Simulation and Design Software Systems for Testing and Training Process Automation Projects

Dr. Osama Mohammed Elmardi Suleiman Khayal
Mechanical Engineering Department
Faculty of Engineering and Technology
Nile Valley University, Atbara, Sudan

1. Introduction

Chemical engineers are accustomed to software for designing processes and simulation. Simulation systems such as Matlab and Aspen Plus are commonly referenced in chemical engineering curricula as required courseware and study tools. Automation professionals are also becoming used to applying simulation to operator training, system testing, and commissioning of plant process control systems. Plant design simulation programs are substantially different from systems used for training and commissioning. Many of the most common plant design simulation programs are steady state, low-resolution simulations that are not usable for automation or plant life-cycle management.

2. Simulation

Simulation is usually integrated into the plant life cycle at the front-end engineering and design stage and used to test application software using a simulated I/O system and process models in an offline environment. The same simulation system can then be used to train operations staff on the automation and control systems and the application software that will be running on the hardware platform. In the most advanced cases, integration with manufacturing execution systems (MES) and electronic batch records (EBR) systems can be tested while the operations staff is trained. Once installed, the simulation system can be used to test and validate upgrades to the control system before they are installed. The simulator then becomes an effective tool for testing control system modifications in a controlled, offline environment. In addition, plant operations staff and new operators can become qualified on new enhancements and certified on the existing operating system. The simulation system can be used as a test bed to try new control strategies, build new product recipes, and design new interlock strategies prior to proposing those changes as projects to production management. The simulator can also be an effective risk management tool by providing the ability to conduct failure testing in an offline environment rather than on the operating process.

Simulation's ROI has been proven to be effectual and substantial across all process industries, from batch to continuous processes. The savings come from identifying and correcting automation system errors prior to startup and commissioning and by identifying so-called "sleeping" errors, or inadequacies in the system's application software. Additional savings come from accelerating operators' learning curves and the ability to train operators on upset or emergency conditions they normally do not encounter in day-to-day operations. This reduces operator errors in responding to abnormal situations.

3. Best practices for simulation systems in automation

Nonintrusive simulation interfaces allow the user to test the control system configuration without making any modifications to the configuration database. As far as the operator is concerned, the system is "live"; as far as the database is concerned, there are no changes. By nature, a nonintrusive simulation interface will provide a virtual I/O interface to the process controller application code that supports I/O and process simulation and modeling. It allows the application software to run in a normal mode without any modification so that the testing application software is identical to the production application software. In addition, the nonintrusive interface will not produce "dead code" that formerly was necessary to spoof the control system during testing.

This type of simulation interface supports complete and thorough testing of the application software and provides a benchmark of process controller performance, including CPU and memory loading, application software order execution, and timing.

4. Ground-up testing and training

Ground-up testing and training is an incremental approach, integrated with the automation project life cycle. This approach to application software testing and training has several benefits. Ground-up testing allows identification and correction of issues early in the project, when they can be corrected before being propagated throughout the system.

Additionally, ground-up testing allows the end-user and operations staff to gain acceptance and familiarity with the automation system throughout the entire project instead of at one final acceptance test.

Best practices dictate that training and testing are inextricably linked throughout the automation project. Here are some general guidelines for following this best practice:

● *Control modules* are the base-level database elements of the process automation system. These include motors, discrete valves, analog loops, and monitoring points. Testing of these elements can be effectively accomplished with simple tieback simulations and automated test scripts. Operator training on these elements can also bolster buy-in of the automation system and provide a review of usability features.

● *Equipment modules* are the next level of an automation system and generally refer to simple continuous unit operations such as charging paths, valve manifolds, and package equipment. Testing these elements can be effectively accomplished with tieback simulations and limited process dynamics. Operator training on these elements is also valuable.

● *Sequence, batch controls, and continuous advanced controls* are the next layer of the automation system. Effective testing of these controls generally requires a mass balance simulation with effective temperature and pressure dynamics models. Operator training at this level is necessary due to the complexity of the controls and the user interface.

● *MES applications and business system integration* is the final layer of most automation systems. Mass and heat balance models are usually required for effective testing of this layer. Training at this level may be extended beyond the operations staff to include quality assurance, information technology, and other affected departments.

● *Display elements* for each layer of the automation should be tested with the database elements listed here. In other words, control module faceplates are tested with the control modules, and batch help screens are tested with the batch controls.

5. Simulation system selection

The proven best practice is to use actual automation system controllers (or equivalent soft controllers) and application software with a simulation "companion" system. Simulation systems that use a "rehosted" automation system should be eliminated for system testing and avoided for operator training. Use of the actual automation system components with a simulation system allows effective testing and training on HMI use, display access familiarity, process and emergency procedures, response to process upsets, and control system dynamics. This approach builds automation system confidence in operations staff, resulting in more effective use of the automation system and greater benefits.

6. Simulation for automation in the validated industries

Automation system users and integrators for the validated industries need to be concerned about the GAMP4 Guidelines when they are applying simulation systems to their automation projects.

The GAMP4 Guidelines clearly state that simulation systems are allowable tools for automation system testing. The guidelines also make two requirements for the treatment of the automation system application software. First, they require that the application software be "frozen" prior to software integration and system acceptance testing. Second, they require the removal of "dead code" prior to testing. These two requirements dictate the use of nonintrusive simulation interfaces.

The GAMP4 Guidelines also state several requirements for the supplier of simulation systems for testing of automation projects. The supplier must have a documented quality and software development program in line with industry best practices. The product used should be designed specifically for process control system testing and operator training. Finally, the product should be a commercially available, off- the-shelf (COTS) tool, delivered in validated, tested object code.

Additionally, operator training management modules allow comprehensive development of structured operator training sessions with scripted scenarios and process events. Large or small automation projects can both use a simulation system with a scalable client/server architecture and a stable, repeatable simulation engine.

7. Conclusion

The use of simulation systems for testing and training process automation projects has been proven to reduce time to market and increase business results. The same systems can be utilized in automation life-cycle management to reduce operational costs and improve product quality.

References

1. Martin Berutti, Mynah, (2006). Optimizing Results in Automation Projects with Simulation, Technologies, published on www.controlglobal.com.
2. Agha, G. A., I. A. Mason, S. F. Smith, and C. L. Talcott, 1997: A foundation for actor computation. Journal of Functional Programming, 7(1), 1–72.
3. Allen, F. E., 1970: Control flow analysis. SIGPLAN Notices, 5(7), 1–19.
4. Alur, R., S. Kannan, and M. Yannakakis, 1999: Communicating hierarchical state machines. In 26th International Colloquium on Automata, Languages, and Programming, Springer, vol. LNCS 1644, pp. 169–178.
5. Andalam, S., P. S. Roop, and A. Girault, 2010: Predictable multithreading of embedded applications using PRET-C. In Formal Methods and Models for Code sign (MEMOCODE), IEEE/ACM, Grenoble, France, pp. 159–168. doi:10.1109/MEMCOD. 2010.5558636.
6. Andr´e, C., 1996: Sync Charts: a visual representation of reactive behaviors. Tech. Rep. RR 95–52, revision: RR (96–56), University of Sophia-Antipolis. Available from: http://www-sop.inria.fr/members/Charles.Andre/CA% 20Publis/SYNCCHARTS/overview.html.
7. Andr´e, C., F. Mallet, and R. d. Simone, 2007: Modeling time(s). In Model Driven Engineering Languages and Systems (Models/UML), Springer, Nashville, TN, vol. LNCS 4735, pp. 559–573. doi:10.1007/978-3-540-75209-7_38.
8. Arbab, F., 2006: A behavioral model for composition of software components. L'Object, Lavoisier, 12(1), 33–76. doi:10.3166/objet.12.1.33-76.
9. Arvind, L. Bic, and T. Ungerer, 1991: Evolution of data-flow computers. In Gaudiot, J.-L. and L. Bic, eds., Advanced Topics in Data-Flow Computing, Prentice-Hall.
10. Baccelli, F., G. Cohen, G. J. Olster, and J. P. Quadrat, 1992: Synchronization and Linearity, An Algebra for Discrete Event Systems. Wiley, New York.

