

References for Erasmus

Peter Grogono and Brian Shearing

Sunday 26th April, 2009

References

- [1] Luca Aceto, Anna Ingólfssdóttir, Kim Guldstrand Larsen, and Jiří Srba. *Reactive Systems: Modelling, Specification and Verification*. Cambridge University Press, 2007.
- [2] Ada. Ada 95 Reference Manual. Revised International Standard ISO/IEC 8652:1995, 1995. www.adahome.com/rm95. Accessed 2008/03/12.
- [3] Andrei Alexandrescu, Hans Boehm, Kevlin Henney, Ben Hutchings, Doug Lea, and Bill Pugh. Memory model for multithreaded C++: issues. Document Number: WG21/N1777=J16/05-0037, March 2005. www.open-std.org/jtc1/sc22/wg21/docs/papers/2004/n1680.pdf.
- [4] Andrei Alexandrescu, Hans Boehm, Kevlin Henney, Doug Lea, and Bill Pugh. Memory model for multithreaded C++. Document Number: WG21/N1680=J16/04-0120, September 2004.
- [5] Sten Andler. Predicate path expressions. In *POPL '79: Proceedings of the 6th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 226–236, New York, NY, USA, 1979. ACM Press.
- [6] Gregory R. Andrews. *Concurrent Programming: Principles and Practice*. Addison-Wesley, 1991.
- [7] Gregory R. Andrews. *Foundations and Multithreaded, Parallel, and Distributed Programming*. Addison-Wesley, 2000.
- [8] Andrew W. Appel. *Modern Compiler Implementation in Java*. Cambridge University Press, second edition, 2002.
- [9] Architecture Board ORMSC. Model driven architecture (mda). Technical Report ormsc/2001-07-01, OMG, 2001. Available at www.omg.org.
- [10] J. Armstrong, M. Williams, C. Wikström, and R. Verding. *Concurrent Programming in Erlang*. Prentice Hall, 1996.
- [11] Joe Armstrong. The development of Erlang. In *ICFP '97: Proceedings of the Second ACM SIGPLAN International Conference on Functional Programming*, pages 196–203, New York, NY, USA, 1997. ACM Press.
- [12] Joe Armstrong. Getting Erlang to talk to the outside world. In *ERLANG '02: Proceedings of the 2002 ACM SIGPLAN workshop on Erlang*, pages 64–72, New York, NY, USA, 2002. ACM Press.

- [13] Joe Armstrong. A history of Erlang. In *HOPL III: Proceedings of the Third ACM SIGPLAN Conference on the History of Programming Languages*, pages 6.1–6.26, New York, NY, USA, 2007. ACM Press.
- [14] Joe Armstrong. *Programming Erlang: Software for a Concurrent World*. The Pragmatic Bookshelf, 2007.
- [15] K. Arnold and J. Gosling. *The Java Programming Language*. Addison-Wesley, 1996.
- [16] Krste Asanovic et al. The landscape of parallel computing research: A view from Berkeley. Technical Report UCB/EECS-2006-183, EECS Department, University of California, Berkeley, December 18 2006.
- [17] Peter Atkins. *Galileo's Finger*. Oxford University Press, 2003.
- [18] John Aycock. A brief history of just-in-time. *ACM Computing Surveys*, 35(2):97–113, 2003.
- [19] Ozalp Babaoglu, Geoffrey Canright, Andreas Deutsch, Gianni A. Di Caro, Frederick Ducatelle, Luca M. Gambardella, Niloy Ganguly, Márk Jelasity, Roberto Montemanni, Alberto Montresor, and Tore Urnes. Design patterns from biology for distributed computing. *ACM Trans. Auton. Adapt. Syst.*, 1(1):26–66, 2006.
- [20] David F. Bacon, Perry Cheng, and V. T. Rajan. A unified theory of garbage collection. In *OOPSLA '04: Proceedings of the 19th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, pages 50–68, New York, NY, USA, 2004. ACM.
- [21] J.C.M. Baeten. A brief history of process algebra, ??
- [22] Rajive Bagrodia. Synchronization of asynchronous processes in CSP. *ACM Trans. Program. Lang. Syst.*, 11(4):585–597, 1989.
- [23] S. Bailliez et al. *Apache Ant*. The Jakarta Project, 2000. jakarta.apache.org/ant/-index.html.
- [24] Henri E. Bal, Jennifer G. Steiner, and Andrew S. Tanenbaum. Programming languages for distributed computing systems. *ACM Computing Surveys*, 21(3):261–322, 1989.
- [25] Purushotham V. Bangalore. Generating parallel applications for distributed memory systems using aspects, components, and patterns. In *ACP4IS '07: Proceedings of the 6th workshop on Aspects, components, and patterns for infrastructure software*, page 3, New York, NY, USA, 2007. ACM Press.
- [26] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, October 1999.
- [27] Fred R.M. Barnes and Peter H. Welch. Communicating mobile processes. In Ian East, Jeremy Martin, Peter Welch, David Duce, and Mark Green, editors, *Communicating Process Architectures*, pages 201–218. IOS Press, 2004.
- [28] Fred R.M. Barnes and Peter H. Welch. Occam- π : blending the best of CSP and the π -calculus. www.cs.kent.ac.uk/projects/ofa/kroc. Accessed 2008/03/13., 2006.
- [29] Gareth Baxter, Marcus Frean, James Noble, Mark Rickerby, Hayden Smith, Matt Visser, Hayden Melton, and Ewan Tempero. Understanding the shape of Java software. In *OOPSLA '06: Proceedings of the 21st annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications*, pages 397–412, New York, NY, USA, 2006. ACM Press.

- [30] K. Beck and E. Gamma. JUnit test infected: Programmers love writing tests. www.junit.org, 2000.
- [31] M. Ben-Ari. *Principles of Concurrent and Distributed Programming*. Addison-Wesley, second edition, 2006.
- [32] Mordechai Ben-Ari. Algorithms for on-the-fly garbage collection. *ACM Transactions on Programming Languages and Systems*, 6(3):333–344, 1984.
- [33] Nick Benton, Luca Cardelli, and Cédric Fournet. Modern concurrency abstractions for C#. *ACM Transactions on Programming Languages and Systems*, 26(5):769–804, September 2004.
- [34] Aysu Betin-Can, Tevfik Bultan, Mikael Lindvall, Benjamin Lux, and Stefan Topp. Application of design for verification with concurrency controllers to air traffic control software. In *ASE '05: Proceedings of the 20th IEEE/ACM International Conference on Automated software engineering*, pages 14–23, 2005.
- [35] Kevin Bierhoff and Jonathan Aldrich. Modular typestate checking of aliased objects. In *OOPSLA '07: Proceedings of the 22nd annual ACM SIGPLAN conference on Object oriented programming systems and applications*, pages 301–320, New York, NY, USA, 2007. ACM.
- [36] Gavin M. Bierman, Erik Meijer, and Mads Torgersen. Lost in translation: formalizing proposed extensions to C#. In *OOPSLA '07: Proceedings of the 22nd annual ACM SIGPLAN conference on Object oriented programming systems and applications*, pages 479–498, New York, NY, USA, 2007. ACM.
- [37] Andrew P. Black, Jie Huang, Rainer Koster, Jonathan Walpole, and Calton Pu. Infopipes: An abstraction for multimedia streaming. *Multimedia Systems*, 8:406–419, 2002.
- [38] Jasmin Christian Blanchette. Overview of the Creol language, May 2007.
- [39] G Blaschek. *Object-Oriented Programming with Prototypes*. Springer, 1994.
- [40] Conrad Bock. UML 2 activity and action models. *Journal of Object Technology (www.jot.fm)*, 2(4):43–53, July–August 2003. www.jot.fm/issues/issue_2003_07/-column3.
- [41] Conrad Bock. UML 2 activity and action models part 2: Actions. *Journal of Object Technology (www.jot.fm)*, 2(5):41–56, September 2003. www.jot.fm/issues/-issue_2003_09/column4.
- [42] Conrad Bock. UML 2 activity and action models part 3: Control nodes. *Journal of Object Technology (www.jot.fm)*, 2(6):7–23, November 2003. www.jot.fm/issues/-issue_2003_11/column1.
- [43] Barry Boehm. A view of 20th and 21st century software engineering. In *ICSE '06: Proceeding of the 28th international conference on Software engineering*, pages 12–29, New York, NY, USA, 2006. ACM.
- [44] Hans-J. Boehm. Threads cannot be implemented as a library. In *Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '05)*, pages 261–268, 2005.
- [45] Hans-J. Boehm, Russ Atkinson, and Michael Plass. Ropes: an alternative to strings. *Software—Practice and Experience*, 25(12):1315–1330, December 1995.
- [46] Gilad Bracha. JSR-000014 Adding Generics to the Java Programming Language. Java Specification Requests, August 2001. www.jcp.org/en/jsr/detail?id=14.
- [47] Per Brinch Hansen. *Operating System Principles*. Prentice Hall, 1973.

