Department of Computer Science
University of Manchester

Manchester M13 9PL, England

Technical Report Series

UMCS–92–12–2

C. B. Jones and A. M. McCauley

Formal Methods – Selected Historical References

Formal Methods – Selected Historical References

C. B. Jones and A. M. McCauley*

Department of Computer Science
University of Manchester
Oxford Rd., Manchester, U.K.
cbj@cs.man.ac.uk

1992-12-09

Abstract

This report contains citations to papers which are of historical interest in the area of formal approaches to software development.

Technical Reports issued by the Department of Computer Science, Manchester University, are available by anonymous ftp from `ml.cs.man.ac.uk` (130.88.13.4) in the directory `/pub/TR`. The files are stored as PostScript, in compressed form, with the report number as filename. Alternatively, reports are available by post from The Computer Library, Department of Computer Science, The University, Oxford Road, Manchester M13 9PL, U.K.

Introduction

This technical report lists references located during research undertaken into the history of the field of research which has become known as 'formal methods'. This historical work has so far resulted in a paper of which a technical report version is available ('The Search for Tractable Ways of Reasoning about Programs', UMCS-92-4-4). The accumulated references are also being used as reference material in preparation – with Fred Schneider – for 'The Quest For Program Correctness: Selected Readings' and a paper on the history of 'programming language semantics' which is being written with Joe Stoy. Since a substantial number of references have been discovered it seems worthwhile to make the list available to others. The references are being collated and checked by AM; both AM and CBJ would be grateful for corrections. The current list[1] has been checked for accuracy as far is possible with the sources available. It should be noted that this is not a complete list of all Formal Methods references and the authors would therefore be grateful for both corrections and suggestions for further items.

It should also be noted that there is often more than one citation for the same text. This is particularly true where a technical report has later been published as a paper in a major journal. Pre-prints are not listed unless they are of special significance because – for example – they were influential before final publication. It seems reasonable to focus on papers more than ten years old.

The authors regret that they cannot undertake to provide copies of the cited material.

References

[1] J.-R. Abrial and S. A. Schuman. Non-deterministic system specification. In *[305]*, pages 34–50, 1979.

[2] J.-R. Abrial, S. A. Schuman, and B. Meyer. Specification language. In R. M. McKeag and A. M. Macnaghten, editors, *On the Construction of Programs*, pages 343–410. Cambridge, 1980.

[3] P. Aczel. A note on program verification. Manuscript, Manchester, January 1982.

[4] P. Aczel. *Non-Well-Founded Sets*. CSLI Lecture Notes: 14. Center for the Study of Language and Information, Stanford, 1988.

[5] C. D. Allen, D. N. Chapman, and C. B. Jones. A formal definition of ALGOL 60. Technical Report 12.105, IBM Laboratory Hursley, August 1972.

[6] J. Alton, H. Weiskittel, and J. Latham-Jackson. Catalogue of the papers of Christopher Strachey (1916–75). Technical Report CSAC 71/1/80, Contemporary Scientific Archives, Deposited in the Bodleian Library, Oxford, 1980.

[7] K. R. Apt. Ten years of Hoare's logic: A survey – part I. *ACM Transactions on Programming Languages and Systems*, 3:431–483, 1981.

[8] K. R. Apt. Ten years of Hoare's logic: A survey – part II: Nondeterminism. *Theoretical Computer Science*, 28:83–109, 1984.

[9] K. R. Apt, N. Francez, and W.-P. de Roever. A proof system for communicating sequential processes. Technical Report RUU-CS-80-4, University of Utrecht, The Netherlands, May 1980. Also published as [10].

[1]Compiling this list has resulted in the production of a second list of other material, particularly unpublished manuscripts. At the time of publication the authors are still trying to develop precise citations for them.

[10] K. R. Apt, N. Francez, and W.-P. de Roever. A proof system for communicating sequential processes. *ACM Transactions on Programming Langauges and Systems*, 2:359–385, 1980.

[11] J. Arsac. *Foundations of Programming*, volume 23 of *APIC Studies in Data Processing*. Academic Press, 1985. Translated by F. Duncan.

[12] E. A. Ashcroft. Proving assertions about parallel programs. *Journal of Computer and System Sciences*, 10:110–135, 1975.

[13] E. A. Ashcroft, M. Clint, and C. A. R. Hoare. Remarks on "Program proving: Jumps and functions". *Acta Informatica*, 6:317–318, 1976.

[14] E. A. Ashcroft and Z. Manna. Formalization of properties of parallel programs. Technical Report AIM–110, Stanford Artificial Intelligence Project, February 1970. Published as [15].

[15] E. A. Ashcroft and Z. Manna. Formalization of properties of parallel programs. In B. Meltzer and D. Michie, editors, *Machine Intelligence, 6*, pages 17–41. Edinburgh University Press, 1971.

[16] E. A. Ashcroft and W. W. Wadge. R for semantics. Technical Report CS-79-37, Faculty of Mathematics, University of Waterloo, Canada, December 1979.

[17] A. Avron. Foundations and proof theory of 3-valued logics. Technical Report ECS-LFCS-88-48, Department of Computer Science, University of Edinburgh, April 1988.

[18] A. Avron. Natural 3-valued logics – characterization and proof theory. *Journal of Symbolic Logic*, 56(1):276–294, March 1991.

[19] R. J. R. Back. Correctness of explicitly specified procedures. Technical Report IW 154/80, Mathematisch Centrum, Amsterdam, Decmeber 1980. Preprint.

[20] R. J. R. Back. Correctness preserving program refinements: Proof theory and applications. Technical report, Mathematisch Centrum Tract, 131, 1980.

[21] R. J. R. Back. On correct refinement of programs. *Journal of Computer and System Sciences*, 23:49–68, August 1981.

[22] R. J. R. Back. Proving total correctness of nondeterministic programs in infinitary logic. *Acta Informatica*, 15:233–249, 1981.

[23] R. C. Backhouse. *Program Construction and Verification*. Prentice-Hall International, 1986.

[24] R. C. Backhouse, P. Chisholm, G. Malcolm, and E. Saaman. Do-it-Yourself type theory. *Formal Aspects of Computing*, 1:19–84, 1989.

[25] J. Backus. Can programming be liberated from the von Neuman style?: a functional style and its algebra of programs. *Communications of the ACM*, 21:613–641, 1978.

[26] J. Backus, J. H. Williams, and E. L. Wimmers. FL language manual (preliminary version). Technical Report RJ 5339, IBM Almaden Research Center, San Jose, 1986.

[27] J. W. Backus, F. L. Bauer, J. Green, C. Katz, J. McCarthy, P. Naur, A. J. Perlis, H. Rutishauser, K. Samelson, B. Vauquois, J. H. Wegstein, A. van Wijngaarden, and M. Woodger. Revised report on the algorithmic language Algol 60. *Communications of the ACM*, 6(1):1–17, 1963.

[28] J. C. M. Baeten and W. P. Weijland. *Process Algebra*. Cambridge University Press, 1990.

[29] K. Bandat. Tentative steps towards a formal definition of semantics of PL/I. Technical Report TR 25.056, IBM Laboratory, Vienna, July 1963.

[30] K. Bandat. Heinz Zemanek and the IBM laboratory. In *[466]*, pages 53–60, 1985.

[31] H. P. Barendregt. *The Lambda Calculus – Its Syntax and Semantics*. North Holland, 1981.

[32] H. Barringer, J. H. Cheng, and C. B. Jones. A logic covering undefinedness in program proofs. *Acta Informatica*, 21:251–269, 1984.

[33] H. Barringer, R. Kuiper, and A. Pnueli. Now you can compose temporal logic specification. In *Proceedings of 16th ACM STOC*, pages 51–63, Washington, April–May 1984.

[34] D. W. Barron, J. N. Buxton, D. F. Hartley, and C. Strachey. The main features of CPL. *Computer Journal*, 6:134–143, 1963.

[35] D. W. Barron and C. Strachey. Programming. In L. Fox, editor, *Advances in Programming and Non-numerical Computation*, pages 49–82. Pergamon Press, 1966.

[36] J. L. Bates and R. L. Constable. Proofs as programs. Technical Report TR 82-530, Cornell University, February 1983. Also published as [37].

[37] J. L. Bates and R. L. Constable. Proofs as programs. *ACM Transactions on Programming Languages and Systems*, 7:113–136, 1985.

[38] F. L. Bauer and K. Samelson. *Language Hierarchies and Interfaces*. Number 46 in Lecture Notes in Computer Science. Springer-Verlag, 1976.

[39] F. L. Bauer and H. Wössner. *Algorithmic Language and Program Development*. Springer-Verlag, 1982.

[40] H. Bekič. Towards a mathematical theory of processes. Technical Report TR 25.125, IBM Lab. Vienna, 1971.

[41] H. Bekič, D. Bjørner, W. Henhapl, C. B. Jones, and P. Lucas. A formal definition of a PL/I subset. Technical Report 25.139, IBM Laboratory Vienna, December 1974.