11.Baier, C. and M. E. Majster-Cederbaum, 1994: Denotational semantics in the CPO and metric approach. Theoretical Computer Science, 135(2), 171–220.
12. Balarin, F., H. Hsieh, L. Lavagno, C. Passerone, A. L. Sangiovanni-Vincentelli, and Y. Watanabe, 2003: Metropolis: an integrated electronic system design environment. Computer, 36(4).
13.Baldwin, P., S. Kohli, E. A. Lee, X. Liu, and Y. Zhao, 2004: Modeling of sensor nets in Ptolemy II. In Information Processing in Sensor Networks (IPSN), Berkeley, CA, USA. Available from: http://ptolemy.eecs.berkeley.edu/publications/papers/04/VisualSense/.
14. 2005: Visual sense: Visual modeling for wireless and sensor network systems. Technical Report UCB/ERL M05/25, EECS Department, University of California. Available from: http://ptolemy.eecs.berkeley.edu/publications/papers/05/visualsense/index.htm.
15. Basu, A., M. Bozga, and J. Sifakis, 2006: Modeling heterogeneous real-time components in BIP. In International Conference on Software Engineering and Formal Methods (SEFM), Pune, pp. 3–12.
16. Benveniste, A. and G. Berry, 1991: The synchronous approach to reactive and real-time systems. Proceedings of the IEEE, 79(9), 1270–1282.
17. Benveniste, A., P. Caspi, P. Le Guernic, and N. Halbwachs, 1994: Data-flow synchronous languages. In Bakker, J. W. d., W.-P. d. Roever, and G. Rozenberg, eds., A Decade of Concurrency Reflections and Perspectives, Springer-Verlag, Berlin, vol. 803 of LNCS, pp. 1–45.
18. Benveniste, A. and P. Le Guernic, 1990: Hybrid dynamical systems theory and the SIGNAL language. IEEE Tr. on Automatic Control, 35(5), 525–546.
19.Berry, G., 1976: Bottom-up computation of recursive programs. Revue Franaise dAutomatique, Informatique et Recherche Oprationnelle, 10(3), 47–82.
20.—, 1999: The Constructive Semantics of Pure Esterel - Draft Version 3. Book
21.Draft. Available from: http://www-sop.inria.fr/meije/esterel/doc/ main-papers.html.
22.—, 2003: The effectiveness of synchronous languages for the development of safety critical systems. White paper, Esterel Technologies. Available from: http://www.esterel-technologies.com.
23.Berry, G. and G. Gonthier, 1992: The Esterel synchronous programming language: Design, semantics, implementation. Science of Computer Programming, 19(2), 87–152. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary? doi=10.1.1.17.5606.
24.Bhattacharya, B. and S. S. Bhattacharyya, 2000: Parameterized dataflow modeling of DSP systems. In International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Istanbul, Turkey, pp. 1948–1951.
25.Bhattacharyya, S. S., J. T. Buck, S. Ha, and E. Lee, 1995: Generating compact code from dataflow specifications of multi rate signal processing algorithms. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 42(3), 138–150. doi:10.1109/81.376876.

26. Bhattacharyya, S. S., J. T. Buck, S. Ha, and E. A. Lee, 1993: A scheduling framework for minimizing memory requirements of multirate DSP systems represented as dataflow graphs. In VLSI Signal Processing VI, IEEE, Veldhoven , The Netherlands, pp. 188–196. doi:10.1109/VLSISP.1993.404488.
27. Bhattacharyya, S. S. and E. A. Lee, 1993: Scheduling synchronous dataflow graphs for efficient looping. Journal of VLSI Signal Processing Systems, 6(3), 271–288. doi: 10.1007/BF01608539.
28. Bhattacharyya, S. S., P. Murthy, and E. A. Lee, 1996a: APGAN and RPMC: Complementary heuristics for translating DSP block diagrams into efficient software implementations. Journal of Design Automation for Embedded Systems, 2(1), 33–60. doi:10.1023/A:1008806425898.
29. Bhattacharyya, S. S., P. K. Murthy, and E. A. Lee, 1996b: Software Synthesis from Dataflow Graphs. Kluwer Academic Publishers, Norwell, Mass.
30. Bilsen, G., M. Engels, R. Lauwereins, and J. A. Peperstraete, 1996: Cyclo-static dataflow. IEEE Transactions on Signal Processing, 44(2), 397–408. doi:10.1109/78. 485935.
31. Bock, C., 2006: SysML and UML 2 support for activity modeling. Systems Engineering, 9(2), 160 –185.
32. Booch, G., I. Jacobson, and J. Rumbaugh, 1998: The Unified Modeling Language User Guide. Addison-Wesley.
33. Boussinot, F., 1991: Reactive c: An extension to c to program reactive systems. Software Practice and Experience, 21(4), 401–428.
34. Box, G. E. P. and N. R. Draper, 1987: Empirical Model-Building and Response Surfaces. Wiley Series in Probability and Statistics, Wiley.
35. Brock, J. D. and W. B. Ackerman, 1981: Scenarios, a model of non-determinate computation. In Conference on Formal Definition of Programming Concepts, Springer-Verlag, vol. LNCS 107, pp. 252–259.
36. Broenink, J. F., 1997: Modeling, simulation and analysis with 20-Sim. CACSD, 38(3), 22–25.
37. Brooks, C., C. Cheng, T. H. Feng, E. A. Lee, and R. von Hanxleden, 2008: Model
38. engineering using multimode ling. In International Workshop on Model Co-Evolution and Consistency Management (MCCM), Toulouse, France. Available from: http://chess.eecs.berkeley.edu/pubs/486.html.
39. Brooks, C., E. A. Lee, X. Liu, S. Neuendorffer, Y. Zhao, and H. Zheng, 2004: Heterogeneous concurrent modeling and design in Java. Tech. Rep. Technical Memorandum UCB/ERL M04/16, University of California. Available from: http://ptolemy. eecs.berkeley.edu/papers/04/ptIIDesignSoftware/.
40. Brooks, C. H. and E. A. Lee, 2003: Ptolemy II coding style. Tech. Rep. Technical Memorandum UCB/ERL M03/44, University of California at Berkeley. Available from: http://ptolemy.eecs.berkeley.edu/publications/papers/ 03/coding style/.
41. Broy, M., 1983: Applicative real time programming. In Information Processing 83, IFIP World Congress, North Holland Publ. Company, Paris, pp. 259–264.
42.