- [48] Per Brinch Hansen. The programming language Concurrent Pascal. *IEEE Transactions on Software Engineering*, 1(2):199–207, June 1975.
- [49] Per Brinch Hansen. The design of Edison. *Software—Practice and Experience*, 11(4):363–396, April 1981.
- [50] Per Brinch Hansen. Joyce—a programming language for distributed systems. *Software—Practice and Experience*, 17(1):29–50, January 1987.
- [51] Per Brinch Hansen. A multiprocessor implementation of Joyce. *Software—Practice and Experience*, 19(6):579–592, June 1989.
- [52] Per Brinch Hansen. Monitors and Concurrent Pascal: A personal history. In *HOPL-II: The second ACM Conference on the History of Programming Languages*, pages 1–35. ACM Press, April 1993.
- [53] Per Brinch Hansen. SuperPascal — a publication language for parallel scientific computing. *Concurrency—Practice and Experience*, 6(5):461–483, August 1994.
- [54] Per Brinch Hansen. Efficient parallel recursion. *ACM SIGPLAN Notices*, 30(12):9–16, December 1995.
- [55] Per Brinch Hansen. *The Search for Simplicity—Essays in Parallel Programming*. IEEE Computer Society Press, 1996.
- [56] Per Brinch Hansen. Java’s insecure parallelism. *ACM SIGPLAN Notices*, 34(4):38–45, April 1999.
- [57] Philip J. Brooke and Richard F. Paige. Exceptions in concurrent Eiffel. *Journal of Object Technology (www.jot.fm)*, 6(10):111–126, November-December 2007.
- [58] Frederick P. Brooks, Jr. *The Mythical Man-Month*. Addison-Wesley, 1975.
- [59] Frederick P. Brooks, Jr. No silver bullet — essence and accident in software engineering. *IEEE Computer*, 20(4):10–19, April 1987.
- [60] Frederick P. Brooks, Jr. *The Mythical Man-Month*. Addison-Wesley, anniversary edition, 1995.
- [61] Manfred Broy and Ketil Stølen. *Specification and Development of Interactive Systems*. Springer, 2001.
- [62] Mark Brunelli. Question & Answer: Smalltalk with object-oriented programming pioneer Kay. searchsoa.techtarget.com/news/interview/0,289202,-sid26_gci962762,00.html. Accessed 2008/03/25, 2004.
- [63] P. A. Buhr, David Till, and C. R. Zarnke. Assignment as the sole means of updating objects. *Software—Practice and Experience*, 24(9):838–870, September 1994.
- [64] Peter A. Buhr, Michel Fortier, and Michael H. Coffin. Monitor classification. *ACM Computing Surveys*, 27(1):63–107, 1995.
- [65] Peter A. Buhr and Ashif S. Harji. Implicit-signal monitors. *ACM Trans. Program. Lang. Syst.*, 27(6):1270–1343, 2005.
- [66] Mark Burgess. An approach to understanding policy based on autonomy and voluntary cooperation. In *Ambient Networks: Proc. 16th IFIP/IEEE Workshop on Distributed Systems: Operations and Management*, LNCS 3775, pages 97–108. Springer, 2005.
- [67] Mark Burgess and Siri Fagernes. Promise theory — a model of autonomous objects for pervasive computing and swarms. In *ICNS ’06: Proceedings of the International Conference on Networking and Services*, page 118, Washington, DC, USA, 2006. IEEE Computer Society.

- [68] Alan Burns and Andy Wellings. *Concurrency in Ada*. Cambridge University Press, second edition, 1998.
- [69] Rod Burstall. Christopher Strachey—understanding programming languages. *Higher-Order and Symbolic Computation*, 13(1-2):51–55, 2000.
- [70] J.N. Buxton, P. Naur, and B. Randell. *Software Engineering*. Petrocelli, 1975. Report on two NATO Conferences held in Garmisch, Germany (October 1968) and Rome, Italy (October 1969).
- [71] Brian Cabana, Suad Alagić, and Jeff Faulkner. Parametric polymorphism for Java: Is there any hope in sight? *ACM SIGPLAN Notices*, 39(12):22–31, December 2004.
- [72] R. H. Campbell and R. B. Kolstad. Practical applications of Path Pascal in systems programming. In *ACM 79: Proceedings of the 1979 annual conference*, pages 81–87, New York, NY, USA, 1979. ACM Press.
- [73] R.H. Campbell and A.N. Habermann. The specification of process synchronization by path expressions. In G. Goos and J. Hartmanis, editors, *Lecture Notes in Computer Science*, volume 16, pages 89–102. Springer, 1974.
- [74] Roy H. Campbell and Robert B. Kolstad. Path expressions in Pascal. In *ICSE '79: Proceedings of the 4th international conference on Software engineering*, pages 212–219, Piscataway, NJ, USA, 1979. IEEE Press.
- [75] Roy H. Campbell and Robert B. Kolstad. An overview of Path Pascal’s design. *ACM SIGPLAN Notices*, 15(9):13–14, 1980.
- [76] P. Canning, W. Cook, W. Hill, W. Olthoff, and J. Mitchell. F-bounded polymorphism for object-oriented programming. In *Proceedings of the Fourth International Conference on Programming Languages and Architecture*, pages 273–280, 1989.
- [77] Richard Carlsson, Konstantinos Sagonas, and Jesper Wilhelmsson. Message analysis for concurrent programs using message passing. *ACM Transactions on Programming Languages and Systems*, 28(4):715–746, 2006.
- [78] Denis Caromel, Ludovic Henrio, and Bernard Paul Serpette. Asynchronous and deterministic objects. In *POPL '04: Proceedings of the 31st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 123–134, New York, NY, USA, 2004. ACM Press.
- [79] Bernard Carré and Jonathan Garnsworthy. SPARK — an annotated Ada subset for safety-critical programming. In *Proceedings of TRI-ADA '90 Conference*, pages 392–402. ACM Press, 1990.
- [80] Nicholas Carriero and David Gelernter. How to write parallel programs: a guide to the perplexed. *ACM Computing Surveys*, 21(3):323–357, 1989.
- [81] Giuseppe Castagna. Covariance and contravariance: Conflict without a cause. *ACM Transactions on Programming Languages and Systems*, 17(3):431–447, May 1995.
- [82] Ernest Chang and Rosemary Roberts. An improved algorithm for decentralized extrema-finding in circular configurations of processes. *Commun. ACM*, 22(5):281–283, 1979.
- [83] Douglas W. Clark and C. Cordell Green. An empirical study of list structure in Lisp. *Communications of the ACM*, 20(2):78–87, 1977.