[42] H. Bekič, H. Izbicki, C. B. Jones, and F. Weissenböck. Some experiments with using a formal language definition in compiler development. Laboratory Note LN 25.3.107, IBM Laboratory, Vienna, December 1975.

[43] H. Bekič and C. B. Jones, editors. *Programming Languages and Their Definition*, volume 177 of *Lecture Notes in Computer Science*. Springer-Verlag, 1984.

[44] H. Bekič and K. Walk. Formalization of storage properties. In *[183]*, pages 28–61. 1971.

[45] U. Berger, W. Meixner, and B. Möller. Calculating a garbage collector. In M. Broy and M. Wirsing, editors, *Methods of Programming: Selected papers on the CIP-Project*, Lecture Notes in Computer Science, Vol. 544, pages 138–192. Springer-Verlag, 1991.

[46] J. A. Bergstra and J. W. Klop. A formalized proof system for total correctness of WHILE programs. Technical Report IW 175/81, Mathematisch Centrum, Amsterdam, October 1981. Preprint.

[47] J. A. Bergstra and J. V. Tucker. Two theorems about the completeness of Hoare's logic. Technical Report IW 165/81, Mathematisch Centrum, Amsterdam, April 1981. Preprint.

[48] D. Bjørner, C. A. R. Hoare, and H. Langmaack, editors. *VDM'90: VDM and Z – Formal Methods in Software Development*, volume 428 of *Lecture Notes in Computer Science*. Springer-Verlag, 1990.

[49] D. Bjørner, C. B. Jones, M. Mac an Airchinnigh, and E. J. Neuhold, editors. *VDM'87 – A Formal Definition at Work*, volume 252 of *Lecture Notes in Computer Science*. Springer-Verlag, 1987.

[50] D. Bjørner, editor. *Abstract Software Specifications: 1979 Copenhagen Winter School Proceedings*, volume 86 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1980.

[51] D. Bjørner and C. B. Jones, editors. *The Vienna Development Method: The Meta-Language*, volume 61 of *Lecture Notes in Computer Science*. Springer-Verlag, 1978.

[52] D. Bjørner and C. B. Jones. *Formal Specification and Software Development*. Prentice Hall International, 1982.

[53] D. Bjørner and O. N. Oest, editors. *Towards a Formal Description of Ada*, volume 98 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1980.

[54] S. R. Blamey. *Partial Valued Logic*. PhD thesis, Oxford University, 1980.

[55] A. Blikle. A metalanguage for naive denotational semantics. Technical Report 104, Consiglio Nazionale Delle Ricerche, ETS, Pisa, 1983.

[56] A. Blikle. Three-valued predicates for software specification and validation. In *[57]*, pages 243–266, 1988.

[57] R. Bloomfield, L. S. Marshall, and R. B. Jones, editors. *VDM'88: VDM – The Way Ahead*, volume 328 of *Lecture Notes in Computer Science*. Springer-Verlag, 1988.

[58] E. K. Blum, M. Paul, and S. Takasu, editors. *Mathematical Studies of Information Processing*, volume 75 of *Lecture Notes in Computer Science*. Springer-Verlag, 1979.

[59] A. Borning. Computer system reliability and nuclear war. *Communications of the ACM*, 30:112–131, February 1987.

[60] K. Bothe. Specification and verificiation of abstract data types. Seminarbericht 13, Humboldt-Universität zu Berlin, April 1979.

[61] K. Bothe. A generalized abstract data type concept. Preprint. (Neue Folge) 3, Humboldt-Universität zu Berlin, 1980.

[62] K. Bothe. An algorithmic interface structure for PASCAL compilers: Compiler portability by modularization. Seminarbericht 73, Humboldt-Universität zu Berlin, September 1985.

[63] R. S. Boyer and J. S. Moore. *A Computational Logic*. Academic Press, 1979.

[64] R. S. Boyer and J. S. Moore. *The Correctness Problem in Computer Science*. International Lecture Series in Computer Science. Academic Press, London, 1981.

[65] R. S. Boyer and J. S. Moore. A verification condition generator for FORTRAN. In *[64]*, pages 9–102. Academic Press, 1981.

[66] P. Brinch Hansen. *Operating System Principles*. Prentice-Hall Series in Automatic Computation. Prentice-Hall, 1973.

[67] P. Brinch Hansen. Concurrent Pascal – a programming language for operating system design. Technical Report 10, Information Science, Cal. Tech., April 1974.

[68] P. Brinch Hansen. A programming methodology for operating system design. In J. L. Rosenfeld, editor, *Information Processing 74*, pages 394–397, 1974. Proceedings of IFIP'74.

[69] P. Brinch Hansen. The programming language concurrent Pascal. *IEEE Transactions on Software Engineering*, 1:199–207, June 1975.

[70] P. Brinch Hansen. The programming language concurrent Pascal. In *[38]*, pages 82–110. 1976.

[71] P. Brinch Hansen. The Solo operating system: A concurrent pascal program. *Software — Practice and Experience*, 6:141–149, 1976.

[72] P. Brinch Hansen. Distributed processes: A concurrent programming concept. *Communications of the ACM*, 21:934–941, 1978.

[73] P. Brinch Hansen. EDISON – a multiprocessor language. Technical Report Unnumbered, University of Southern California, Computer Science Department, September 1980.

[74] C. Bron and M. M. Fokkinga. Exchanging robustness of a program for a relaxation of its specification. Memorandum 178, Twente University of Technology, The Netherlands, September 1977.

[75] C. Bron, M. M. Fokkinga, and A. C. M. de Haas. A proposal for dealing with abnormal termination of programs. Memorandum 150, Twente University of Technology, The Netherlands, November 1976.

[76] S. D. Brookes. A model for communicating sequential processes. Technical Report CMU-CS-83-149, Department of Computer Science, Carnegie-Mellon University, January 1983.

[77] S. D. Brookes. A semantics and proof system for communicating processes. Technical Report CMU-CS-83-134, Department of Computer Science, Carnegie-Mellon University, May 1983.

[78] S. D. Brookes, C. A. R. Hoare, and A. W. Roscoe. A theory of communicating sequential processes. *Journal of the ACM*, 31:560–599, July 1984.

[79] M. Broy. Denotational semantics of concurrent programs with shared memory. In M. Fontet and K. Mehlhorn, editors, *STACS 84: Symposium of Theoretical Aspects of Computer Science, Paris, April 1984*, volume 166 of *Lecture Notes in Computer Science*, pages 163–173. Springer-Verlag, 1984.

[80] M. Broy. Semantics of communicating processes. *Information & Control*, 61:202–246, 1984.

[81] M. Broy. A theory for nondeterminism, parallelism, communication and concurrency. *Theoretical Computer Science*, 45:1–61, 1986.

[82] M. Broy and F. L. Bauer. A systematic approach to language constructs for concurrent programs. *Science of Computer Programming*, 4:103–139, 1984.

[83] M. Broy and G. Schmidt. *Theoretical Foundations of Programming Methodology*. NATO Advanced Study Institutes Series. D. Reidel, 1982. Lecture notes of an International Summer School.

[84] J. R. Buchanan and D. L. Luckham. On automating the construction of programs. Technical Report STAN-CS-74-433, Computer Science Department, Stanford University, May 1974.

[85] W. H. Burge. The evaluation, classification and interpretation of expressions. In *Proceedings of the 19th ACM National Conference, 1964*, 1964. Paper A1.4.

[86] W. H. Burge. Proving the correctness of a compiler. Technical Report RC-2111, IBM Yorktown Heights, New York, June 1968.

[87] R. M. Burstall. Semantics of assignment. In E. Dale and D. Michie, editors, *Machine Intelligence, 2*, pages 3–20. Edinburgh University Press, 1967.

[88] R. M. Burstall. Formal description of program structure and semantics in first order logic. In B. Meltzer and D. Michie, editors, *Machine Intelligence 5*, pages 79–98. Edinburgh University Press, 1969.

[89] R. M. Burstall. Proving properties of programs by structural induction. *Computer Journal*, 12:41–48, 1969. Earlier available as Experimental Programming Report, No. 17, DMIP, Edinburgh, 1968.

[90] R. M. Burstall. Program proving as hand simulation with a little induction. In J. L. Rosenfeld, editor, *Information Processing 74*, pages 308–312, 1974. Proceedings of IFIP'74.

[91] R. M. Burstall and J. Darlington. A transformation system for developing recursive programs. *Journal of the ACM*, 24:44–67, 1977.

[92] R. M. Burstall and J. A. Goguen. The semantics of CLEAR, a specification language. In *in [50]*, pages 292–332. 1980.

[93] R. M. Burstall and J. A. Goguen. An informal introduction to specifications using CLEAR. In *[64]*, pages 185–214. 1981.