43. Broy, M. and G. Stefanescu, 2001: The algebra of stream processing functions. Theoretical Computer Science, 258, 99–129.
44. Bryant, V., 1985: Metric Spaces - Iteration and Application. Cambridge University Press.
45. Buck, J. T., 1993: Scheduling dynamic dataflow graphs with bounded memory using the token flow model. Ph.D. Thesis Tech. Report UCB/ERL 93/69, University of California, Berkeley. Available from: http://ptolemy.eecs.berkeley.edu/ publications/papers/93/jbuckThesis/.
46. Buck, J. T., S. Ha, E. A. Lee, and D. G. Messerschmitt, 1994: Ptolemy: A framework for simulating and prototyping heterogeneous systems. Int. Journal of Computer Simulation, special issue on "Simulation Software Development", 4, 155–182. Available from: http://ptolemy.eecs.berkeley.edu/publications/papers/ 94/JEurSim/.
47. Burch, J. R., R. Passerone, and A. L. Sangiovanni-Vincentelli, 2001: Overcoming heterophobia: Modeling concurrency in heterogeneous systems. In International Conference on Application of Concurrency to System Design, p. 13.
48. Buss, A. H. and P. J. Sanchez, 2002: Building complex models with LEGOs (listener event graph objects). Winter Simulation Conference (WSC 02), 1, 732–737.
49. Cardelli, L., 1997: Type systems. In Tucker, A. B., ed., The Computer Science and Engineering Handbook, CRC Press, chap. 103, pp. 2208–2236, http://lucacardelli.name/Papers/TypeSystems
50. Cardelli, L. and P.Wegner, 1985: On understanding types, data abstraction, and polymorphism. ACM Computing Surveys (CSUR), 17(4), 471 – 523.
51. Carloni, L. P., R. Passerone, A. Pinto, and A. Sangiovanni-Vincentelli, 2006: Languages and tools for hybrid systems design. Foundations and Trends in Electronic Design Automation, 1(1/2). doi:10.1561/1000000001.
52. Caspi, P., P. Raymond, and S. Tripakis, 2007: Synchronous Programming. In Lee, I., J. Leung, and S. Son, eds., Handbook of Real-Time and Embedded Systems, Chapman & Hall, pp. 14–1 — 14–21. Available from: http://www-erimag.imag.fr/ ˜tripakis/papers/handbook07.pdf.
53. Cassandras, C. G., 1993: Discrete Event Systems, Modeling and Performance Analysis. Irwin.
54. Cataldo, A., E. A. Lee, X. Liu, E. Matsikoudis, and H. Zheng, 2006: A constructive fixed-point theorem and the feedback semantics of timed systems. In Workshop on Discrete Event Systems (WODES), Ann Arbor, Michigan. Available from: http://ptolemy.eecs.berkeley.edu/publications/papers/06/constructive/.
55. Chandy, K. M. and J. Misra, 1979: Distributed simulation: A case study in design and verification of distributed programs. IEEE Trans. on Software Engineering, 5(5), 440–452.
56. Clarke, E. M., O. Grumberg, and D. A. Peled, 2000: Model checking. MIT Press, ISBN 0-262-03270-8.
57. Coffman, E. G., Jr. (Ed), 1976: Computer and Job Scheduling Theory. Wiley.
58. Conway, M. E., 1963: Design of a separable transition-diagram compiler. Communications of the ACM, 6(7), 396–408.

59.Corbett, J. C., J. Dean, M. Epstein, A. Fikes, C. Frost, J. Furman, S. Ghemawat,
60.A. Gubarev, C. Heiser, P. Hochschild,W. Hsieh, S. Kanthak, E. Kogan, H. Li, A. Lloyd, S. Melnik, D. Mwaura, D. Nagle, S. Quinlan, R. Rao, L. Rolig, Y. Saito, M. Szymaniak, C. Taylor, R. Wang, and D. Woodford, 2012: Spanner: Googles globally-distributed database. In OSDI.
61.Cousot, P. and R. Cousot, 1977: Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fix points. In Symposium on Principles of Programming Languages (POPL), ACM Press, pp. 238–252.
62.Creeger, M., 2005: Multicore CPUs for the masses. ACM Queue, 3(7), 63–64.
63.Davey, B. A. and H. A. Priestly, 2002: Introduction to Lattices and Order. Cambridge University Press, second edition ed.
64.de Alfaro, L. and T. Henzinger, 2001: Interface automata. In ESEC/FSE 01: the Joint 8th European Software Engineering Conference and 9th ACM SIGSOFT International Symposium on the Foundations of Software Engineering.
65.Dennis, J. B., 1974: First version data flow procedure language. Tech. Rep. MAC TM61, MIT Laboratory for Computer Science.
66.Derler, P., E. A. Lee, and S. Matic, 2008: Simulation and implementation of the ptides programming model. In IEEE International Symposium on Distributed Simulation and Real Time Applications (DS-RT), Vancouver, Canada.
67.Dijkstra, E. W., 1968: Go to statement considered harmful (letter to the editor). Communications of the ACM, 11(3), 147–148.
68.Edwards, S. A. and E. A. Lee, 2003a: The semantics and execution of a synchronous block-diagram language. Science of Computer Programming, 48(1), 21–42. doi: 10.1016/S0167-6423(02)00096-5.
69.—, 2003b: The semantics and execution of a synchronous block-diagram language. Science of Computer Programming, 48(1), 21–42. Available from: http://ptolemy. eecs.berkeley.edu/papers/03/block diagram/.
70.Eidson, J. C., 2006: Measurement, Control, and Communication Using IEEE 1588. Springer. doi:10.1007/1-84628-251-9.
71.Eidson, J. C., E. A. Lee, S. Matic, S. A. Seshia, and J. Zou, 2012: Distributed real-time software for cyber-physical systems. Proceedings of the IEEE (special issue on CPS), 100(1), 45–59. doi:10.1109/JPROC.2011.2161237.
72.Eker, J. and J. W. Janneck, 2003: Cal language report: Specification of the cal actor language. Tech. Rep. Technical Memorandum No. UCB/ERL M03/48, University of California, Berkeley, CA. Available from: http://ptolemy.eecs.berkeley. edu/papers/03/Cal/index.htm.
73.Eker, J., J. W. Janneck, E. A. Lee, J. Liu, X. Liu, J. Ludvig, S. Neuendorffer, S. Sachs, and Y. Xiong, 2003: Taming heterogeneity—the Ptolemy approach. Proceedings of the IEEE, 91(2), 127–144. Available from: http://www.ptolemy.eecs. berkeley.edu/publications/papers/03/TamingHeterogeneity/.
74.Encyclopedia Britannica, 2010: Ockham’s razor. Encyclopedia Britannica Online, Retrieved June 24, 2010. Available from: http://www.britannica.com/EBchecked/topic/424706/Ockhams-razor.