- [84] Dave Clarke, Sophia Drossopoulou, James Noble, and Tobias Wrigstad. Tribe: a simple virtual class calculus. In *AOSD '07: Proceedings of the 6th international conference on Aspect-oriented software development*, pages 121–134, New York, NY, USA, 2007. ACM.
- [85] Christian Collberg, Ginger Myles, and Michael Stepp. An empirical study of Java bytecode programs. *Software—Practice and Experience*, 37(6):581–641, May 2007.
- [86] Michael Compton and Richard Walker. A run-time system for SCOOP. *Journal of Object Technology (www.jot.fm)*, 1(3):119–157, 2007. Special issue: TOOLS USA 2002 Proceedings.
- [87] Larry Constantine and Ed Yourdon. *Structured Design*. Prentice Hall, 1979.
- [88] Melvin E. Conway. Design of a separable transition-diagram compiler. *Communications of the ACM*, 6(7):396–408, 1963.
- [89] W. Cook. *A denotational semantics of inheritance*. PhD thesis, Brown University, 1989.
- [90] William R. Cook and Siddhartha Rai. Safe query objects: statically typed objects as remotely executable queries. In *ICSE '05: Proceedings of the 27th international conference on Software engineering*, pages 97–106, New York, NY, USA, 2005. ACM.
- [91] George Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed Systems: Concepts and Design*. Addison-Wesley, third edition, 2001.
- [92] David E. Culler, Andrea C. Arpaci-Dusseau, Seth Copen Goldstein, Arvind Krishnamurthy, Steven Lumetta, Thorsten von Eicken, and Katherine A. Yelick. Parallel programming in Split-C. In *Supercomputing*, pages 262–273, 1993.
- [93] Carlos A. Cunha, ao L. Sobral Jo and Miguel P. Monteiro. Reusable aspect-oriented implementations of concurrency patterns and mechanisms. In *AOSD '06: Proceedings of the 5th international conference on Aspect-oriented software development*, pages 134–145, New York, NY, USA, 2006. ACM Press.
- [94] O.-J. Dahl, E.W. Dijkstra, and C.A.R. Hoare. *Structured Programming*. Academic Press, 1972.
- [95] Willem-Paul de Roever et al. *Concurrency Verification: Introduction to Compositional and Noncompositional Methods*. Cambridge University Press, 2001.
- [96] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified data processing on large clusters. In *OSDI'04: Sixth Symposium on Operating System Design and Implementation*, December 2004.
- [97] Daniel C. Dennett. *The Intentional Stance*. MIT Press, 1987.
- [98] Daniel C. Dennett. *Darwin's Dangerous Idea*. Simon & Schuster, 1995.
- [99] Edsger W. Dijkstra. Cooperating sequential processes. In F. Genuys, editor, *Programming Languages: NATO Advanced Study Institute*, pages 43–112. Academic Press, 1968.
- [100] Edsger W. Dijkstra. Hierarchical ordering of sequential processes. *Acta Informatica*, 1(2):115–138, June 1971. Reprinted in [185].
- [101] E.W. Dijkstra. Cooperating sequential processes. Technical Report Technical Report EWD-123, Technological University, Eindhoven, The Netherlands, 1965. Reprinted in [130].

- [102] Elfriede Dustin, Jeff Rashka, and John Paul. *Automated Software Testing: Introduction, Management, and Performance*. Addison-Wesley, 1999.
- [103] Miguel Oliveira e Silva. Concurrent object-oriented programming: The MP-Eiffel approach. *Journal of Object Technology (www.jot.fm)*, 3(4):97–124, April 2004. TOOLS USA 2003.
- [104] Jonathan Edwards. No ifs, ands, or buts: uncovering the simplicity of conditionals. In *OOPSLA '07: Proceedings of the 22nd annual ACM SIGPLAN conference on Object oriented programming systems and applications*, pages 639–658, New York, NY, USA, 2007. ACM.
- [105] Stephen A. Edwards and Olivier Tardieu. Efficient code generation from shim models. In *LCTES '06: Proceedings of the 2006 ACM SIGPLAN/SIGBED conference on Language, compilers and tool support for embedded systems*, pages 125–134, New York, NY, USA, 2006. ACM Press.
- [106] Craig Eichelkraut and Letha Etzkorn. Describing agent based real-time distributed systems using design patterns. In *ACM-SE 45: Proceedings of the 45th annual southeast regional conference*, pages 156–161, New York, NY, USA, 2007. ACM Press.
- [107] Susan Eisenbach and Chris Sadler. Reuse and abuse. *Journal of Object Technology (www.jot.fm)*, 6(1):139–167, January-February 2007.
- [108] J. Eker, J.W. Janneck, E.A. Lee, Jie Liu, Xiaojun Liu, J. Ludvig, S. Neuendorfer, S. Sachs, and Yuhong Xiong. Taming heterogeneity—the Ptolemy approach. *Proceedings of the IEEE*, 91(1):127–144, January 2003.
- [109] Torbjörn Ekman, Peter Mechlenborg, and Ulrik Pagh Schultz. Flexible language interoperability. *Journal of Object Technology (www.jot.fm)*, 6(8):95–116, 2007.
- [110] A. Endres. IBM Böblingen’s early software contributions. *IEEE Annals of the History of Computing*, 26(3):31–41, 2004.
- [111] Ted Faison. Interaction patterns for communicating processes. In *PLoP-98*, August 1998.
- [112] Ted Faison. *Event-Based Programming: Taking Events to the Limit*. APress, 2006.
- [113] Reedy Feggins. Designing component-based architectures with Rational Rose RealTime. <http://www.ibm.com/developerworks/rational/library/797.html>. Accessed 2008/03/25., 2003.
- [114] Stuart Feldman. A conversation with Alan Kay. *Queue*, 2(9):20–30, 2005.
- [115] Xinyu Feng. Local rely-guarantee reasoning. In *Proc. 36th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'09)*, January 2009. To appear.
- [116] Robert E. Filman, Tzilla Elrad, Siobhan Clarke, and Mehmet Aksit. *Aspect-Oriented Software Development*. Addison-Wesley, 2004.
- [117] Vincenzo De Florio and Chris Blondia. A survey of linguistic structures for application-level fault tolerance. *ACM Computing Surveys*, 40(2):1–37, 2008.
- [118] Robert W. Floyd. The paradigms of programming. *Communications of the ACM*, 22(8):455–460, 1979. Turing Award acceptance speech.
- [119] Philip W.L. Fong. Discretionary capability confinement. Technical report, University of Regina, Regina, Saskatchewan, Canada, July 2006.