[94] R. M. Burstall and P. J. Landin. Programs and their proofs: an algebraic approach. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 17–43. Edinburgh University Press, 1969.

[95] J. N. Buxton and B. Randell, editors. *Software Engineering Techniques*. NATO Science Committee, 1970. Report on a conference Rome, Italy, 27th to 31st October 1969.

[96] M. Campbell-Kelly. Christopher Strachey, 1916–1975: A biographical note. *Annals of the History of Computing*, 7:19–42, January 1985.

[97] A. Caracciolo di Forino and L. Carlucci. On and algorithmic interpretation of the formal definition of PL/I. Internal Note B 69-4, Consiglio Nazionale delle Ricerche, University of Pisa, April 1969.

[98] L. Cardelli. *An Algebraic Approach to Hardware Description and Verification*. PhD thesis, Computer Science Dept., University of Edinburgh, 1982.

[99] L. Cardelli and P. Wegner. On understanding types, data abstraction, and polymorphism. *ACM Computing Surveys*, 17:471–522, December 1985.

[100] A. K. Chandra and Z. Manna. Program schemas with equality. Technical Report CS-250, Computer Science Department, Stanford University, December 1971.

[101] K. M. Chandy and J. Misra. *Parallel Program Design: A Foundation*. Addison-Wesley, 1988.

[102] G. Chroust, editor. *Heinz Zemanek – Ein Computerpionier*. R. Oldenbourg, 1985.

[103] A. Church. *The Calculi of Lambda-Conversion*. Princeton University Press, 1941.

[104] CIP Language Group. *The Munich Project CIP, Volume I: The Wide Spectrum Language CIP-L*, volume 183 of *Lecture Notes in Computer Science*. Springer-Verlag, 1985.

[105] CIP System Group. *The Munich Project CIP, Volume II: The Program Transformation System CIP-S*, volume 292 of *Lecture Notes in Computer Science*. Springer-Verlag, 1987.

[106] M. Clint and C. A. R. Hoare. Program proving: Jumps and functions. *Acta Informatica*, 1:214–224, 1972.

[107] J. Coenen, W.-P. de Roever, and J. Zwiers. Assertional data reification proofs: Survey and perspective. In J. M. Morris and R. Shaw, editors, *4th Refinement Workshop*, pages 97–114. Springer-Verlag, 1991.

[108] A. Colmerauer. PROLOG and infinite trees. In K. L. Clark and S. A. Tärnlund, editors, *Logic Programming*. Academic Press, 1982.

[109] R. L. Constable. Constructive mathematics and automatic program writers. In C. V. Freiman, editor, *Information Processing 71*, volume 1, pages 229–233. North-Holland, 1971. Proceedings of IFIP'71.

[110] R. L. Constable et al. *Implementing Mathematics with the Nuprl Proof Development System*. Prentice-Hall, 1986.

[111] D. C. Cooper. The equivalence of certain computations. *BCS, Computer Journal*, 9:45–52, 1966.

[112] D. C. Cooper. Theorem-proving in computers. In L. Fox, editor, *Advances in Programming and Non-numerical Computation*, pages 155–182. Pergamon Press, 1966.

[113] D. C. Cooper. Mathematical proofs about computer programs. In N. L. Collins and D. Michie, editors, *Machine Intelligence, 1*, pages 17–28. Olliver and Boyd, 1967.

[114] D. C. Cooper. Some transformations and standard forms of graphs, with applications to computer programs. In E. Dale and D. Michie, editors, *Machine Intelligence, 2*, pages 21–32. Edinburgh University Press, 1967.

[115] D. C. Cooper. Program schemes, programs and logic. In *[183]*, pages 62–70. 1971.

[116] P. Cousot. A Hoare-style axiomatization of Burstall's intermittent assertions method for non-deterministic programs. Technical Report LRIM-83-04, Metz University, France, September 1973.

[117] D. Craigen. A technical review of four verification systems: Gypsy, Affirm, FDM and Revised Special. Technical Report FR-85-5401-01, I. P. Sharp Associates, August 1985.

[118] O-J. Dahl. Can program proving be made practical? In M. Amirchahy and D. Néel, editors, *EEC-Crest Course on Programming Foundations*, pages 57–114. IRIA, 1978. Also printed as Technical Report 33 of Institute of Informatics, University of Oslo.

[119] O.-J. Dahl, E. W. Dijkstra, and C. A. R. Hoare, editors. *Structured Programming*. Academic Press, 1972.

[120] O.-J. Dahl, D. F. Langmyhr, and O. Owe. Preliminary report on the specification and programming language ABEL. Technical Report 106, University of Oslo, Institute of Informatics, University of Oslo, December 1986.

[121] O.-J. Dahl, B. Myhrhaug, and K. Nygaard. SIMULA 67 common base language. Technical Report S-2, Norwegian Computing Center, Oslo, 1968.

[122] J. Darlington. *A Semantic Approach to Automatic Program Improvement*. PhD thesis, University of Edinburgh, 1972.

[123] J. Darlington and R. M. Burstall. A system which automatically improves programs. *Acta Informatica*, 6:41–60, 1976.

[124] M. Davis. *The Undecidable*. Raven Press, 1965.

[125] J. de Bakker. *Mathematical Theory of Program Correctness*. Prentice-Hall International, 1980.

[126] J. W. de Bakker. Formal definition of algorithmic languages, with an application to the definition of ALGOL 60. Technical Report MR–74, Stichting Mathematisch Centrum, May 1965.

[127] J. W. de Bakker. Axiomatics of simple assignment statements. Technical Report 94, Mathematisch Centrum, Amsterdam, June 1968.

[128] J. W. de Bakker. Semantics of programming languages. Technical Report Unnumbered, Mathematical Centre, Amsterdam, 1968.

[129] J. W. de Bakker. Semantics of programming languages. In J. T. Tou, editor, *Advances in Information Systems Science*, volume 2, pages 173–227. Plenum Press, 1969.

[130] J. W. de Bakker. Axiom systems for simple assignment statements. In *[183]*, pages 1–22. 1971.

[131] J. W. de Bakker. Recursive procedures. Draft copy of Mathematical Centre Tract 24, August 1971.

[132] J. W. de Bakker. Inleiding bewijsmethoden. *Mathematical Centre Syllabus*, 21(1):1–17, 1975.

[133] J. W. de Bakker. Correctness proofs for assignment statements. Technical Report IW 55/77, Mathematisch Centrum, Amsterdam, January 1977. Preprint.

[134] J. W. de Bakker. Recursive programs as predicate transformers. Technical Report IW 83/77, Mathematisch Centrum, Amsterdam, June 1977. Preprint.

[135] J. W. de Bakker, J. W. Klop, and J.-J.Ch Meyer. Correctness of programs with function procedures. Technical Report IW 170/81, Mathematisch Centrum, Amsterdam, July 1981. Preprint.

[136] J. W. de Bakker and D. Scott. A theory of programs. Manuscript notes for IBM Seminar, Vienna, August 1969.

[137] J. W. de Bakker and J. I. Zucker. Processes and the denotational semantics of concurrency. Technical Report IW 209/82, Mathematisch Centrum, Amsterdam, September 1982. Preprint.

[138] N. G. de Bruijn. The mathematical language AUTOMATH – its usage and some of its extensions. In *Symposium on Automatic Demonstration*, volume 125 of *Lecture Notes in Mathematics*, pages 29–61. Springer-Verlag, 1970.

[139] R. de Nicola. Two complete axiom systems for a theory of communicating sequential processes. Technical Report CSR-154-83, Department of Computer Science, University of Edinburgh, December 1983.

[140] W.-P. de Roever. Recursion and parameter mechanisms: An axiomatic approach. In J. Loeckx, editor, *Automata Languages and Programming*, volume 14 of *Lecture Notes in Computer Science*. Springer-Verlag, 1974.

[141] W.-P. de Roever. Call-by-value versus call-by-name: A proof theoretic comparison. Technical Report IW 23/76, Mathematical Center, Amsterdam, September 1976.

[142] W.-P. de Roever. Dijkstra's predicate transformer, non-determinism, recursion and termination. Technical Report 37, I.R.I.S.A., University of Rennes, 1976.

[143] W.-P. de Roever. The quest for compositionality: A survey of assertion-based proof systems for concurrent programs: Part I: Concurrency based on shared variables. In *[466]*, pages 181–205, 1985.

[144] W.-P. de Roever, Jr. *Recursive Program Schemes: Semantics and Proof Theory*. PhD thesis, Mathematisch Centrum, Amsterdam, 1974.

[145] P. Degano and U. Montanari. Distributed systems, partial ordering of events, and event structures. In M. Broy, editor, *Control Flow and Data Flow: Concepts of Distributed Programming – NATO ASI Series F: Computer and System Sciences, Vol. 14*, pages 7–106. Springer-Verlag, 1985.