75. Falk, J., J. Keiner, C. Haubelt, J. Teich, and S. S. Bhattacharyya, 2008: A generalized static data flow clustering algorithm for mpsoc scheduling of multimedia applications. In Embedded Software (EMSOFT), ACM, Atlanta, Georgia, USA.
76. Faustini, A. A., 1982: An operational semantics for pure dataflow. In Proceedings of the 9th Colloquium on Automata, Languages and Programming (ICALP), Springer-Verlag, vol. Lecture Notes in Computer Science (LNCS) Vol. 140, pp. 212–224.
77. Feng, T. H., 2009: Model transformation with hierarchical discrete-event control. PhD Thesis UCB/EECS-2009-77, EECS Department, UC Berkeley. Available from: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-77.html.
78. Feng, T. H. and E. A. Lee, 2008: Real-time distributed discrete-event execution with fault tolerance. In Real-Time and Embedded Technology and Applications Symposium (RTAS), IEEE, St. Louis, MO, USA. Available from: http://chess.eecs. berkeley.edu/pubs/389.html.
79. Feng, T. H., E. A. Lee, H. D. Patel, and J. Zou, 2008: Toward an effective execution policy for distributed real-time embedded systems. In 14th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), St. Louis, MO, USA. Available from:https://chess.eecs.berkeley.edu/pubs/402.
80. Feng, T. H., E. A. Lee, and L. W. Schruben, 2010: Ptera: An event-oriented model of computation for heterogeneous systems. In EMSOFT, ACM Press, Scottsdale, Arizona, USA. doi:10.1145/1879021.1879050.
81. Feredj, M., F. Boulanger, and A. M. Mbobi, 2009: A model of domain-polymorph component for heterogeneous system design. The Journal of Systems and Software, 82, 112–120.
82. Fitzgerald, J., P. G. Larsen, K. Pierce, M. Verhoef, and S. Wolff, 2010: Collaborative modeling and co-simulation in the development of dependable embedded systems. In Integrated Formal Methods (IFM), Springer-Verlag, vol. LNCS 6396, pp. 12–26. doi:10.1007/978-3-642-16265-7_2.
83. Fitzgerald, J. S., P. G. Larsen, and M. Verhoef, 2008: Vienna development method. In Wiley Encyclopedia of Computer Science and Engineering, John Wiley & Sons, Inc. doi:10.1002/9780470050118.ecse447.
84. Foley, J., A. van Dam, S. Feiner, and J. Hughes, 1996: Computer Graphics, Principles and Practice. Addison-Wesley, 2nd ed.
85. Friedman, D. P. and D. S. Wise, 1976: CONS should not evaluate its arguments. In Third Int. Colloquium on Automata, Languages, and Programming, Edinburg University Press.
86. Fritzson, P., 2003: Principles of Object-Oriented Modeling and Simulation with Modelica 2.1. Wiley.
87. Fuhrmann, H. and R. v. Hanxleden, 2010: Taming graphical modeling. In Model Driven Engineering Languages and Systems (MODELS) 13th International Conference, MODELS 2010, Oslo, Norway, October 3-8, 2010, Springer, Oslo, Norway, vol. 6394, pp. 196–210. doi:10.1007/978-3-642-16145-2_14.

88. Fuhrmann, H. and R. von Hanxleden, 2008: On the pragmatics of model-based design. In Foundations of Computer Software. Future Trends and Techniques for Development – Monterey Workshop, Springer, Budapest, Hungary, vol. LNCS 6028, pp. 116–140. doi:10.1007/978-3-642-12566-9_7.
89. Fujimoto, R., 2000: Parallel and Distributed Simulation Systems. John Wiley and Sons.
90. Gaderer, G., P. Loschmidt, E. G. Cota, J. H. Lewis, J. Serrano, M. Cattin, P. Alvarez, P. M. Oliveira Fernandes Moreira, T. Wlostowski, J. Dedic, C. Prados, M. Kreider, R.Baer, S.Rauch, and T.Fleck, 2009: The white rabbit project. In Int. Conf. on Accelerator and Large Experimental Physics Control Systems, Kobe, Japan.
91. Galletly, J., 1996: Occam-2. University College London Press, 2nd ed.
92. Ganter, B. and R. Wille, 1998: Formal Concept Analysis: Mathematical Foundations Springer-Verlag, Berlin.
93. Geilen, M. and T. Basten, 2003: Requirements on the execution of Kahn process networks. In European Symposium on Programming Languages and Systems, Springer, LNCS, pp. 319–334. Available from: http://www.ics.ele.tue.nl/˜tbasten/papers/esop03.pdf.
94. Geilen, M., T. Basten, and S. Stuijk, 2005: Minimizing buffer requirements of synchronous dataflow graphs with model checking. In Design Automation Conference (DAC), ACM, Anaheim, California, USA, pp. 819–824. doi:10.1145/1065579. 1065796.
95. Geilen, M. and S. Stuijk, 2010: Worst-case performance analysis of synchronous dataflow scenarios. In CODES+ISSS, ACM, Scottsdale, Arizona, USA, pp. 125–134.
96. Girault, A., B. Lee, and E. A. Lee, 1999: Hierarchical finite state machines with multiple concurrency models. IEEE Transactions On Computer-aided Design Of Integrated Circuits And Systems, 18(6), 742–760.
97. Goderis, A., C. Brooks, I. Altintas, E. A. Lee, and C. Goble, 2009: Heterogeneous composition of models of computation. Future Generation Computer Systems, 25(5), 552–560. doi:doi:10.1016/j.future.2008.06.014.
98. Goessler, G. and A. Sangiovanni-Vincentelli, 2002: Compositional modeling in Metropolis. In Second International Workshop on Embedded Software (EMSOFT), Springer-Verlag, Grenoble, France.
99. Golomb, S.W., 1971: Mathematical models: Uses and limitations. IEEE Transactions on Reliability, R-20(3), 130–131. doi:10.1109/TR.1971.5216113.
100. Gu, Z., S. Wang, S. Kodase, and K. G. Shin, 2003: An end-to-end tool chain for multi view modeling and analysis of avionics mission computing software. In Real-Time Systems Symposium (RTSS), pp. 78 – 81.
101. Ha, S. and E. A. Lee, 1991: Compile-time scheduling and assignment of dataflow program graphs with data-dependent iteration. IEEE Transactions on Computers, 40(11), 1225–1238. doi:10.1109/12.102826.