- [120] Bryan Ford. Parsing expression grammars: a recognition-based syntactic foundation. In *POPL '04: Proceedings of the 31st ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 111–122, New York, NY, USA, 2004. ACM.
- [121] M. Fowler. *Refactoring: Improving the Design of Existing Code*. Addison-Wesley, 1999.
- [122] Martin Fowler et al. *Patterns of Enterprise Application Architecture*. Addison-Wesley, 2003.
- [123] Randolph Franklin. On an improved algorithm for decentralized extrema finding in circular configurations of processors. *Commun. ACM*, 25(5):336–337, 1982.
- [124] Keir Fraser and Tim Harris. Concurrent programming without locks. *ACM Trans. Comput. Syst.*, 25(2):5, 2007.
- [125] Matteo Frigo, Charles E. Leiserson, and Keith H. Randall. The implementation of the Cilk-5 multithreaded language. In *1998 ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 212–223, June 1998.
- [126] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.
- [127] Ronald Garcia, Jaako Järvi, and Andrew Lumsdaine. A comparative study of language support for generic programming. In *ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, pages 115–134, October 2003.
- [128] Vijay K. Garg. *Concurrent and Distributed Computing in Java*. Wiley, 2004.
- [129] David Garlan and Mary Shaw. An introduction to software architecture. Technical Report CMU-CS-94-166, School of Computer Science, Carnegie Mellon University, January 1994.
- [130] F. Genuys, editor. *Programming Languages (NATO Advanced Study Institute)*. Academic Press, 1968.
- [131] C.M. Geschke, J.H. Morris, Jr., and E.H. Satterthwaite. Early experience with Mesa. *Communications of the ACM*, 20(8):540–553, August 1977.
- [132] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The Google file system. In *ACM Symposium on Operating System Principles (SOSP'03)*, pages 29–43, October 2003.
- [133] Sukumar Ghosh. *Distributed Systems: an Algorithm Approach*. Chapman & Hall/CRC, 2007.
- [134] David K. Gifford and Nathan Glasser. Remote pipes and procedures for efficient distributed communication. *ACM Trans. Comput. Syst.*, 6(3):258–283, 1988.
- [135] Robert L. Glass. The Standish Report: does it really describe the software crisis? *Communications of the ACM*, 49(8):15–16, August 2006.
- [136] Brian Goetz, Tim Peierls, Joshua Bloch, Joseph Bowbeer, David Holmes, and Doug Lea. *Java Concurrency in Practice*. Addison-Wesley, 2006.
- [137] A. Goldberg and D. Robson. *Smalltalk-80, The Language and its Implementation*. Addison-Wesley, 1983, Reprinted with corrections 1985.
- [138] James Gosling, Bill Joy, and Guy Steele. *The Java Language Specification*. Addison-Wesley, 1996.

- [139] James Gosling, Bill Joy, Guy Steele, and Gilad Bracha. *The Java Language Specification*. Addison-Wesley, second edition, 2000.
- [140] D. Gries. *The Science of Programming*. Springer, 1981.
- [141] Ralph E. Griswold and Madge T. Griswold. *The Icon Programming Language*. Prentice Hall, second edition, 1990.
- [142] Peter Grogono. Modular concurrency. Keynote Speech for Canadian University Software Engineering Conference (CUSEC) 2006, January 2006.
- [143] Peter Grogono. Complexity in systems and software, April 2007. Invited talk for IEEE Computer Society (Montreal chapter).
- [144] Peter Grogono. Ports, protocols, and processes: a programming paradigm?, November 2007. Invited talk for the Advanced Programming Languages Study Group of the British Computer Society.
- [145] Peter Grogono. Living with concurrency. Keynote Speech at Concordia Undergraduate Software Engineering Conference (CUSEC 2008), January 2008.
- [146] Peter Grogono, Nurudeen Lameed, and Brian Shearing. Modularity + concurrency=manageability. Technical Report TR E-04, Department of Computer Science and Software Engineering, Concordia University, September 2007.
- [147] Peter Grogono, Nurudeen Lameed, and Brian Shearing. Modularity + concurrency=manageability, 2007. <http://users.encs.concordia.ca/~grogono/Erasmus/erasmus.html>.
- [148] Peter Grogono and Brian Shearing. A modular language for concurrent programming. Technical Report TR E-01, Department of Computer Science and Software Engineering, Concordia University, September 2006.
- [149] Peter Grogono and Brian Shearing. A modular language for concurrent programming, 2006. <http://users.encs.concordia.ca/~grogono/Erasmus/erasmus.html>.
- [150] Peter Grogono and Brian Shearing. Modular concurrency: a new approach to manageable software. Technical Report TR E-02, Department of Computer Science and Software Engineering, Concordia University, April 2007.
- [151] Peter Grogono and Brian Shearing. Modular concurrency: a new approach to manageable software, 2007. <http://users.encs.concordia.ca/~grogono/Erasmus/erasmus.html>.
- [152] Peter Grogono and Brian Shearing. A note on communication. Technical Report TR E-05, Department of Computer Science and Software Engineering, Concordia University, August 2007.
- [153] Peter Grogono and Brian Shearing. A note on communication, 2007. <http://users.encs.concordia.ca/~grogono/Erasmus/erasmus.html>.
- [154] Peter Grogono and Brian Shearing. Towards concurrent software engineering: Preparing for paradigm shift. Technical Report TR E-03, Department of Computer Science and Software Engineering, Concordia University, April 2007.
- [155] Peter Grogono and Brian Shearing. Towards concurrent software engineering: Preparing for paradigm shift, 2007. <http://users.encs.concordia.ca/~grogono/Erasmus/erasmus.html>.
- [156] Peter Grogono and Brian Shearing. Concurrent software engineering: Preparing for paradigm shift. In Bipin C. Desai, editor, *Proceedings of the Canadian Conference on Computer Science & Software Engineering (C³S²E'08)*, pages 99–108. ACM Press, May 2008.