[146] P. Degano, R. De Niccola, and U. Montanari. Partial ordering derivations for CCS. In L. Budach, editor, *Fundamentals of Computation Theory,FCT 85. Cottbus, GDR, September 1985*, volume 199 of *Lecture Notes in Computer Science*, pages 520–533. Springer-Verlag, 1985.

[147] R. A. DeMillo, R. J. Lipton, and A. J. Perlis. Social processes and proofs of theorems and programs. *Communications of the ACM*, 22:271–280, May 1979.

[148] B. T. Denvir, W. T. Harwood, M. I. Jackson, and M. J. Wray. *The Analysis of Concurrent Systems: Cambridge, September 1983, Proceedings of a Workshop*, volume 207 of *Lecture Notes in Computer Science*. Springer Verlag, Berlin, 1985.

[149] N. Dershowitz and Z. Manna. Inference rules for program annotation. Technical Report STAN-CS-77-631, Computer Science Department, Stanford University, October 1977.

[150] E. W. Dijkstra. Recursive programming. *Numerische Mathematik*, 2:312–318, 1960.

[151] E. W. Dijkstra. Programming considered as a human activity. In W. A. Kalenich, editor, *Information Processing 1965. Proceedings of the IFIP Congress*, pages 213–217, New York City, May 1965.

[152] E. W. Dijkstra. A constructive approach to the problem of program correctness. *BIT*, 8:174–186, 1968.

[153] E. W. Dijkstra. Cooperating sequential processes. In F. Genuys, editor, *Programming Languages*, pages 43–112. Academic Press, New York, 1968.

[154] E. W. Dijkstra. Go to statement considered harmful. *Communications of the ACM*, 11(3):147–148, 1968.

[155] E. W. Dijkstra. A short introduction to the art of programming. Technisch Hogeschool Eindhoven, EWD-316, 1971.

[156] E. W. Dijkstra. The humble programmer. *Communications of the ACM*, 15:859–866, 1972.

[157] E. W. Dijkstra. Guarded commands, nondeterminacy and formal derivation of programs. *Communications of the ACM*, 18:453–457, 1975.

[158] E. W. Dijkstra. A synthesis emerging? In *[164]*, pages 147–160. Springer-Verlag, 1975. EWD508-0.

[159] E. W. Dijkstra. *A Discipline of Programming*. Prentice-Hall, 1976.

[160] E. W. Dijkstra. Guarded commands, non-determinacy and a calculus for the derivation of programs. In *[38]*, pages 111–124. 1976.

[161] E. W. Dijkstra. On-the-Fly garbage collection: An exercise in cooperaton. In *[38]*, pages 43–56. 1976.

[162] E. W. Dijkstra. A personal summary of the Gries-Owicki theory. In *[164]*, pages 188–199. Springer-Verlag, 1976. EWD554-0.

[163] E. W. Dijkstra. Introduction: Why correctness must be a mathematical concern. In *[64]*, pages 1–8. Academic Press, 1981.

[164] E. W. Dijkstra. *Selected Writings on Computing: A Personal Perspective*. Texts and Monographs in Computer Science. Springer-Verlag, 1982.

[165] E. W. Dijkstra. A tutorial on the split binary semaphore. In *[83]*, pages 555–565. 1982.

[166] E. W. Dijkstra and C. S. Scholten. *Predicate Calculus and Program Semantics*. Springer-Verlag, 1990.

[167] B. Dömölki, Zs. Farkas, and E. Sántáné–Tóth. On the formal description of software objects. In *Second Hungarian Computer Science Conference, Budapest, 27 June–2 July, 1977*, pages 338–361, 1977. Preprints I.

[168] J. E. Donahue. Complementary definitions of programming language semantics. Technical Report CSRG-62, University of Toronto, Canada, November 1975.

[169] J. E. Donahue. *Complementary Definitions of Programming Language Semantics*, volume 42 of *Lecture Notes in Computer Science*. Springer-Verlag, 1976.

[170] J. E. Donahue and A. Demers. Data types are values. Technical Report CSL-83-5, Xerox Corporation, Palo Alto, March 1984.

[171] J. E. Donahue, J. D. Gannon, J. V. Guttag, and J. J. Horning. Three approaches to reliable software: Language design, dyadic specification, complementary semantics. Technical Report CSRG-45, Computer Systems Research Group, University of Toronto, December 1974.

[172] V. Donzeau-Gouge, G. Kahn, and B. Lang. A complete machine-checked definition of a simple programming language using denotational semantics. Research Report 330, IRIA Laboria, France, October 1978.

[173] V. Donzeau-Gouge, G. Kahn, and B. Lang. On the formal definition of ADA. In *[301]*, pages 475–489. Springer-Verlag, 1980.

[174] A. J. W. Duijvestijn. Correctness proof of an in-place permutation. *BIT*, pages 318–324, 1972.

[175] F. G. Duncan. Possibilities for refining an object program compiled with an Algol translator. *BIT*, 5:85–95, 1965.

[176] H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 1: Equations and Initial Semantics*. EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1985.

[177] H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 2: Module Specifications and Constraints*. EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1990.

[178] S. Eilenberg and C. C. Elgot. Iteration and recursion. Technical Report RC 2148, IBM Research, July 1968.

[179] C. C. Elgot. A notion of interpretability of algorithms in algorithms. Technical Report TR 25.068, IBM Laboratory, Vienna, August 1966.

[180] C. C. Elgot. Algebraic theories and program schemes. In *[183]*, pages 71–88. 1971.

[181] C. C. Elgot. Remarks on one-argument program schemes. In *[512]*, pages 59–64. 1972.

[182] C. C. Elgot and A. Robinson. Random access stored-program machines: An approach to programming languages. *Journal of the ACM*, 11:365–399, October 1964.

[183] E. Engeler. *Symposium on Semantics of Algorithmic Languages*. Number 188 in Lecture Notes in Mathematics. Springer-Verlag, 1971.

[184] A. P. Ershov. On programming arithmetic operators. *Communications of the ACM*, 1(8):3–6, August 1958.

[185] A. P. Ershov. Parallel programming. Technical Report CS-224, Computer Science Department, Stanford University, July 1971.

[186] A. P. Ershov. Theory of program schemata. In C. V. Freiman, editor, *Information Processing 71*, volume 1, pages 28–46. North-Holland, 1971. Proceedings of IFIP'71.

[187] A. P. Ershov. Axiomatics for memory allocation. *Acta Informatica*, 6(1):61–76, 1976.

[188] A. P. Ershov. *Origins of Programming: Discourses on Methodology*. Springer-Verlag, 1990. Original Russian in 1977.

[189] A. P. Ershov and G. D. Chinin. Design specifications of a quality compiler factory. In *Constructing Quality Software. International Federation for Information Processing, Technical Committee 2 on Programming. Working Conference*, pages 203–228, Novosibirsk, May 1977.

[190] A. Evans. *Syntax Analysis by a Production Language*. PhD thesis, Carnegie Institute of Technology, 1965.

[191] A. D. Falkoff, K. E. Iverson, and E. H. Sussenguth. A formal description of SYSTEM/360. *IBM Systems Journal*, 3(2 and 3), 1964.

[192] J. A. Feldman. *A Formal Semantics for Computer Oriented Languages*. PhD thesis, Carnegie Institute of Technology, May 1964. Reprinted June 1965.

[193] L. Flon. *On the Design and Verification of Operating Systems*. PhD thesis, Carnegie-Mellon University, May 1977.

[194] L. Flon and N. Suzuki. Consistent and complete proof rules for the total correctness of parallel programs. Technical Report CSl-78-6, Xerox, Palo Alto, 1978.

[195] R. W. Floyd. On the nonexistence of a phrase structure grammar for ALGOL 60. *Communications of the ACM*, 5:483–484, 1962.

[196] R. W. Floyd. The syntax of programming langauges–a survey. *IEEE Transactions on Electronic Computers*, 13, 4:346–353, August 1964.

[197] R. W. Floyd. Assigning meanings to programs. In *Proc. Symp. in Applied Mathematics, Vol.19: Mathematical Aspects of Computer Science*, pages 19–32. American Mathematical Society, 1967.

[198] M. Foley and C. A. R. Hoare. Proof of a recursive program: Quicksort. *BCS, Computer Journal*, 14:391–395, November 1971.

[199] N. Francez, C. A. R. Hoare, D. J. Lehmann, and W.-P. de Roever. Semantics of nondeterminism, concurrency and communication. *Journal of Computer and System Sciences*, 19:290–308, December 1979.

[200] N. Francez and A. Pnueli. A proof method for cyclic programs. *Acta Informatica*, 9:133–157, 1978.

[201] R. Gerth. A sound and complete Hoare axiomatization of the Ada-Rendezvous. Technical Report RUU-CS-82-5, University of Utrecht Vakgroep Informatica, April 1982. Extended Abstract.

[202] J. A. Goguen. On homomorphisms, correctness, termination, unfoldments, and equivalence of flow diagram programs. *Journal of Computer and System Sciences*, 8:333–365, 1974.