102. Halbwachs, N., P. Caspi, P. Raymond, and D. Pilaud, 1991: The synchronous data flow programming language LUSTRE. Proceedings of the IEEE, 79(9), 1305–1319.
103. Hardebolle, C. and F. Boulanger, 2007: ModHel'X: A component-oriented approach to multi- formalism modeling. In MODELS 2007 Workshop on Multi-Paradigm Modeling , Elsevier Science B.V., Nashville, Tennessee, USA.
104. Harel, D., 1987: Statecharts: A visual formalism for complex systems. Science of Computer Programming, 8(3), 231–274.
105. Harel, D., H. Lachover, A. Naamad, A. Pnueli, M. Politi, R. Sherman, A. Shtull-Trauring, and M. Trakhtenbrot, 1990: STATEMATE: A working environment for the development of complex reactive systems. IEEE Transactions on Software Engineering, 16(4), 403 – 414. doi:10.1109/32.54292.
106. Harel, D. and A. Pnueli, 1985: On the development of reactive systems. In Apt, K. R., ed., Logic and Models for Verification and Specification of Concurrent Systems, Springer-Verlag, vol. F13 of NATO ASI Series, pp. 477–498.
107. Henzinger, T. A., 2000: The theory of hybrid automata. In Inan, M. and R. Kurshan, eds., Verification of Digital and Hybrid Systems, Springer-Verlag, vol. 170 of NATO ASI Series F: Computer and Systems Sciences, pp. 265–292.
108. Henzinger, T. A., B. Horowitz, and C. M. Kirsch, 2001: Giotto: A time-triggered language for embedded programming. In EMSOFT 2001, Springer-Verlag, Tahoe City, CA, vol. LNCS 2211, pp. 166–184.
109. Herrera, F. and E. Villar, 2006: A framework for embedded system specification under different models of computation in System C. In Design Automation Conference (DAC), ACM, San Francisco.
110. Hewitt, C., 1977: Viewing control structures as patterns of passing messages. Journal of Artificial Intelligence, 8(3), 323–363.
111. Hoare, C. A. R., 1978: Communicating sequential processes. Communications of the ACM, 21(8), 666–677.
112. Hop croft, J. and J. Ullman, 1979: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley, Reading, MA.
113. Hsu, C.-J., F. Keceli, M.-Y. Ko, S. Shahparnia, and S. S. Bhattacharyya, 2004: DIF: An interchange format for dataflow-based design tools. In International Workshop on Systems, Architectures, Modeling, and Simulation, Samos, Greece.
114. Hu, T. C., 1961: Parallel sequencing and assembly line problems. Operations Research, 9(6), 841–848.
115. Ingalls, R. G., D. J. Morrice, and A. B. Whinston, 1996: Eliminating canceling edges from the simulation graph model methodology. In WSC '96: Proceedings of the 28th conference on Winter simulation, IEEE Computer Society, Washington, DC, USA, ISBN 0-7803-3383-7, pp. 825–832.
116. Jantsch, A., 2003: Modeling Embedded Systems and SoCs - Concurrency and Time in Models of Computation. Morgan Kaufmann.
117. Jantsch, A. and I. Sander, 2005: Models of computation and languages for embedded system design. IEE Proceedings on Computers and Digital Techniques, 152(2), 114–129.

118. Jefferson, D., 1985: Virtual time. ACM Trans. Programming Languages and Systems, 7(3), 404–425.
119. Johannessen, S., 2004: Time synchronization in a local area network. IEEE Control Systems Magazine, 61–69.
120. Johnston, W. M., J. R. P. Hanna, and R. J. Millar, 2004: Advances in dataflow programming languages. ACM Computing Surveys, 36(1), 1–34.
121. Kahn, G., 1974: The semantics of a simple language for parallel programming. In Proc. of the IFIP Congress 74, North-Holland Publishing Co., pp. 471–475.
122. Kahn, G. and D. B. Macqueen, 1977: Coroutines and networks of parallel processes. In Gilchrist, B., ed., Information Processing, North-Holland Publishing Co., pp. 993–998.
123. Karzai, G., A. Lang, and S. Neema, 2005: Design patterns for open tool integration. Software and Systems Modeling, 4(2), 157–170. doi:10.1007/s10270-004-0073-y.
124. Kay, S. M., 1988: Modern Spectral Estimation: Theory & Application. Prentice-Hall, Englewood Cliffs, NJ.
125. Kiczales, G., J. Lamping, A. Mendhekar, C. Maeda, C. V. Lopes, J.-M. Loingtier, and J. Irwin, 1997: Aspect-oriented programming. In ECOOP, European Conference in Object-Oriented Programming, Springer-Verlag, Finland, vol. LNCS 1241.
126. Kienhuis, B., E. Deprettere, P. van der Wolf, and K. Vissers, 2001: A methodology to design programmable embedded systems. In Deprettere, E., J. Teich, and S. Vassiliadis, eds., Systems, Architectures, Modeling, and Simulation (SAMOS), Springer-Verlag, vol. LNCS 2268.
127. Kodosky, J., J. MacCrisken, and G. Rymar, 1991: Visual programming using structured data flow. In IEEE Workshop on Visual Languages, IEEE Computer Society Press, Kobe, Japan, pp. 34–39.
128. Kopetz, H., 1997: Real-Time Systems : Design Principles for Distributed Embedded Applications. Springer.
129. Kopetz, H. and G. Bauer, 2003: The time-triggered architecture. Proceedings of the IEEE, 91(1), 112–126.
130. Lamport, L., R. Shostak, and M. Pease, 1978: Time, clocks, and the ordering of events in a distributed system. Communications of the ACM, 21(7), 558–565.
131. Landin, P. J., 1965: A correspondence between Algol 60 and Church's lambda notation. Communications of the ACM, 8(2), 89–101.
132. Le Guernic, P., T. Gauthier, M. Le Borgne, and C. Le Maire, 1991: Programming real-time applications with SIGNAL. Proceedings of the IEEE, 9(9), 1321 – 1336. doi: 10.1109/5.97301.
133. Lee, E. A., 1986: A coupled hardware and software architecture for programmable digital signal processors. PhD Thesis UCB/ERL M86/54, University of California. Available from: http://ptolemy.eecs.berkeley.edu/publications/papers/ 86/LeePhDThesis/.
134. —, 1999: Modeling concurrent real-time processes using discrete events. Annals of Software Engineering, 7, 25–45. doi:10.1023/A:1018998524196.

135. —, 2006: The problem with threads. Computer, 39(5), 33–42. doi:10.1109/MC. 2006.180.
136. —, 2008a: Cyber physical systems: Design challenges. In International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC), IEEE, Orlando, Florida, pp. 363 – 369. doi:10.1109/ISORC.2008.25.
137. —, 2008b: Threaded Composite: A mechanism for building concurrent and parallel Ptolemy II models. Technical Report UCB/EECS-2008-151, EECS Department, University of California, Berkeley. Available from: http://www.eecs.berkeley. edu/Pubs/TechRpts/2008/EECS-2008-151.html.
138. —, 2009: Finite state machines and modal models in Ptolemy II. Tech. Rep. UCB/EECS-2009-151, EECS Department, University of California, Berkeley. Available from: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/ EECS-2009-151.html.
139. —, 2010a: CPS foundations. In Design Automation Conference (DAC), ACM, Anaheim, California, USA, pp. 737–742. doi:10.1145/1837274.1837462.
140. —, 2010b: Disciplined heterogeneous modeling. In Petriu, D. C., N. Rouquette, and O. Haugen, eds., Model Driven Engineering, Languages, and Systems (MODELS), IEEE, pp. 273–287. Available from: http://chess.eecs.berkeley.edu/ pubs/679.html.
141. Lee, E. A., E. Goei, H. Heine, and W. Ho, 1989: Gabriel: A design environment for programmable DSPs. In Design Automation Conference (DAC), Las Vegas, NV, pp. 141–146. Available from: http://ptolemy.eecs.berkeley.edu/publications/papers/89/gabriel/.
142. Lee, E. A. and S. Ha, 1989: Scheduling strategies for multiprocessor real-time DSP. In Global Telecommunications Conference (GLOBECOM), vol. 2, pp. 1279 –1283. doi:10.1109/GLOCOM.1989.64160.
143. Lee, E. A., X. Liu, and S. Neuendorffer, 2009a: Classes and inheritance in actor-oriented design. ACM Transactions on Embedded Computing Systems (TECS), 8(4), 29:1– 29:26. doi:10.1145/1550987.1550992.
144. Lee, E. A., S. Matic, S. A. Seshia, and J. Zou, 2009b: The case for timing-centric distributed software. In IEEE International Conference on Distributed Computing Systems Workshops: Workshop on Cyber-Physical Systems, IEEE, Montreal, Canada, pp. 57–64. Available from: http://chess.eecs.berkeley.edu/pubs/607. html.
145. Lee, E. A. and E. Matsikoudis, 2009: The semantics of dataflow with firing. In Huet, G., G. Plotkin, J.-J. L´evy, and Y. Bertot, eds., From Semantics to Computer Science: Essays in memory of Gilles Kahn, Cambridge University Press. Available from: http://ptolemy.eecs.berkeley.edu/publications/papers/08/DataflowWithFiring/.
146. Lee, E. A. and D. G. Messerschmitt, 1987a: Static scheduling of synchronous data flow programs for digital signal processing. IEEE Transactions on Computers, C-36(1), 24– 35. doi:10.1109/TC.1987.5009446.