- [157] Peter Grogono and Brian Shearing. MEC Reference Manual. Technical Report TR E-06, Department of Computer Science and Software Engineering, Concordia University, February 2008.
- [158] Peter Grogono and Brian Shearing. MEC Reference Manual, 2008. <http://users.encs.concordia.ca/~grogono/Erasmus/erasmus.html>.
- [159] Peter Grogono and Brian Shearing. MEC Tests. Technical Report TR E-07, Department of Computer Science and Software Engineering, Concordia University, February 2008.
- [160] Peter Grogono and Brian Shearing. MEC Tests, 2008. <http://users.encs.concordia.ca/~grogono/Erasmus/erasmus.html>.
- [161] Peter Grogono and Brian Shearing. Modular concurrency: a new approach to manageable software. In *3rd International Conference on Software and Data Technologies (ICSOFT 2008)*, pages 47–54, July 2008.
- [162] Peter Grogono and Brian Shearing. Protocol satisfaction. Technical Report TR E-08, Department of Computer Science and Software Engineering, Concordia University, February 2008.
- [163] Peter Grogono and Brian Shearing. Protocol satisfaction, 2008. <http://users.encs.concordia.ca/~grogono/Erasmus/erasmus.html>.
- [164] Dan Grossman. The transactional memory/garbage collection analogy. In *OOPSLA '07: Proceedings of the 22nd annual ACM SIGPLAN conference on Object oriented programming systems and applications*, pages 695–706, New York, NY, USA, 2007. ACM.
- [165] Raphael Güntensperger and Jürg Gutknecht. Active C#. In *2nd International Workshop .NET Technologies'2004*, pages 47–59, May 2004.
- [166] David R. Hanson and Ralph E. Griswold. The SL5 procedure mechanism. *Communications of the ACM*, 21(5):392–400, May 1978.
- [167] S. Harbison. *Modula-3*. Prentice Hall, 1992.
- [168] Tim Harris and Keir Fraser. Language support for lightweight transactions. In *OOPSLA '03: Proceedings of the 18th Annual ACM SIGPLAN Conference on Object oriented programming systems and applications*, pages 388–402, October 2003.
- [169] A.C. Hartmann. *A Concurrent Pascal Compiler for Minicomputers*. Number 50 in Lecture Notes in Computer Science. Springer, 1977.
- [170] Brian Hayes. The post-OOP paradigm. *The American Scientist*, 91(2):106–110, March–April 2003.
- [171] George T. Heineman and William T. Councill. *Component-Based Software Engineering*. Addison-Wesley, 2001.
- [172] Christian Heinlein. Null values in programming languages. In H. R. Arabnia, editor, *Proc. Int. Conf. on Programming Languages and Compilers (PLC'05)*, pages 123–129, June 2005.
- [173] Christian Heinlein. Open types and bidirectional relationships as an alternative to classes and inheritance. *Journal of Object Technology (www.jot.fm)*, 6(3):101–151, March–April 2007.

- [174] Thomas A. Henzinger, Ranjit Jhala, and Rupak Majumdar. Permissive interfaces. In *ESEC/FSE-13: Proceedings of the 10th European Software Engineering Conference (held jointly with 13th ACM SIGSOFT International Symposium on Foundations of Software Engineering)*, pages 31–40, 2005.
- [175] Maurice Herlihy and Nir Shavit. *The Art of Multiprocessor Programming*. Morgan Kaufmann, 2008.
- [176] Carl Hewitt and Henry Baker Jr. Actors and continuous functionals. Technical Report MIT/LCS/TR-194, MIT, December 1977. MIT-LCS-TR-194.pdf.
- [177] Carl Hewitt, Peter Bishop, Richard Steiger, Irene Greif, Brian Smith, Todd Matson, and Roger Hale. Behavioral semantics of nonrecursive control structures. In *Programming Symposium, Proceedings Colloque sur la Programmation*, pages 385–407, London, UK, 1974. Springer-Verlag.
- [178] D. S. Hirschberg and J. B. Sinclair. Decentralized extrema-finding in circular configurations of processors. *Communications of the ACM*, 23(11):627–628, 1980.
- [179] C. A. R. Hoare. Hints on programming language design. In C. Bunyan, editor, *Computer Systems Reliability*, volume 20 of *State of the Art Report*, pages 505–534. Pergamon, 1974. Reprinted in [184].
- [180] C. A. R. Hoare. Monitors: an operating system structuring concept. *Communications of the ACM*, 17(10):549–557, October 1974.
- [181] C. A. R. Hoare. Communicating sequential processes. *Communications of the ACM*, 21(8):666–677, August 1978.
- [182] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.
- [183] C. A. R. Hoare. Letter to Per Brinch Hansen, 1991. Reprinted in [52].
- [184] C. A. R. Hoare and C.B. Jones. *Essays in Computing Science*. Prentice Hall, 1989.
- [185] C. A. R. Hoare and R. H. Perrott. *Operating Systems Techniques*. Academic Press, 1972.
- [186] Lorin Hochstein and Victor R. Basili. The ASC-Alliance project: a case study of large-scale parallel scientific code development. *IEEE Computer*, 41(3):50–58, March 2008.
- [187] Daniel H. Hoffman and David M. Weiss, editors. *Software Fundamentals: Collected Papers by David L. Parnas*. Addison-Wesley, 2001.
- [188] Neville Holmes. The problem with Unicode. *IEEE Computer*, 36(6):114–6, June 2003.
- [189] R. C. Holt. A short introduction to Concurrent Euclid. *ACM SIGPLAN Notices*, 17(5):60–79, 1982.
- [190] Richard C. Holt. Some deadlock properties of computer systems. *ACM Computing Surveys*, 4(3):179–196, 1972.
- [191] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.
- [192] Andrew Hunt and David Thomas. *The Pragmatic Programmer: From Journeyman to Master*. Addison-Wesley, 1999.
- [193] Galen C. Hunt and James R. Larus. Singularity: rethinking the software stack. *SIGOPS Operating System Review*, 41(2):37–49, 2007.
- [194] Costin Iancu and Erich Strohmaier. Optimizing communication overlap for high-speed networks. In *PPoPP’07*, pages 35–45, March 2007.

- [195] Daniel Jackson. *Software Abstractions: Logic, Language, and Analysis*. MIT Press, 2006.
- [196] M.A. Jackson. *Principles of Program Design*. Academic Press, 1975.
- [197] M.A. Jackson. Information systems: Modelling, sequencing and transformation. In *Proceedings of the 3rd International Conference on Software Engineering (ICSE 1978)*, pages 72–81, 1978.
- [198] M.A. Jackson. Information systems: Modelling, sequencing and transformation. In R.M. McKeag and A.M. MacNaughten, editors, *On the Construction of Programs*. Cambridge University Press, 1980.
- [199] Christian L. Jacobsen and Matthew C. Jadud. Towards concrete concurrency: occam- π on the LEGO mindstorms. In *SIGCSE '05: Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education*, pages 431–435, New York, NY, USA, 2005. ACM Press.
- [200] Nima Jafroodi. A type system for the erasmus language. Master’s thesis, Department of Computer Science and Software Engineering, Concordia University, January 2008.
- [201] Einar Broch Johnsen and Olaf Owe. An asynchronous communication model for distributed concurrent objects. *Software and Systems Modeling*, 6:39–58, 2007.
- [202] Einar Broch Johnsen, Olaf Owe, and Ingrid Chich Yu. Creol: a type-safe object-oriented model for distributed concurrent systems. *Theoretical Computer Science*, 365:23–66, 2006.
- [203] A.C. Kay. The early history of Smalltalk. In T.J. Bergin Jr. and R.G. Gibson Jr., editors, *History of Programming Languages–II*. ACM Press, 1996.
- [204] Alan C. Kay. The early history of Smalltalk. *ACM SIGPLAN Notices*, 28(3):69–95, 1993.
- [205] J.O. Kephart and D.M. Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, January 2003.
- [206] B.W. Kernighan and R. Pike. *The UNIX Programming Environment*. Prentice Hall, 1984.
- [207] Willard Khorfage and Arthur P. Goldberg. Hermes language experiences. *Software—Practice and Experience*, 25(4):389–402, April 1995.
- [208] R. Koster, A.P. Black, J. Huang, J. Walpole, and C. Pu. Thread transparency in information flow middleware. *Software—Practice and Experience*, 33:351–373, 2003.
- [209] G. Krasner. *Smalltalk–80, Bits of History, Words of Advice*. Addison-Wesley, 1983.
- [210] Bent Bruun Kristensen, Ole Lehrmann Madsen, and Birger Mö ller-Pedersen. The when, why and why not of the BETA programming language. In *HOPL III: Proceedings of the Third ACM SIGPLAN Conference on History of Programming Languages*, pages 10–1–10–57, New York, NY, USA, 2007. ACM Press.
- [211] Philippe Kruchten. Casting software design in the function-behavior-structure framework. *IEEE Software*, 22(2):52–58, March/April 2005.
- [212] Béchir Ktari, Hamido Fujita, Mohamed Mejri, and Daniel Godbout. Towards a new software development environemnt. *Knowledge-Based Systems*, 20:683–693, 2007.