[203] J. A. Goguen and K. Parsaye-Ghomi. Algebraic denotational semantics using parameterized abstract models. Technical Report CSL-119, Computer Science Laboratory, SRI International, February 1981.

[204] J. A. Goguen and J. W. Thatcher. Initial algebra semantics. Technical Report RC 4865, IBM Yorktown Heights, 1974. Extended Abstract.

[205] J. A. Goguen, J. W. Thatcher, and E. G. Wagner. An initial algebra approach to the specification, correctness, and implementation of abstract data types. Technical Report RC 6487, IBM T.J. Watson Research Center, October 1976.

[206] J. A. Goguen, J. W. Thatcher, E. G. Wagner, and J. B. Wright. Initial algebra semantics. Technical Report RC 5243, IBM Yorktown Heights, January 30th 1975.

[207] J. A. Goguen, J. W. Thatcher, E. W. Wagner, and J. B. Wright. Initial algebra semantics and continuous algebras. *Journal of the ACM*, 24:68–95, 1977.

[208] A. Goldberg and D. Robson. *Smalltalk-80: The Language and its Implementation*. Addison-Wesley, 1983.

[209] H. H. Goldstine and J. von Neumann. Planning and coding of problems for an electronic computing instrument, 1947. Part II, Vol. 1 of a Report prepared for U.S. Army Ord. Dept.; also published as pages 80–151 of [569].

[210] D. I. Good, R. M. Cohen, C. G. Hoch, L. W. Hunter, and D. F. Hare. Report on the language Gypsy, version 2.0. Technical Report ICSCA-CMP-10, University of Texas at Austin, September 1978.

[211] M. Gordon. Operational reasoning and denotational semantics. Technical Report STAN-CS-75-506, Stanford University, Computer Science Department, August 1975.

[212] M. Gordon. Why higher-order logic is a good formalism for specifying and verifying hardware. In G. Milne and P.A. Subrahmanyam, editors, *Formal Aspects of VLSI Design*, pages 153–177. North-Holland, 1986.

[213] M. J. Gordon, A. J. Milner, and C. P. Wadsworth. *Edinburgh LCF*, volume 78 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1979.

[214] M. J. C. Gordon. *The Denotational Description of Programming Languages: An Introduction*. Springer-Verlag, 1979.

[215] M. J. C. Gordon. *Programming Language Theory and its Implementation*. Prentice-Hall International, 1988.

[216] S. Gorn. Common programming language task: Final report No. AD59UR1. Contract No. DA-36-039-SC-75047, DA Proj. No. 3-28-01-201, PR and C No. 58-ELC/D-4457, Part I, Section 5: On The Logical Design of Formal Mixed Languages, 1959.

[217] S. Gorn. Specification languages for mechanical languages and their processors – a baker's dozen. *Communications of the ACM*, 4:532–542, 1961.

[218] D. Gries. An exercise in proving parallel programs correct. In *[38]*, pages 57–81. 1976.

[219] D. Gries, editor. *Programming Methodology: A Collection of Articles by Members of IFIP W.G. 2.3*. Springer-Verlag, 1978.

[220] D. Gries. *The Science of Programming*. Springer-Verlag, 1981.

[221] D. Gries and G. Levin. A procedural call proof rule (with a simple explanation). Technical Report TR 79-379, Cornell University, New York, May 1979.

[222] O. Grümberg, N. Francez, J. A. Maskowsky, and W.-P. de Roever. A proof rule for fair termination of guarded commands. Technical Report RUU-CS-81-2, University of Utrecht, The Netherlands, January 1981.

[223] Y. Gurevich. Logic and the challenge of computer science. Technical Report CRL-TR-10-85, Computing Research Laboratory, University of Michigan, September 1985.

[224] J. V. Guttag. *The Specification and Application to Programming of Abstract Data Types*. PhD thesis, University of Toronto, Computer Systems Research Group, September 1975. CSRG-59.

[225] J. V. Guttag, J. J. Horning, and R. L. London. A proof rule for Euclid procedures. Technical Report ISI/RR-77-60, University of Southern California, Information Sciences Institute, May 1977.

[226] J. V. Guttag, J. J. Horning, and R. L. London. A proof rule for Euclid procedures. Technical Report ISI/RR-77-60, University of Southern California, Information Sciences Institute, May 1977.

[227] J. V. Guttag, J. J. Horning, and J. M. Wing. Larch in five easy pieces. Technical Report 5, DEC, SRC, July 1985.

[228] J. V. Guttag, E. Horowitz, and D. R. Musser. Abstract data types and software validation. Technical Report ISI/RR-76-48, University of Southern California, Information Sciences Institute, August 1976.

[229] J. V. Guttag, E. Horowitz, and D. R. Musser. The design of data type specifications. Technical Report ISI/44-76-49, University of Southern California, Information Sciences Institute, November 1976.

[230] A. Hansal. A formal definition of a relational data base system. Technical Report UKSC 0080, IBM UK Scientific Centre, Peterlee, Co. Durham, June 1976.

[231] S. L. Hantler and A. C. Chibib. *Effigy Reference Manual*. IBM Yorktown Heights, January 20 1975. Technical Report RC 5225.

[232] S. L. Hantler and J. C. King. An introduction to proving the correctness of programs. *ACM Computing Surveys*, 8:331–353, September 1976.

[233] D. Harel. On the total correctness of nondeterministic programs. Technical Report RC 7691, IBM, Yorktown Heights, New York, May 1979.

[234] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. Technical Report ECS-LFCS-87-23, Dept of Computer Science, University of Edinburgh, March 1987.

[235] R. Harper, R. Milner, and M. Tofte. The definition of standard ML version 2. Technical Report ECS-LFCS-88-62, Deptartment of Computer Science, University of Edinburgh, August 1988.

[236] J. He, C. A. R. Hoare, and J. W. Sanders. Data refinement refined: Resumé. In B. Robinet and R. Wilhelm, editors, *ESOP'86*, volume 213 of *Lecture Notes in Computer Science*, pages 187–196. Springer-Verlag, 1986.

[237] E. C. R. Hehner. *The Logic of Programming*. Prentice-Hall International, 1984.

[238] E. C. R. Hehner, L. E. Gupta, and A. J. Malton. Predicative methodology. *Acta Informatica*, 23:487–505, 1986.

[239] P. Henderson and J. H. Morris. A lazy evaluator. Technical Report 85, University of Newcastle-upon-Tyne, January 1976. Believed to be a preprint of a paper which appeared in the ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages in Atlanta, January 1976.

[240] W. Henhapl. A proof of correctness for the reference mechanism to automatic variables in the F-compiler. Technical Report LN 25.3.048, IBM Laboratory Vienna, Austria, November 1968.

[241] W. Henhapl and C. B. Jones. The block concept and some possible implementations, with proofs of equivalence. Technical Report 25.104, IBM Laboratory Vienna, April 1970.

[242] W. Henhapl and C. B. Jones. On the interpretation of GOTO statements in the ULD. Technical Report LN 25.3.065, IBM Laboratory, Vienna, March 1970.

[243] W. Henhapl and C. B. Jones. A run-time mechanism for referencing variables. *Information Processing Letters*, 1:14–16, 1971.

[244] W. Henhapl and C. B. Jones. A formal definition of ALGOL 60 as described in the 1975 modified report. In *[51]*, pages 305–336. Springer-Verlag, 1978.

[245] M. Hennessy and R. Milner. On observing nondeterminism and concurrency. In J. W.de Bakker and J. van Leeuwen, editors, *Automata, Languages and Programming. Seventh Colloqium, Noordwijkerhout, July 1980*, volume 85 of *Lecture Notes in Computer Science*, pages 299–309. Springer-Verlag, 1980.

[246] M. C. B. Hennessy and G. D. Plotkin. Full abstraction for a simple parallel programming language. In J. Bečvář, editor, *Mathematical Foundations of Computer Science 1979. Proceedings, Olomouc, Czechoslovakia*, volume 74 of *Lecture Notes in Computer Science*, pages 108–120. Springer-Verlag, 1979.

[247] P. Hitchcock. *An Approach to Formal Reasoning about Programs*. PhD thesis, Department of Computer Science, University of Warwick, June 1974.

[248] P. Hitchcock and D. Park. Induction rules and termination proofs. In *Proceedings of a Symposium on Automata, Languages and Programming, France 3-7 July, 1972*, pages 225–251. North-Holland, 1973.

[249] C. A. R. Hoare. Algorithm 63, Partition; Algorithm 64, Quicksort; Algorithm 65, Find. *Communications of the ACM*, 4(7):321–322, July 1961.

[250] C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12:576–580, 583, October 1969.

[251] C. A. R. Hoare. Procedures and parameters: An axiomatic approach. In E. Engeler, editor, *Symposium on Semantics of Algorithmic Languages*, volume 188 of *Lecture Notes in Mathematics*, pages 102–116. Springer-Verlag, 1971.