147. —, 1987b: Synchronous data flow. Proceedings of the IEEE, 75(9), 1235–1245. doi: 10.1109/PROC.1987.13876.
148. Lee, E. A. and S. Neuendorffer, 2000: MoML - a modeling markup language in XML. Tech. Rep. UCB/ERL M00/12, UC Berkeley. Available from: http://ptolemy. eecs.berkeley.edu/publications/papers/00/moml/.
149. Lee, E. A., S. Neuendorffer, and M. J.Wirthlin, 2003: Actor-oriented design of embedded hardware and software systems. Journal of Circuits, Systems, and Computers, 12(3), 231–260. Available from: http://ptolemy.eecs.berkeley.edu/papers/ 03/actorOrientedDesign/.
150. Lee, E. A. and T. M. Parks, 1995: Dataflow process networks. Proceedings of the IEEE, 83(5), 773–801. doi:10.1109/5.381846.
151. Lee, E. A. and A. Sangiovanni-Vincentelli, 1998: A framework for comparing models of computation. IEEE Transactions on Computer-Aided Design of Circuits and Systems,
152. 17(12), 1217–1229. Available from: http://ptolemy.eecs.berkeley.edu/
153. publications/papers/98/framework/.
154. Lee, E. A. and S. A. Seshia, 2011: Introduction to Embedded Systems - A Cyber- Physical Systems Approach. LeeSeshia.org, Berkeley, CA. Available from: http: //LeeSeshia.org.
155. Lee, E. A. and S. Tripakis, 2010: Modal models in Ptolemy. In 3rd International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools (EOOLT), Link¨oping University Electronic Press, Link¨oping University, Oslo, Norway, vol. 47, pp. 11–21. Available from: http://chess.eecs.berkeley.edu/pubs/ 700.html.
156. Lee, E. A. and P. Varaiya, 2011: Structure and Interpretation of Signals and Systems. LeeVaraiya.org, 2nd ed. Available from: http://LeeVaraiya.org.
157. Lee, E. A. and H. Zheng, 2005: Operational semantics of hybrid systems. In Morari, M. and L. Thiele, eds., Hybrid Systems: Computation and Control (HSCC), Springer- Verlag, Zurich, Switzerland, vol. LNCS 3414, pp. 25–53. doi:10.1007/ 978-3-540-31954-2_2.
158. —, 2007: Leveraging synchronous language principles for heterogeneous modeling and design of embedded systems. In EMSOFT, ACM, Salzburg, Austria, pp. 114 – 123. doi:10.1145/1289927.1289949.
159. Leung, M.-K., T. Mandl, E. A. Lee, E. Latronico, C. Shelton, S. Tripakis, and B. Lickly, 2009: Scalable semantic annotation using lattice-based ontologies. In International Conference on Model Driven Engineering Languages and Systems (MODELS), ACM/IEEE, Denver, CO, USA. Available from: http://chess.eecs.berkeley.edu/pubs/611.html.
160. Lickly, B., 2012: Static model analysis with lattice-based ontologies. PhD Thesis Technical Report No. UCB/EECS-2012-212, EECS Department, University of California, Berkeley. Available from: http://www.eecs.berkeley.edu/Pubs/ TechRpts/2012/EECS-2012-212.html.
161. Lickly, B., C. Shelton, E. Latronico, and E. A. Lee, 2011: A practical ontology framework for static model analysis. In International Conference on Embedded

Software (EMSOFT), ACM, pp. 23–32. Available from: http://chess.eecs.berkeley.edu/pubs/862.html.
162. Lin, Y., R. Mullenix, M. Woh, S. Mahlke, T. Mudge, A. Reid, and K. Flautner, 2006: SPEX: A programming language for software defined radio. In Software Defined Radio Technical Conference and Product Exposition, Orlando. Available from: http:// www.eecs.umich.edu/˜sdrg/publications.php.
163. Liskov, B. and S. Zilles, 1974: Programming with abstract data types. ACM Sigplan Notices, 9(4), 50–59. doi:10.1145/942572.807045.
164. Liu, J., B. Wu, X. Liu, and E. A. Lee, 1999: Interoperation of heterogeneous CAD tools in Ptolemy II. In Symposium on Design, Test, and Micro fabrication of MEMS/ MOEMS, Paris, France. Available from: http://ptolemy.eecs.berkeley.edu/publications/papers/99/toolinteraction/.
165. Liu, X. and E. A. Lee, 2008: CPO semantics of timed interactive actor networks. Theoretical Computer Science, 409(1), 110–125. doi:10.1016/j.tcs.2008.08.044.
166. Liu, X., E. Matsikoudis, and E. A. Lee, 2006: Modeling timed concurrent systems. In CONCUR 2006 - Concurrency Theory, Springer, Bonn, Germany, vol. LNCS 4137, pp. 1–15. doi:10.1007/11817949_1.
167. Lynch, N., R. Segala, F. Vaandrager, and H. Weinberg, 1996: Hybrid I/O automata. In Alur, R., T. Henzinger, and E. Sontag, eds., Hybrid Systems III, Springer-Verlag, vol. LNCS 1066, pp. 496–510.
168. Lzaro Cuadrado, D., A. P. Ravn, and P. Koch, 2007: Automated distributed simulation in Ptolemy II. In Parallel and Distributed Computing and Networks (PDCN), Acta Press.
169. Maler, O., Z. Manna, and A. Pnueli, 1992: From timed to hybrid systems. In Real-Time: Theory and Practice, REX Workshop, Springer-Verlag, pp. 447–484.
170. Malik, S., 1994: Analysis of cyclic combinational circuits. IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, 13(7), 950–956.
171. Manna, Z. and A. Pnueli, 1992: The Temporal Logic of Reactive and Concurrent Systems. Springer, Berlin.
172. —, 1993: Verifying hybrid systems. In Hybrid Systems, vol. LNCS 736, pp. 4–35.
173. Maraninchi, F. and T. Bhouhadiba, 2007: 42: Programmable models of computation for a component-based approach to heterogeneous embedded systems. In 6th ACM International Conference on Generative Programming and Component Engineering (GPCE), Salzburg, Austria, pp. 1–3.
174. Maraninchi, F. and Y. R´emond, 2001: Argos: an automaton-based synchronous language. Computer Languages, (27), 61–92.
175. Matic, S., I. Akkaya, M. Zimmer, J. C. Eidson, and E. A. Lee, 2011: Ptides model on a distributed test bed emulating smart grid real-time applications. In Innovative Smart
176. Grid Technologies (ISGT-EUROPE), IEEE, Manchester, UK. Available from: http: //chess.eecs.berkeley.edu/pubs/857.html.