- [213] Martin Kuhlemann, Marko Rosenmüller, Sven Apel, and Thomas Leich. On the duality of aspect-oriented and feature-oriented design patterns. In *ACP4IS '07: Proceedings of the 6th workshop on Aspects, components, and patterns for infrastructure software*, page 5, New York, NY, USA, 2007. ACM Press.
- [214] Thomas Kuhn. *The Structure of Scientific Revolutions*. University of Chicago Press, 1962. Third Edition published in 1996.
- [215] Thomas Kühne and Daniel Schreiber. Can programming be liberated from the two-level style: multi-level programming with deepJava. In *OOPSLA '07: Proceedings of the 22nd annual ACM SIGPLAN conference on Object oriented programming systems and applications*, pages 229–244, New York, NY, USA, 2007. ACM.
- [216] John Lakos. *Large-Scale C++ Software Design*. Professional Computing Series. Addison-Wesley, 1996.
- [217] Nurudeen Lameed. Implementing concurrency in a process-based language. Master’s thesis, Department of Computer Science and Software Engineering, Concordia University, March 2008.
- [218] Nurudeen Lameed and Peter Grogono. Separating program semantics from deployment. In *3rd International Conference on Software and Data Technologies (ICSOFT 2008)*, pages 63–70, July 2008.
- [219] Leslie Lamport. Concurrent reading and writing. *Commun. ACM*, 20(11):806–811, 1977.
- [220] B. W. Lampson. *Specifying Systems : The TLA+ Language and Tools for Hardware and Software Engineers*. Addison-Wesley, 2002.
- [221] B. W. Lampson, J. G. Mitchell, and E. H. Satterthwaite. On the transfer of control between contexts. In *Programming Symposium, LNCS 19*, pages 181–203. Springer, April 1974.
- [222] Butler W. Lampson and David D. Redell. Experience with processes and monitors in Mesa. *Communications of the ACM*, 23(2):105–117, February 1980.
- [223] James Larus. Spending moore’s dividend. Technical Report MSR–TR–2008–69, Microsoft Research, February 2008.
- [224] Hugh C. Lauer and Roger M. Needham. On the duality of operating system structures. *ACM SIGOPS Operating Systems Review*, 13(2):3–19, April 1979. Originally published in Proc. Second International Symposium on Operating Systems, IRIA, October 1978.
- [225] Robert B. Laughlin. *A Different Universe: Reinventing Physics from the Bottom Down*. Basic Books, 2005.
- [226] Gérard Le Lann. Distributed systems — towards a formal approach. In *IFIP Congress*, pages 155–160, 1977.
- [227] Doug Lea. *Concurrent Programming in Java: Design Principles and Patterns*. Prentice Hall, 1999.
- [228] Doug Lea, Pete Soper, and Miles Sabin. The Java Isolation API: Introduction, applications and inspiration. bitser.net/isolate-interest/slides.pdf. Accessed 2007/06/14, 2004.
- [229] Gary T. Leavens and Todd D. Millstein. Multiple dispatch as dispatch on tuples. In *ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, pages 374–387, 1998.

- [230] Edward Lee et al. Overview of the Ptolemy project. Technical Report UCB/ERL M03/25, Department of Electrical Engineering and Computer Science, University of California at Berkeley, July 2003.
- [231] Edward A. Lee. Overview of the Ptolemy project. Technical Report Technical Memorandum No. UCB/ERL M03/25, University of California, Berkeley, July 2003. ptolemy.eecs.berkeley.edu/publications/papers/03/overview.
- [232] Edward A. Lee. Building unreliable systems out of reliable components: The real time story. Technical Report UCB/EECS-2005-5, EECS Department, University of California, Berkeley, October 7 2005.
- [233] Edward A. Lee. The problem with threads. *IEEE Computer*, 39(5):33–42, May 2006.
- [234] Daniel Lehmann and Michael O. Rabin. On the advantages of free choice: a symmetric and fully distributed solution to the dining philosophers problem. In *POPL '81: Proceedings of the 8th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 133–138, New York, NY, USA, 1981. ACM.
- [235] Andreas Leitner, Patrick Eugster, Manuel Oriol, and Ilinca Ciupa. Reflecting on an existing language. *Journal of Object Technology (www.jot.fm)*, 6(9):319–339, October 2007. Special Issue: TOOLS Europe 2007.
- [236] Calvin Lin and Lawrence Snyder. *Principles of Parallel Programming*. Addison-Wesley, 2009.
- [237] Tim Lindholm and Frank Yellin. *The Java Virtual Machine*. Addison-Wesley, 1997.
- [238] Barbara Liskov. Primitives for distributed computing. In *SOSP '79: Proceedings of the Seventh ACM Symposium on Operating Systems Principles*, pages 33–42, New York, NY, USA, 1979. ACM.
- [239] Barbara H. Liskov and Jeannette M. Wing. A behavioral notion of subtyping. *ACM Transactions on Programming Languages and Systems*, 16(6):1811–1841, 1994.
- [240] Hans Loeper, Amro Khatib, and Peter Neubert. Concurrent objects in Ada 95. *Ada Lett.*, XVII(6):47–64, 1997.
- [241] Edward Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20:130–141, March 1963.
- [242] Panagiotis Louridas, Diomidis Spinellis, and Vasileios Vlachos. Power laws in software. *ACM Transactions on Software Engineering and Methodology*, 18(1):1–26, 2008.
- [243] Jeff Magee and Jeff Kramer. *Concurrency; State Models and Java Programs*. Wiley, 1999.
- [244] Jeff Magee and Jeff Kramer. *Concurrency; State Models and Java Programs*. Wiley, second edition, 2006.
- [245] Jeremy Manson. *The Java Memory Model*. PhD thesis, University of Maryland, 2004.
- [246] Michael J. Manthey and Bernard M. E. Moret. The computational metaphor and quantum physics. *Communications of the ACM*, 26(2):137–145, February 1983.
- [247] S. Matsuoka and A. Yonezawa. Analysis of inheritance anomaly in object-oriented concurrent programming language. In *Research Directions in Concurrent Object-Oriented Programming*, pages 107–150. MIT Press, 1993.