[252] C. A. R. Hoare. Proof of a program: FIND. *Communications of the ACM*, 14:39–45, January 1971.

[253] C. A. R. Hoare. Notes on data structuring. In O.-J. Dahl, E. W. Dijkstra, and C. A. R. Hoare, editors, *Structured Programming*, pages 83–174. Academic Press, 1972.

[254] C. A. R. Hoare. Proof of a structured program: 'the sieve of Eratosthenes'. *Computer Journal*, 15:321–325, November 1972.

[255] C. A. R. Hoare. Proof of correctness of data representations. *Acta Informatica*, 1:271–281, 1972.

[256] C. A. R. Hoare. Towards a theory of parallel programming. In C. A. R. Hoare and R. Perrot, editors, *Operating System Techniques*, pages 61–71. Academic Press, 1972.

[257] C. A. R. Hoare. Monitors: An operating system structuring concept. *Communications of the ACM*, 17:549–557, October 1974.

[258] C. A. R. Hoare. Parallel programming: An axiomatic approach. *Computer Languages*, 1:151–160, June 1975.

[259] C. A. R. Hoare. Parallel programming: An axiomatic approach. In *[38]*, pages 11–42. 1976.

[260] C. A. R. Hoare. Proof of correctness of data representation. In *[38]*, pages 183–193. 1976.

[261] C. A. R. Hoare. Communicating sequential processes. *Communications of the ACM*, 21:666–677, August 1978.

[262] C. A. R. Hoare. Some properties of predicate transformers. *Journal of the ACM*, 25:461–80, July 1978.

[263] C. A. R. Hoare. A calculus of total correctness for communicating processes. *Science of Computer Programming*, 1:49–72, October 1981.

[264] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.

[265] C. A. R. Hoare. Programs are predicates. In C. A. R. Hoare and J.C. Shepherdson, editors, *Mathematical Logic and Programming Languages*, pages 141–154. Prentice-Hall, 1985.

[266] C. A. R. Hoare et al. Data refinement refined. Typescript, Programming Research Group, Oxford University., May 1985.

[267] C. A. R. Hoare, I. J. Hayes, J. He, C. Morgan, A. W. Roscoe, J. W. Sanders, I. H. Sørensen, J. M. Spivey, and B. A. Sufrin. The laws of programming. *Communications of the ACM*, 30:672–687, 1987. see Corrigenda in *ibid* 30:770.

[268] C. A. R. Hoare and J. He. The weakest prespecification. *Information Processing Letters*, 24:127–32, January 1987.

[269] C. A. R. Hoare, J. He, and J. W. Sanders. Prespecification in data refinement. *Information Processing Letters*, 25:71–76, May 1987.

[270] C. A. R. Hoare and C. B. Jones. *Essays in Computing Science*. Prentice Hall International, 1989.

[271] C. A. R. Hoare and J.R. Kennaway. A theory of non-determinism. In *Proceedings ICALP '80*, volume 85 of *Lecture Notes in Computer Science*, pages 338–350. Springer-Verlag, 1980.

[272] C. A. R. Hoare and P. E. Lauer. Consistent and complementary formal theories of the semantics of programming languages. *Acta Informatica*, 3:135–153, 1974.

[273] C. A. R. Hoare and J.C. Shepherdson, editors. *Mathematical Logic and Programming Languages*. Prentice-Hall, 1985. The papers in this book were first published in the Philosophical Transactions of the Royal Society Series A, Vol. 312, 1984.

[274] C. A. R. Hoare and N. Wirth. An axiomatic definition of the programming language Pascal. *Acta Informatica*, 2:335–355, 1973.

[275] A. Hodges. *Alan Turing: The Enigma*. Burnett Books, 1983. Vintage edition, 1992.

[276] A. Hoogewijs. Partial-predicate logic in computer science. *Acta Informatica*, 24:381–393, 1987.

[277] J. Hooman and W.-P. de Roever. The quest goes on: A survey of proof systems for partial correctness of CSP. In J. W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Current Trends in Concurrency*, volume 224 of *Lecture Notes in Computer Science*. Springer-Verlag, 1986.

[278] G. E. Hughes and M. J. Cresswell. *A Companion to Modal Logic*. Methuen, 1984.

[279] J. Hughes. Graph reduction with super-combinators. Technical Report PRG-28, Oxford University Computing Laboratory, Programming Research Group, June 1982.

[280] T. E. Hull, W. H. Enright, and A. E. Sedgwick. The correctness of numerical algorithms. *ACM SIGPLAN Notices*, 7(1):66–73, January 1972.

[281] S. Igarashi. *An Axiomatic Approach to the Equivalence Problems of Algorithms with Applications*. PhD thesis, University of Tokyo, 1964. Reprinted as Report of the Computer Centre University of Tokyo, No. 1, in 1968.

[282] S. Igarashi. A formalization of the descriptions of languages and the related problems in a Gentzen-Type formal system. RAAG Research Notes 80, Research Association of Applied Geometry, University of Tokyo, May 1964.

[283] S. Igarashi. On the equivalence of programs represented by Algol-like statements. *Report of the Computer Centre University of Tokyo*, 1(1), April–September 1968.

[284] S. Igarashi. Semantics of Algol-like statements. Technical Report CS167, Stanford University, Computer Science Dept, June 1970.

[285] S. Igarashi. Semantics of ALGOL-like statements. In *[183]*, pages 117–177. 1971.

[286] S. Igarashi. Admissibility of fixed-point induction in first-order logic of typed theories. Technical Report STAN-CS-72-287, Stanford University Computer Science Dept, May 1972.

[287] S. Igarashi. Automatic program verification I: A logical basis and its implementation. Technical Report STAN-CS-73-365, Stanford University Computer Science Dept, May 1973.

[288] INMOS. *occam 2: Reference Manual*. Prentice Hall, 1988.

[289] K. E. Iverson. *A Programming Language*. J. Wiley, 1962.

[290] C. B. Jones. A technique for showing that two functions preserve a relation between their domains. Technical Report LR 25.3.067, IBM Laboratory, Vienna, April 1970.

[291] C. B. Jones. Yet another proof of the correctness of block implementation. Technical Report LN 25.3.075, IBM Laboratory, Vienna, August 1970.

[292] C. B. Jones. Formal development of correct algorithms: an example based on Earley's recogniser. *ACM SIGPLAN Notices*, 7(1):150–169, January 1972.

[293] C. B. Jones. Formal development of programs. Technical Report 12.117, IBM Laboratory Hursley, April 1973.

[294] C. B. Jones. Formal definition in compiler development. Technical Report 25.145, IBM Laboratory Vienna, February 1976.

[295] C. B. Jones. Implementation bias in constructive specification of abstract objects. unpublished manuscript, September 1977.

[296] C. B. Jones. Constructing a theory of a data structure as an aid to program development. *Acta Informatica*, 11:119–137, 1979.

[297] C. B. Jones. *Software Development: A Rigorous Approach*. Prentice Hall International, 1980.

[298] C. B. Jones. Denotational semantics of goto: An exit formulation and its relation to continuations. In *[51]*, pages 278–304. Springer-Verlag, 1981.

[299] C. B. Jones. *Development Methods for Computer Programs including a Notion of Interference*. PhD thesis, Oxford University, June 1981. Printed as Technical Monograph No. PRG-25.

[300] C. B. Jones and P. Lucas. Proving correctness of implementation techniques. In E. Engeler, editor, *Symposium On Semantics of Algorithmic Languages*, volume 188 of *Lecture Notes in Mathematics*, pages 178–211. Springer-Verlag, 1971.

[301] N. D. Jones. *Semantics-Directed Compiler Generation. Proceedings of a Workshop, Aarhus, Denmark, January 1980*, volume 94 of *Lecture Notes in Computer Science*. Springer-Verlag, 1980.

[302] N. D. Jones and D. A. Schmidt. Compiler generation from denotational semantics. In *[301]*, pages 70–93. Springer-Verlag, 1980.

[303] G. Kahn. Normalisation et documentation des programmes. In *Synthése, Manipulation et Transformation de Programmes*, pages 97–110. IRIA sesori, May 1978. Journées D'études Sesori.

[304] G. Kahn. A preliminary theory for parallel programs. Research report 6, INRIA, France, January 1973.

[305] G. Kahn, editor. *Semantics of Concurrent Computation: Proceedings, Evian, France 1979*, volume 70 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1979.

[306] G. Kahn and D. MacQueen. Coroutines and networks of parallel processes. In B. Gilchrist, editor, *Information Processing'77*, pages 993–998, 1977.

[307] D. M. Kaplan. Correctness of a compiler for Algol-like programs. Artificial Intelligence Memo 48, Stanford University, July 1967.

[308] D. M. Kaplan. A formal theory concerning the equivalence of algorithms. Artificial Intelligence Memo 59, Stanford University, May 1968.