177. Matsikoudis, E., C. Stergiou, and E. A. Lee, 2013: On the schedulability of real-time discrete-event systems. In International Conference on Embedded Software (EMSOFT), ACM, Montreal, Canada.
178. Matthews, S. G., 1995: An extensional treatment of lazy data flow deadlock. Theoretical Computer Science, 151(1), 195–205.
179. Messerschmitt, D. G., 1984: A tool for structured functional simulation. IEEE Journal on Selected Areas in Communications, SAC-2(1).
180. Mills, D. L., 2003: A brief history of NTP time: confessions of an internet timekeeper. ACM Computer Communications Review, 33.
181. Milner, R., 1978: A theory of type polymorphism in programming. Journal of Computer and System Sciences, 17, 348–375.
182. —, 1980: A Calculus of Communicating Systems, vol. 92 of Lecture Notes in Computer Science. Springer.
183. Misra, J., 1986: Distributed discrete event simulation. ACM Computing Surveys, 18(1), 39–65.
184. Modelica Association, 2009: Modelica R - a unified object-oriented language for physical systems modeling: Language specification version 3.1. Report. Available from: http: //www.Modelica.org.
185. Moir, I. and A. Seabridge, 2008: Aircraft Systems: Mechanical, Electrical, and Avionics Subsystems Integration. AIAA Education Series, Wiley, third edition ed.
186. Moreira, O., T. Basten, M. Geilen, and S. Stuijk, 2010: Buffer sizing for rate-optimal single-rate dataflow scheduling revisited. IEEE Transactions on Computers, 59(2), 188–201. doi:10.1109/TC.2009.155.
187. Morris, J. H. and P. Henderson, 1976: A lazy evaluator. In Conference on the Principles of Programming Languages (POPL), ACM.
188. Mosterman, P. J. and H. Vangheluwe, 2004: Computer automated multi-paradigm modeling: An introduction. Simulation: Transactions of the Society for Modeling and Simulation International Journal of High Performance Computing Applications, 80(9), 433– 450.
189. Motika, C., H. Fuhrmann, and R. v. Hanxleden, 2010: Semantics and execution of domain specific models. In Workshop Methodische Entwicklung von Modellierungswerkzeugen (MEMWe 2010) at conference INFORMATIK 2010, Bonner K¨ollen Verlag, Leipzig, Germany, vol. GI-Edition – Lecture Notes in Informatics (LNI),.
190. Murata, T., 1989: Petri nets: Properties, analysis and applications. Proceedings of IEEE, 77(4), 541–580. doi:10.1109/5.24143.
191. Murthy, P. K. and S. S. Bhattacharyya, 2006: Memory Management for Synthesis of DSP Software. CRC Press.
192. Murthy, P. K. and E. A. Lee, 2002: Multidimensional synchronous dataflow. IEEE Transactions on Signal Processing, 50(8), 2064–2079. doi:10.1109/TSP.2002. 800830.
193. Object Management Group (OMG), 2007: A UML profile for MARTE, beta 1. OMG Adopted Specification ptc/07-08-04, OMG. Available from: http://www.omg. org/omgmarte/.

194. —, 2008a: System modeling language specification v1.1. Tech. rep., OMG. Available from: http://www.sysmlforum.com.
195. —, 2008b: A UML profile for MARTE, beta 2. OMG Adopted Specification ptc/08-06- 09, OMG. Available from: http://www.omg.org/omgmarte/.
196. Olson, A. G. and B. L. Evans, 2005: Deadlock detection for distributed process networks. In ICASSP.
197. Parks, T. M., 1995: Bounded scheduling of process networks. Ph.D. Thesis Tech. Report
198. UCB/ERL M95/105, UC Berkeley. Available from: http://ptolemy.eecs.berkeley.edu/papers/95/parksThesis .
199. Parks, T. M. and D. Roberts, 2003: Distributed process networks in Java. In International Parallel and Distributed Processing Symposium, Nice, France.
200. Patel, H. D. and S. K. Shukla, 2004: SystemC Kernel Extensions for Heterogeneous System Modeling. Kluwer.
201. Pino, J. L., T. M. Parks, and E. A. Lee, 1994: Automatic code generation for heterogeneous multiprocessors. In International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Adelaide, Australia, pp. 445–448. doi:10.1109/ICASSP. 1994.389626.
202. Pree, W. and J. Templ, 2006: Modeling with the timing definition language (TDL). In Automotive Software Workshop San Diego (ASWSD) on Model-Driven Development of Reliable Automotive Services, Springer, San Diego, CA, LNCS.
203. Press,W. H., S. Teukolsky,W. T. Vetterling, and B. P. Flannery, 1992: Numerical Recipes in C: the Art of Scientific Computing. Cambridge University Press.
204. Prochnow, S. and R. von Hanxleden, 2007: Statechart development beyond WYSIWYG. In International Conference on Model Driven Engineering Languages and Systems (MoDELS), ACM/IEEE, Nashville, TN, USA.
205. Ramadge, P. and W. Wonham, 1989: The control of discrete event systems. Proceedings of the IEEE, 77(1), 81–98.
206. Reed, G. M. and A.W. Roscoe, 1988: Metric spaces as models for real-time concurrency. In 3rd Workshop on Mathematical Foundations of Programming Language Semantics, London, UK, pp. 331–343.
207. Rehof, J. and T. A. Mogensen, 1996: Tractable constraints in finite semilattices. In SAS ’96: Proceedings of the Third International Symposium on Static Analysis, Springer- Verlag, London, UK, ISBN 3-540-61739-6, pp. 285–300.
208. Rehof, J. and T. . Mogensen, 1999: Tractable constraints in finite semilattices. Science of Computer Programming, 35(2-3), 191–221.
209. Ritchie, D. M. and K. L. Thompson, 1974: The UNIX time-sharing system. Communications of the ACM, 17(7), 365 – 375.
210. Rodiers, B. and B. Lickly, 2010: Width inference documentation. Technical Report UCB/EECS-2010-120, EECS Department, University of California. Available from: http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/ EECS-2010-120.html.