- [248] D. May. Occam. *ACM SIGPLAN Notices*, 18(4):69–79, April 1983.
- [249] M. D. McIlroy. Mass produced software components. In *NATO Conference on Software Engineering, NATO Science Committee, Garmisch, Germany*, pages 88–98. Petrocelli-Charter, 1968.
- [250] M. Douglas McIlroy. Power series, power serious. *J. Functional Programming*, 9(3):325–337, 1999.
- [251] Linda McIver and Damian Conway. Seven deadly sins of introductory programming language design. In *Proceedings of the 1996 International Conference on Software Engineering: Education and Practice (SE:EP '96)*, page 309, Washington, DC, USA, 1996. IEEE Computer Society.
- [252] Erik Meijer. Confessions of a used programming language salesman: Getting the masses hooked on Haskell. Unpublished, 2006. research.microsoft.com/emeijer/Papers/ICFP06.pdf.
- [253] Bertrand Meyer. *Object-Oriented Software Construction*. Prentice Hall, second edition, 1997.
- [254] Giuseppe Milicia and Vladimiro Sassone. The inheritance anomaly: ten years after. In *SAC '04: Proceedings of the 2004 ACM Symposium on Applied Computing*, pages 1267–1274, New York, NY, USA, 2004. ACM.
- [255] Robin Milner. *A Calculus of Communicating Systems*. Springer, 1980.
- [256] Robin Milner. *Communication and Concurrency*. Prentice Hall International, 1989.
- [257] Robin Milner. *Communicating and Mobile Systems: The π Calculus*. Cambridge University Press, 1999.
- [258] J. Misra. Axioms for memory access in asynchronous hardware systems. *ACM Transactions on Programming Languages and Systems*, 8(1):142–153, 1986.
- [259] Melanie Mitchell. Complex systems: Network thinking. *Artificial Intelligence*, 170(18):1194–1212, 2006.
- [260] David Monniaux. The pitfalls of verifying floating-point computations. *ACM Transactions on Programming Languages and Systems*, 30(3):1–41, 2008.
- [261] Hanspeter Mössenböck and Niklaus Wirth. Component Pascal Language Report. Technical report, Oberon Microsystems, Inc., March 2001. www.oberon.ch/pdf/CP-Lang.pdf.
- [262] Ana Lúcia De Moura and Roberto Ierusalimsky. Revisiting coroutines. *ACM Trans. Program. Lang. Syst.*, 31(2):1–31, 2009.
- [263] Matthias Neubauer and Peter Thiemann. From sequential programs to multi-tier applications by program transformation. In *ACM Symposium on Principles of Programming Languages (POPL'05)*, pages 221–232, January 2005.
- [264] K. Nygaard and O-J. Dahl. The development of the SIMULA language. In R. Wexelblat, editor, *History of Programming Languages*, pages 439–493. Academic Press, 1981.
- [265] Object Management Group. OMG Unified Modeling Language (OMG UML), Superstructure, V2.1.2, November 2007. <http://www.omg.org/spec/UML/2.1.2/-Superstructure/PDF/>. Accessed 2008/03/15.
- [266] Kunle Olukotun and Lance Hammond. The future of microprocessors. *ACM Queue*, 3(7):26–29, 2005.

- [267] Harold Ossher and Peri Tarr. Using multidimensional separation of concerns to (re)shape evolving software. *Commun. ACM*, 44(10):43–50, 2001.
- [268] D.L. Parnas. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058, December 1972. Reprinted in [187, pp. 145-155].
- [269] David A. Patterson. Latency lags bandwidth. *Commun. ACM*, 47(10):71–75, 2004.
- [270] Arnold Pears, Stephen Seidman, Lauri Malmi, Linda Mannila, Elizabeth Adams, Jens Bennesen, Marie Devlin, and James Paterson. A survey of literature on the teaching of introductory programming. *SIGCSE Bull.*, 39(4):204–223, 2007.
- [271] Alan J. Perlis. Epigrams on programming. *ACM SIGPLAN Notices*, 17(9):7–13, September 1982.
- [272] Frédéric Peschanski and Samuel Hym. A stackless runtime environment for a Pi-calculus. In *VEE '06: Proceedings of the 2nd international conference on Virtual execution environments*, pages 57–67, New York, NY, USA, 2006. ACM Press.
- [273] Gary L. Peterson. Concurrent reading while writing. *ACM Transactions on Programming Languages and Systems*, 5(1):46–55, 1983.
- [274] Robert G. Pettit IV and Hassan Gomaa. Modeling behavioral design patterns of concurrent objects. In *ICSE '06: Proceeding of the 28th international conference on Software engineering*, pages 202–211, New York, NY, USA, 2006. ACM Press.
- [275] Benjamin C. Pierce. *Types and Programming Languages*. MIT Press, 2002.
- [276] T.M. Pigosky. *Practical Software Maintenance*. Wiley, 1996.
- [277] Alex Potanin, James Noble, Marcus Frean, and Robert Biddle. Scale-free geometry in OO programs. *Communications of the ACM*, 48(5):99–103, May 2005.
- [278] Viera K. Proulx and Tanya Cashorali. Calculator problem and the design recipe. *SIGPLAN Notices*, 40(3):4–11, 2005.
- [279] Bill Pugh. The Java Memory Model, 2004. www.cs.umd.edu/~pugh/java/memoryModel.
- [280] Bill Pugh et al. JSR 133: The Java Memory Model, 2004. jcp.org/en/jsr/detail?id=133.
- [281] William Pugh. The Java Memory Model is fatally flawed. *Concurrency: Practice and Experience*, 12(6):445–455, 2000.
- [282] Owen Rees. Using path expressions as concurrency guards. Technical Report APM.1010.00.02, ANSA, 1993.
- [283] John Reynolds. A short course in separation logic. www.cs.cmu.edu/afs/cs.cmu.edu/project/fox-19/member/jcr/wwwaac2003/aac.html, 2003.
- [284] J. P. Rosen. What orientation should Ada objects take? *Commun. ACM*, 35(11):71–76, 1992.
- [285] Barbara G. Ryder and Mary Lou Soffa. The impact of software engineering research on modern programming language design. *ACM Transactions on Software Engineering and Methodology*, 14(4):431–477, October 2005.
- [286] Bo I. Sandén. Concurrent design patterns for resource sharing. In *TRI-Ada '97: Proceedings of the conference on TRI-Ada '97*, pages 173–183, New York, NY, USA, 1997. ACM Press.
- [287] Davide Sangiorgi and David Walker. *The π -calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.