[309] E. W. Karlsen. The draft formal definition of Ada. Technical Report AdaFD/DDC/29/V 1.0, Dansk Datamatik Center, Denmark, March 23 1987.

[310] S. Katz and Z. Manna. Logical analysis of programs. Technical report, The Weizmann Institute of Science, Rehovot, Israel, September 1974. Unnumbered.

[311] S. Katz and Z. Manna. A practical approach to termination. Technical report, The Weizmann Institute of Science, Rehovot, Israel, September 1974. Unnumbered.

[312] R. A. Kemmerer. Verification assessment study: Final report, volume 1 overview, conclusions and future directions. Technical Report C3-CR01-86, Library No. S-228,204, National Computer Security Center, Maryland, USA, March 1986.

[313] J. C. King. *A Program Verifier*. PhD thesis, Department of Computer Science, Carnegie-Mellon University, 1969.

[314] J. C. King. A program verifier. In C. V. Freiman, editor, *Information Processing 71*, pages 234–249. North-Holland, 1971. Proceedings of IFIP'71.

[315] J. C. King. Symbolic execution and program testing. *Communications of the ACM*, 17(7):385–394, July 1976. Preprint available, May 1975.

[316] J. C. King and R. W. Floyd. An interpretation-oriented theorem prover over integers. *Journal of Computer and System Sciences*, 6:305–323, August 1972.

[317] S. C. Kleene. *Introduction to Metamathematics*. Van Nostrad, 1952.

[318] C. D. Kloos. *Semantics of Digital Circuits*, volume 285 of *Lecture Notes in Computer Science*. Springer-Verlag, 1987.

[319] D. E. Knuth. The remaining trouble spots in ALGOL 60. *Communications of the ACM*, 10:611–618, October 1967.

[320] D. E. Knuth. *Fundamental Algorithms*, volume I of *The Art of Computer Programming*. Addison-Wesley Publishing Company, 1968.

[321] D. E. Knuth. Semantics of context-free languages. *Mathematical Systems Theory*, 2:127–145, June 1968.

[322] D. E. Knuth. Examples of formal semantics. In *[183]*, pages 212–235. 1971.

[323] D. E. Knuth. The dangers of computer-science theory. In P. Suppes, L. Henkin, A. Joja, and Gr.C. Moisil, editors, *Studies in Logic and Foundations of Mathematics Vol. 74 (Proc. of the 4th International Congress for Logic, Methodology and Philosophy of Science, Bucharest, 1971)*, pages 189–195. North Holland Publishing Company, 1973.

[324] D. E. Knuth. *Sorting and Searching*, volume III of *The Art of Computer Programming*. Addison-Wesley Publishing Company, 1973.

[325] D. E. Knuth. Structured programming with GO TO statements. Technical Report STAN-CS-74-416, Computer Science Dept, Stanford University, May 1974.

[326] D. E. Knuth. Mathematics and computer science: Coping with finiteness. Technical Report STAN-CS-76-541, Computer Science Dept, Stanford University, February 1976.

[327] D. E. Knuth and L. T. Pardo. The early development of programming languages. In *[422]*, pages 197–273. 1976.

[328] B. Konikowska, A. Tarlecki, and A. Blikle. A three-valued logic for software specification and validation *tertium tamen datur*. In *[57]*, pages 218–242, 1988.

[329] R. Kowalski. Predicate logic as a programming language. In *Information Processing'74*, pages 569–574, 1974.

[330] R. Kowalski. Algorithm = logic + control. *Communications of the ACM*, 22:424–436, 1979.

[331] R. Kowalski. *Logic for Problem Solving*. North-Holland, 1979.

[332] G. Kreisel. Five notes on the application of proof theory to computer science. Technical Report 182, Inst. for Mathematical Studies in the Social Sciences, Stanford University, December 10th 1971.

[333] F. Kroger. *Temporal Logic of Programs*. EATCS Monographs on Theoretical Computer Science. Springer-Verlag, 1987.

[334] R. Kurki-Suonio. Towards better structured definitions of programming languages. Technical Report STAN-CS-75-500, Computer Science Dept, Stanford University, September 1975.

[335] W. Lamersdorf and J. W. Schmidt. Specification of Pascal/R — the formal semantic specification using VDM. Technical Report IFI-HH-B-74/80, University of Hamburg, July 1980.

[336] L. Lamport. Proving the correctness of mutiprocess programs. *IEEE Transactions on Software Engineering*, 3:125–143, 1977.

[337] L. Lamport. The 'Hoare Logic' of concurrent programs. Technical Report CSL-79, SRI International, November 1978. Revised 14 January 1980.

[338] L. Lamport. The 'Hoare logic' of concurrent programs. *Acta Informatica*, 14:21–37, 1980.

[339] L. Lamport. On interprocess communication. Technical Report 8, Digital Systems Research Center, Palo Alto, December 1985.

[340] L. Lamport. Control predicates are better than dummy variables for reasoning about program control. *ACM Transactions on Programming Languages and Systems*, 10:267–281, April 1988.

[341] L. Lamport. A theorem on atomicity in distributed algorithms. Technical Report 28, Digital Systems Research Center, Palo Alto, May 1988.

[342] L. Lamport. A temporal logic of actions. Technical Report 57, DEC, SRC, 1990.

[343] B. W. Lampson. A description of the Cedar language: A Cedar language reference manual. Technical Report CSL-83-15, Xerox Corporation, Palo Alto Research Center, California, December 1983. Printed November 1986.

[344] B. W. Lampson, J. J. Horning, R. L. London, J. G. Mitchell, and G. J. Popek. Report on the programming language Euclid. Technical Report CSL-81-12, Xerox, Palo Alto, October 1981.

[345] P. J. Landin. The mechanical evaluation of expressions. *Computer Journal*, 6:308–320, 1964.

[346] P. J. Landin. A correspondence between ALGOL-60 and Church's lambda-notation. Parts I and II. *Communications of the ACM*, 8:89–101, 158–165, 1965.

[347] P. J. Landin. A λ-calculus approach. In L. Fox, editor, *Advances in Programming and Non-numerical Computation*, pages 97–141. Pergamon Press, 1966.

[348] P. J. Landin. The next 700 programming languages. *Communications of the ACM*, 9:157–166, 1966.

[349] B. Lang. Threshold evaluation and the semantics of call by value, assignment and generic procedures. Technical Report 211, INRIA, France, January 1977.

[350] J. Laski. The morphology of prex – an essay in meta-algorithmics. In D. Michie, editor, *Machine Intelligence, 3*, pages 3–18. Edinburgh University Press, 1968.

[351] H. C. Lauer. *Correctness in Operating Systems*. PhD thesis, Carnegie-Mellon University, 1972.

[352] P. E. Lauer. Formal definition of ALGOL 60. Technical Report TR 25.088, IBM Laboratory Vienna, December 1968.

[353] P. E. Lauer. *Consistent Formal Theories of the Semantics of Programming Languages*. PhD thesis, Queen's University of Belfast, 1971. Printed as TR 25.121, IBM Lab. Vienna.

[354] J. A. N. Lee. The Vienna definition language: A generalization of instruction definitions. Paper prepared for submission to the SIGPLAN Symposium on Programming Language Definition, San Francisco, California, August 1969.

[355] D. J. Lehmann and M. B. Smyth. Algebraic specification of data types — a synthetic approach. Technical Report 115, Dept of Computer Studies, University of Leeds, September 1978.

[356] A. A. Letichevskii. Functional equivalence of discrete processors II. *Cybernetics (USA)*, 6(2):28–42, Mar–Apr 1970.

[357] G. M. Levin. *Proof Rules for Communicating Sequential Processes*. PhD thesis, Cornell University, August 1980.

[358] R. C. Linger, H. D. Mills, and B. L. Witt. *Structured Programming: Theory and Practice*. Addison-Wesley, 1979.

[359] R. J. Lipton. Reduction: A method of proving properties of parallel programs. *Communications of the ACM*, 18(12), December 1975.

[360] B. Liskov and J. Guttag. *Abstraction and Specification in Program Development*. MIT Press, 1986.

[361] B. Liskov and S. Zilles. Programming with abstract data types. Computation Structures Group Memo 99, Massachusetts Institute of Technology, March 1974.

[362] R. L. London. *A Computer Program for Discovering and Proving Sequential Recognition Rules for Well-formed Formulas Defined by a Backus Normal Form Grammar*. PhD thesis, Carnegie Institute of Technology, 1964.

[363] R. L. London. A correctness proof of the Fisher-Galler algorithm using inductive assertions. Technical Report 102, The University of Wisconsin, Computer Sciences Dept, October 1970.

[364] R. L. London. Experience with inductive assertions for proving programs correct. Technical Report 92, The University of Wisconsin, Computer Sciences Dept, May 1970.

[365] R. L. London. Proof of algorithms – a new kind of certification. *Communications of the ACM*, 13(6):371–373, 1970.

[366] R. L. London. Proving programs correct: Some techniques and examples. *BIT*, 10:168–182, 1970.