211. Sander, I. and A. Jantsch, 2004: System modeling and transformational design refinement in ForSyDe. IEEE Transactions on Computer-Aided Design of Circuits and Systems, 23(1), 17–32.
212. Schruben, L. W., 1983: Simulation modeling with event graphs. Communications of the ACM, 26(11), 957–963.
213. —, 1995: Building reusable simulators using hierarchical event graphs. In Winter Simulation Conference (WSC 95), IEEE Computer Society, Los Alamitos, CA, USA, ISBN 0-7803-3018-8, pp. 472–475.
214. Shapiro, F. R., 2006: The Yale Book of Quotations. Yale University Press.
215. Sih, G. C. and E. A. Lee, 1993a: A compile-time scheduling heuristic for interconnection constrained heterogeneous processor architectures. IEEE Transactions on Parallel and Distributed Systems, 4(2), 175–187. doi:10.1109/71.207593.
216. —, 1993b: Declustering : A new multiprocessor scheduling technique. IEEE Transactions on Parallel and Distributed Systems, 4(6), 625–637. doi:10.1109/71.242160.
217. Simitci, H., 2003: Storage Network Performance Analytics. Wiley.
218. Smith, N. K., 1929: Immanuel Kant's Critique of Pure Reason. Macmillan and Co. Available from: http://www.hkbu.edu.hk/˜ppp/cpr/toc.html.
219. Som, T. K. and R. G. Sargent, 1989: A formal development of event graph models as an aid to structured and efficient simulation programs. ORSA Journal on Computing, 1(2), 107–125.
220. Sp¨onemann, M., H. Fuhrmann, R. v. Hanxleden, and P. Mutzel, 2009: Port constraints in hierarchical layout of data flow diagrams. In 17th International Symposium on Graph Drawing (GD), Springer, Chicago, IL, USA, vol. LNCS. Available from: http://rtsys.informatik.uni-kiel.de/˜biblio/ downloads/papers/gd09.pdf.
221. Srini, V., 1986: An architectural comparison of dataflow systems. Computer, 19(3).
222. Sriram, S. and S. S. Bhattacharyya, 2009: Embedded Multiprocessors: Scheduling and Synchronization. CRC press, 2nd ed.
223. Stark, E. W., 1995: An algebra of dataflow networks. Fundamenta Informaticae, 22(1-2), 167–185.
224. Stephens, R., 1997: A survey of stream processing. Acta Informatica, 34(7).
225. Stuijk, S., M. C. Geilen, and T. Basten, 2008: Throughput-buffering trade-off exploration for cyclo-static and synchronous dataflow graphs. IEEE Transactions on Computers, 57(10), 1331–1345. doi:10.1109/TC.2008.58.
226. Thies, W., M. Karczmarek, and S. Amarasinghe, 2002: StreamIt: A language for streaming applications. In 11th International Conference on Compiler Construction, Springer-Verlag, Grenoble, France, vol. LNCS 2304. doi:10.1007/3-540-45937-5_14.
227. Thies, W., M. Karczmarek, J. Sermulins, R. Rabbah, and S. Amarasinghe, 2005: Teleport messaging for distributed stream programs. In Principles and

Practice of Parallel Programming (PPoPP), ACM, Chicago, USA. doi:10.1145/1065944.1065975.

228. Tripakis, S., C. Stergiou, C. Shaver, and E. A. Lee, 2013: A modular formal semantics for Ptolemy. Mathematical Structures in Computer Science, 23, 834–881. Available from: http://chess.eecs.berkeley.edu/pubs/999.html, doi:10.1017/S0960129512000278.

229. Turjan, A., B. Kienhuis, and E. Deprettere, 2003: Solving out-of-order communication in Kahn process networks. Journal on VLSI Signal Processing-Systems for Signal, Image, and Video Technology, 40, 7 – 18. doi:10.1007/s11265-005-4935-5. University of Pennsylvania MoBIES team, 2002: HSIF semantics (version 3, synchronous edition). Tech. Rep. Report, University of Pennsylvania.

230. von der Beeck, M., 1994: A comparison of Statecharts variants. In Langmaack, H., W. P. de Roever, and J. Vytopil, eds., Third International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems, Springer-Verlag, L¨ubeck, Germany, vol. 863 of Lecture Notes in Computer Science, pp. 128–148.

231. von Hanxleden, R., 2009: Sync Charts in C - A proposal for light-weight deterministic concurrency. In ACM Embedded Software Conference (EMSOFT), pp. 11–16. doi: 10.1145/1629335.1629366.

232. Wiener, N., 1948: Cybernetics: Or Control and Communication in the Animal and the Machine. Librairie Hermann & Cie, Paris, and MIT Press. Cambridge, MA.

233. Xiong, Y., 2002: An extensible type system for component-based design. Ph.D. Thesis Technical Memorandum UCB/ERL M02/13, University of California, Berkeley, CA 94720. Available from: http://ptolemy.eecs.berkeley.edu/papers/ 02/type System.

234. Yates, R. K., 1993: Networks of real-time processes. In Best, E., ed., Proc. of the 4th Int. Conf. on Concurrency Theory (CONCUR), Springer-Verlag, vol. LNCS 715.

235. Zeigler, B., 1976: Theory of Modeling and Simulation. Wiley Interscience, New York.

236. Zeigler, B. P., H. Praehofer, and T. G. Kim, 2000: Theory of Modeling and Simulation. Academic Press, 2nd ed.

237. Zhao, Y., 2009: On the design of concurrent, distributed real-time systems. Ph.D. Thesis Technical Report UCB/EECS-2009-117, EECS Department, UC Berkeley. Available from: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-117.html.

238. Zhao, Y., E. A. Lee, and J. Liu, 2007: A programming model for time-synchronized distributed real-time systems. In Real-Time and Embedded Technology and Applications Symposium (RTAS), IEEE, Bellevue, WA, USA, pp. 259 – 268. doi:10.1109/ RTAS.2007.5.

239. Zou, J., 2011: From ptides to ptidyos, designing distributed real-time embedded systems. PhD Dissertation Technical Report UCB/EECS-2011-53, UC

Berkeley. Available from: http://www.eecs.berkeley.edu/Pubs/TechRpts/2011/EECS-2011-53.html.

240. Zou, J., J. Auerbach, D. F. Bacon, and E. A. Lee, 2009a: PTIDES on flexible task graph: Real-time embedded system building from theory to practice. In Languages, Compilers, and Tools for Embedded Systems (LCTES), ACM, Dublin, Ireland. Available from: http://chess.eecs.berkeley.edu/pubs/531.html.

241. Zou, J., S. Matic, E. A. Lee, T. H. Feng, and P. Derler, 2009b: Execution strategies for Ptides, a programming model for distributed embedded systems. In Real-Time and Embedded Technology and Applications Symposium (RTAS), IEEE, San Francisco, CA. Available from: http://chess.eecs.berkeley.edu/pubs/529.html.

AUTHOR

Osama Mohammed Elmardi Suleiman Khayal was born in Atbara, Sudan in 1966. He received his diploma degree in mechanical engineering from Mechanical Engineering College, Atbara, Sudan in 1990. He also received a bachelor degree in mechanical engineering from Sudan University of science and technology – Faculty of engineering in 1998, and a master degree in solid mechanics from Nile valley university (Atbara, Sudan) in 2003, and a PhD in structural engineering in 2017. He contributed in teaching some subjects in other universities such as Red Sea University (Port Sudan, Sudan), Kordofan University (Obayed, Sudan), Sudan University of Science and Technology (Khartoum, Sudan), Blue Nile University (Damazin, Sudan) and Kassala University (Kassala, Sudan). In addition, he supervised more than three hundred under graduate studies in diploma and B.Sc. levels and about thirty master theses. The author wrote about fifty engineering books written in Arabic language, and twenty books written in English language and more than hundred research papers in fluid mechanics, thermodynamics, internal combustion engines and analysis of composite structures. He is currently an associated professor in Department of Mechanical Engineering, Faculty of Engineering and Technology, Nile Valley University Atbara, Sudan. His research interest and favorite subjects include structural mechanics, applied mechanics, control engineering and instrumentation, computer aided design, design of mechanical elements, fluid mechanics and dynamics, heat and mass transfer and hydraulic machinery. The author also works as a technical manager and superintendent of Al – Kamali mechanical and production workshops group which specializes in small, medium and large automotive overhaul maintenance and which situated in Atbara town in the north part of Sudan, River Nile State