- [288] Vijay Saraswat et al. Report on the experimental language x10, February 2006. Draft v 0.41.
- [289] Arno Schödl. The NewtonScript programming language. <http://web.archive.org/web/20041010175535/www.cc.gatech.edu/~schoedl/projects/NewtonScript/>, 2004. Georgia Institute of Technology.
- [290] Bran Selic. What's new in UML 2.0? IBM White Paper, April 2005. Available at <ftp://ftp.software.ibm.com/software/rational/web/whitepapers/intro2uml2.pdf>. Accessed 2008/03/25.
- [291] Bran Selic. Model-driven development: Its essence and opportunities. In *Ninth IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC'06)*, pages 313–319, Los Alamitos, CA, USA, April 2006. IEEE Computer Society.
- [292] Bran Selic. Tutorial: an overview of UML 2. In *ICSE '06: Proceeding of the 28th International Conference on Software Engineering*, pages 1069–1070, New York, NY, USA, 2006. ACM Press.
- [293] Bran Selic and Jim Rumbaugh. Using UML for modeling complex real-time systems. Rational Software Corporation, March 1998.
- [294] Jonathan S. Shapiro, Jonathan M. Smith, and David J. Farber. EROS: a fast capability system. *SIGOPS Operating System Review*, 34(2):21–22, 2000.
- [295] Mary Shaw and David Garlan. *Software Architecture: Perspectives on an Emerging Discipline*. Prentice Hall, 1996.
- [296] Brian Shearing. *Malleable Software*, chapter 1, pages 1–19. TBA Press, 2005.
- [297] Herbert A. Simon. *The Sciences of the Artificial*. MIT Press, second edition, 1981.
- [298] Anthony Simons. The theory of classification. Series of articles in *Journal of Object Technology* (www.jot.fm), 2003–2005.
- [299] Anthony Simons. The theory of classification: Part 18: Polymorphism through the looking glass. *Journal of Object Technology* (www.jot.fm), 4(4):7–18, May-June 2005. www.jot.fm/issues/issue_2005_05/column1.
- [300] David B. Skillicorn and Domenico Talia. Models and languages for parallel computation. *ACM Computing Surveys*, 30(2):123–169, June 1998.
- [301] Neil Smyth. Communicating sequential processes in Ptolemy II. Master's thesis, Electronics Research Laboratory, Berkeley, 1998. Technical Memorandum UCB/ERL M98/70.
- [302] Lawrence Snyder. Type architectures, shared memory, and the corollary of modest potential. *Annual Review of Computer Science*, 1:289–317, June 1986.
- [303] Pete Soper. JSR 121: Application Isolation API Specification. Java Specification Requests, 2002. <http://jcp.org/aboutJava/communityprocess/final/jsr121/index.html>. Accessed 2008/03/15.
- [304] Michael Sparks. Kamaelia: highly concurrent and network systems tamed. R&D White Paper WHP 113, Research & Development: British Broadcasting Corporation, June 2005.
- [305] Robert Strom. *HERMES: A Language for Distributed Computing*. Prentice Hall, 1991.
- [306] Bjarne Stroustrup. *The Design and Evolution of C++*. Addison-Wesley, 1994.
- [307] Bjarne Stroustrup. *The C++ Programming Language*. Addison-Wesley, 1997.

- [308] Sun. JavaDoc. java.sun.com/j2se/javadoc, 2006.
- [309] Herb Sutter. Software and the concurrency revolution. Slides for a talk given at Xerox PARC, March 2006.
- [310] Herb Sutter and James Larus. Software and the concurrency revolution. *ACM Queue*, 3(7):54–62, September 2005.
- [311] Norihisa Suzuki. Analysis of pointer rotation. In *POPL '80: Proceedings of the 7th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 1–11, New York, NY, USA, 1980. ACM Press.
- [312] Hans Svensson and Lars-Åke Fredlund. A more accurate semantics for distributed Erlang. In *Erlang '07: Proceedings of the 2007 SIGPLAN workshop on Erlang Workshop*, pages 43–54, New York, NY, USA, 2007. ACM.
- [313] Hans Svensson and Lars-Åke Fredlund. Programming distributed Erlang applications: pitfalls and recipes. In *Erlang '07: Proceedings of the 2007 SIGPLAN workshop on Erlang Workshop*, pages 37–42, New York, NY, USA, 2007. ACM.
- [314] Clemens Szyperski. *Component Software*. Addison-Wesley, 2002.
- [315] S. Tucker Taft and Robert A. Duff, editors. *Ada 95 Reference Manual: Language and Standard Libraries*. Number 1246 in LNCS. Springer, 1995. International Standard ISO/IEC 8652:1995(E).
- [316] Chandramohan A. Thekkath, Henry M. Levy, and Edward D. Lazowska. Separating data and control transfer in distributed operating systems. *SIGOPS Oper. Syst. Rev.*, 28(5):2–11, 1994.
- [317] W. Thiele. Die entwicklung des PL/1-übersetzers für das platten/-bandbetriebssystem [the development of the PL/1 translator for the disk/tape operating system]. *Elektronische Rechenanlagen [Electronic Computers]*, 11(1):25–35, 1969. (in German).
- [318] Harold Thimbleby. Explaining code for publication. *Software—Practice and Experience*, 33:975–1001, 2003.
- [319] Dave Thomas. The impedance imperative: Tuples + objects + infosets = too much stuff! *Journal of Object Technology (www.jot.fm)*, 2(5):1–5, September-October 2003. www.jot.fm/issues/issue_2003_09/column1.
- [320] Dave Thomas. UML—unified or universal modeling language? *Journal of Object Technology (www.jot.fm)*, 2(1):1–12, January-February 2003. http://www.jot.fm/issues/issue_2003_01/column1.
- [321] Dave Thomas. The legacy and liability of object technology — the dark side of OO. *Journal of Object Technology (www.jot.fm)*, 7(6):27–30, July-August 2008. www.jot.fm/issues/issue_2008_07/column3.
- [322] David Ungar and Randall B. Smith. Self: The power of simplicity. In *OOPSLA '87: Conference proceedings on Object-oriented programming systems, languages and applications*, pages 227–242. ACM Press, 1987.
- [323] David Ungar and Randall B. Smith. Self. In *HOPL III: Proceedings of the third ACM SIGPLAN conference on History of programming languages*, pages 9–1–9–50, New York, NY, USA, 2007. ACM Press.
- [324] Sergi Valverde and Ricard V. Solé. Hierarchical small worlds in software architecture. Santa Fe Institute Working Paper 03-07-044, submitted to IEEE Transactions on Software Engineering, 2003.
- [325] Dimitri van Heesch. Doxygen. www.stack.nl/~dimitri/doxygen, 2006.

- [326] R. van Ommering, F. van der Linden, J. Kramer, and J. Magee. The Koala component model for consumer electronics software. *IEEE Computer*, 33(3):78–85, March 2000.
- [327] P. van Roy, P. Brand, D. Duchier, S. Haridi, M. Henz, and C. Schulte. Logic programming in the context of multiparadigm programming: the Oz experience. to appear, 2005.
- [328] Peter van Roy and Seif Haridi. *Concepts, Techniques, and Models of Computer Programming*. MIT Press, 2001.
- [329] Jan Vitek and Boris Bokowski. Confined types in Java. *Software—Practice and Experience*, 31(6):507–532, May 2000.
- [330] Alessandro Warth and Alan Kay. Worlds: controlling the scope of side effects. Technical Report Research Note RN–2008–003, VPRI Research Institute, September 2008.
- [331] J. Welsh and A. Lister. A comparative study of task communication in Ada. *Software—Practice and Experience*, 11:257–290, 1981.
- [332] Ben Wiedermann and William R. Cook. Extracting queries by static analysis of transparent persistence. In *POPL '07: Proceedings of the 34th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 199–210, New York, NY, USA, 2007. ACM.
- [333] M. V. Wilkes and R. M. Needham. *The Cambridge CAP Computer and its Operating System*. The Computer Science Library. North Holland, 1979.
- [334] Barry Wilkinson and Michael Allen. *Parallel Programming: Techniques and Applications for Using Networked Workstations and Parallel Computers*. Pearson/Prentice Hall, second edition, 2005.
- [335] Niklaus Wirth. Design and implementation of Modula. *Software—Practice and Experience*, 7(1):67–84, 1977.
- [336] Niklaus Wirth. *Programming in Modula-2*. Springer, 1982.
- [337] Niklaus Wirth. The programming language Oberon. *Software—Practice and Experience*, 18(7):671–690, 1988.