn Hendriks, Kostas Triantafyllidis, Jacques Verriet, Jeroen Voeten, and Twan Basten. Model-Driven System-Performance Engineering for Cyber-Physical Systems. In *International Conference on Embedded Software (EMSOFT)*, pages 11–22. ACM, 2021.
- [108] Bram van Hoof, Arjan Holscher, Ralf Gommers, Jeroen Cottaar, Marcel Raas, Samah Khalek, Jan van Kemenade, Maarten Voncken, Roelof de Graaf, Elliot Oti, Stefan Weichselbaum, Richard Droste, Byeong Soo Lee, Chansam Chang, Young Seog Kang, Young Ha Kim, Jeong-Heung Kong, Jong Hoon Jang, Young Sun Nam, and Hyunwoo Hwang. Layout Independent Leveling (LIL) on NXT: 1980Di Immersion Scanners for Enhanced Productivity. In *Optical Microlithography XXX*, volume 10147, pages 374–381. SPIE, 2017.
- [109] Peter Van Stralen and Andy D. Pimentel. Scenario-based Design Space Exploration of MPSoCs. In *International Conference on Computer Design (ICCD)*, pages 305–312, October 2010.

- [110] Yon Vanommeslaeghe, Joachim Denil, Jasper De Viaene, David Ceulemans, Stijn Derammelaere, and Paul De Meulenaere. Leveraging Domain Knowledge for the Efficient Design-Space Exploration of Advanced Cyber-Physical Systems. In *Euromicro Conference on Digital System Design (DSD)*, pages 351–358. IEEE, 2019.
- [111] Yon Vanommeslaeghe, Joachim Denil, Jasper De Viaene, David Ceulemans, Stijn Derammelaere, and Paul De Meulenaere. Ontological Reasoning in the Design Space Exploration of Advanced Cyber-Physical Systems. *Microprocessors and Microsystems*, 85:104151, 2021.
- [112] Yon Vanommeslaeghe, Joachim Denil, Bert Van Acker, and Paul De Meulenaere. Automatic Generation of Workflows for Efficient Design Space Exploration for Cyber-Physical Systems. In *Internet of Things (iThings) and Green Computing and Communications (GreenCom) and Cyber, Physical and Social Computing (CPSCom) and SmartData and Cybermatics*, pages 346–351. IEEE, 2021.
- [113] András Varga and Rudolf Hornig. An Overview of the OMNeT++ Simulation Environment. In *International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*. ICST, 2008.
- [114] Jiang Wan, Arquimedes Canedo, and Mohammad Abdullah Al Faruque. Functional Model-Based Design Methodology for Automotive Cyber-Physical Systems. *IEEE Systems Journal*, 11(4):2028–2039, 2017.
- [115] Ernesto Wandeler, Lothar Thiele, Marcel Verhoef, and Paul Lieveise. System Architecture Evaluation Using Modular Performance Analysis: A Case Study. *Software Tools for Technology Transfer (STTT)*, 8(6):649–667, November 2006.
- [116] Vincent M Weaver. Self-monitoring Overhead of the Linux perf_event Performance Counter Interface. In *International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 102–111, 2015.
- [117] Raphael Weber, Stefan Henkler, and Achim Rettberg. Multi-Objective Design Space Exploration for Cyber-Physical Systems satisfying hard Real-Time and Reliability Constraints. In *Proceedings of IDEAL’14 Workshop*. Springer, April 2014.
- [118] Reinhard Wilhelm, Jakob Engblom, Andreas Ermedahl, Niklas Holsti, Stephan Thesing, David Whalley, Guillem Bernat, Christian Ferdinand, Reinhold Heckmann, Tulika Mitra, Frank Mueller, Isabelle Puaut, Peter Puschner, Jan Staschulat, and Per Stenström. The Worst-case Execution-Time Problem—overview of Methods and Survey of Tools. *Transactions on Embedded Computing Systems (TECS)*, 7(3):36:1–36:53, 2008.
- [119] David Wilner. WindView: A Tool for Understanding Real-Time Embedded Software Through System Visualization. *SIGPLAN Not.*, 30(11):117–123, 1995.

- [120] Yifeng Xiao, Chanwook Oh, Michele Lora, and Pierluigi Nuzzo. Efficient Exploration of Cyber-Physical System Architectures Using Contracts and Subgraph Isomorphism. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6. IEEE, 2024.