[367] R. L. London. Correctness of two compilers for a Lisp subset. Technical Report CS240, Computer Science Dept, Stanford University, October 1971.

[368] R. L. London. Experience with inductive assertions for proving programs correct. In *[183]*, pages 236–251. 1971.

[369] R. L. London, J. V. Guttag, J. J. Horning, B. W. Lampson, J. G. Mitchell, and G. J. Popek. Proof rules for the programming language Euclid. *Acta Informatica*, 10:1–26, 1978.

[370] P. Lucas. On the formalization of syntax and semantics of PL/I. Technical Report TR 25.060, IBM Vienna, November 1965.

[371] P. Lucas. Introduction to the method used for the formal definition of PL/I. Technical Report TR 25.081, IBM Vienna, 28th June 1968. Revised.

[372] P. Lucas. Two constructive realizations of the block concept and their equivalence. Technical Report TR 25.085, IBM Laboratory Vienna, June 1968.

[373] P. Lucas. Equivalence of the verification conditions of Floyd and Scott. LN 25.3.055, IBM Laboratory Vienna, 18th September 1969.

[374] P. Lucas. Formal definition of programming languages and systems. In C. V. Freiman, editor, *Information Processing 71. Proceedings of the IFIP Congress 1971*, volume 1, pages 291–297. North-Holland, 1971.

[375] P. Lucas. On the semantics of programming languages and software devices. In *[512]*, pages 41–57. 1972.

[376] P. Lucas. Formal semantics of programming languages: VDL. *IBM Journal of Research and Development*, 25(5):549–561, September 1981.

[377] P. Lucas. VDM: Origins, Hopes, and Achievements. In *[49]*, pages 1–18, 1987.

[378] P. Lucas and K. Walk. On the documentation of programming ideas. Paper presented at the European Patent-Seminar in Vienna, 1969.

[379] P. Lucas and K. Walk. *On The Formal Description of PL/I*, volume 6, Part 3 of *Annual Review in Automatic Programming*. Pergamon Press, 1969.

[380] D. Luckham. The resolution principal in theorem proving. In N. L. Collins and D. Michie, editors, *Machine Intelligence, 1*, pages 47–61. Olliver and Boyd, 1967.

[381] D. C. Luckham, D. M. R. Park, and M. S. Paterson. On formalised computer programs. *Journal of Computer and System Sciences*, 4:220–249, 1970.

[382] J. Łukasiewicz. O logice trójwartościowej (on three-valued logic). *Ruch Filozoficzny*, 5:169–171, 1920.

[383] S. MacLane and G. Birkoff. *Algebra*. Collier Macmillan, 1967.

[384] D. B. MacQueen. Models for distributed computing. Technical Report 351, INRIA, France, April 1979.

[385] Z. Manna. Formalization of properties of programs. Technical Memorandum AI-64, Stanford Artificial Intelligence Department, July 1968.

[386] Z. Manna. *Termination of Algorithms*. PhD thesis, Carnegie-Mellon University, April 1968.

[387] Z. Manna. The correctness of non-deterministic programs. Memo AI-95, Department of Computer Science, Stanford University, August 1969.

[388] Z. Manna. The correctness of programs. *Journal of Computer and System Sciences*, 3:119–127, 1969.

[389] Z. Manna. Mathematical theory of partial correctness. In *[183]*, pages 252–269. 1971.

[390] Z. Manna. *Mathematical Theory of Computation*. McGraw-Hill, 1974.

[391] Z. Manna. Verification of sequential programs: Temporal axiomation. Technical Report STAN-CS-81-877, Stanford University, Stanford, California, September 1981.

[392] Z. Manna and J. McCarthy. Properties of programs and partial function logic. In B. Meltzer and D. Michie, editors, *Machine Intelligence, 5*, pages 27–37. Edinburgh University Press, 1969.

[393] Z. Manna, S. Ness, and J. Vuillemin. Inductive methods for proving properties of programs. Technical Report 243, Computer Science Department, Stanford University, November 1971.

[394] Z. Manna and A. Pnueli. Formalization of properties of recursively defined functions. Technical Report 82, Department of Computer Science, Stanford University, March 1969.

[395] Z. Manna and A. Pnueli. Temporal verification of concurrent programs: the temporal framework. In *[64]*, pages 215–273. Academic Press, 1981.

[396] Z. Manna and A. Pnueli. Specification and verification of concurrent programs by forall-automata. In *Conference Record of the 14th Annual ACM Symposium on Principles of Programming Languages Munich, West Germany 21–23 January*, pages 1–12. ACM, 1987.

[397] Z. Manna and A. Shamir. The convergence of functions to fixedpoints of recursive definitions. *Theoretical Computer Science*, 6:109–141, 1978.

[398] Z. Manna and R. Waldinger. Synthesis: Dreams $\rightarrow$ programs. Technical Report STAN-CS-77-630, Department of Computer Science, Stanford University, November 1977.

[399] Z. Manna and R. J. Waldinger. Towards automatic program synthesis. In *[183]*, pages 270–310. 1971.

[400] Z. Manna and P. Wolper. Synthesis of communicating processes from temporal logic specifications. Technical Report STAN-CS-81-872, Department of Computer Science, Stanford University, September 1981.

[401] M. Marcotty, H. F. Ledgard, and G. V. Bochmann. A sampler of formal definitions. *ACM Computing Surveys*, 8:191–276, 1976.

[402] E. Marmier. *Automatic Verification of Pascal Programs*. PhD thesis, Swiss Federal Institute of Technology, Zurich, 1975.

[403] I. A. Mason. Hoare's logic in the LF. Technical Report ECS-LFCS-87-32, Laboratory for Foundations of Computer Science, Department of Computer Science, Edinburgh University, June 1987.

[404] B. H. Mayoh. Comparative semantics of programming languages. Technical Report DAIMI PB-173, Aarhus University, Denmark, April 1984.

[405] A. Mazurkiewicz. Iteratively computable relations. *Bulletin de L'Academie Polonaise des Sciences*, XX(9):793–798, 1972. Presented by A. Mostowski on May 6, 1972.

[406] A. Mazurkiewicz. Recursive algorithms and formal languages. *Bulletin de L'Academie Polonaise des Sciences*, XX(9):799–803, 1972. Presented by A. Mostowski on May 6, 1972.

[407] A. Mazurkiewicz. Concurrent program schemes and their interpretations. DAIMI report PB 78, Aarhus University, 1977.

[408] A. Mazurkiewicz. The semantics of concurrent systems: A modular fixed-point trace approach. 5th European Workshop on Applications and Theory of Petri Nets, 1984.

[409] A. Mazurkiewicz. Traces, histories, graphs: instances of a process monoid. In M. P. Chytil and V. Koubek, editors, *Mathematical Foundations of Computer Science 1984, Praha, Czechoslovakia, Proceedings.*, volume 176 of *Lecture Notes in Computer Science*, pages 115–133. Springer-Verlag, 1984.

[410] A. W. Mazurkiewicz. Proving algorithms by tail functions. *Information and Control*, 18(3), April 1971.

[411] J. McCarthy. Programs with common sense. In *Teddington Conference on the Mechanization of Thought Processes*, 1959.

[412] J. McCarthy. Recursive functions of symbolic expressions and their computation by machine, Part I. *Communications of the ACM*, 3:184–195, April 1960.

[413] J. McCarthy. 3-valued sentencial calculus. Manuscript., 1961.

[414] J. McCarthy. Computer programs for checking mathematical proofs. In *Proc. Symp. in Pure Mathematics, Vol. 5*, pages 219–227. American Mathematical Society, 1962.

[415] J. McCarthy. A basis for a mathematical theory for computation. In P. Braffort and D. Hirschberg, editors, *Computer Programming and Formal Systems*, pages 33–70. North-Holland Publishing Company, 1963. (A slightly extended and corrected version of a talk given at the May 1961 Western Joint Computer Conference).

[416] J. McCarthy. Towards a mathematical science of computation. In C. M. Popplewell, editor, *Information Processing'62*, pages 21–28. North-Holland, 1963.

[417] J. McCarthy. A proof-checker for predicate calculus. Stanford Artificial Intelligence Project Memo 27, Computer Science Department, Stanford University, March 1965.

[418] J. McCarthy. A formal description of a subset of ALGOL. In *[547]*, pages 1–12, 1966.

[419] J. McCarthy. The programming language *elephant*. Incomplete Draft., 1985.

[420] J. McCarthy and J. Painter. Correctness of a compiler for arithmetic expressions. Technical Report CS38, Computer Science Department, Stanford University, April 1966. See also pages 33–41 Proc. Symp. in Applied Mathematics, Vol.19: Mathematical Aspects of Computer Science, American Mathematical Society, 1967.

[421] C. L. McGowan. An inductive proof technique for interpreter equivalence. In *[512]*, pages 139–148. 1972.

[422] N. Metropolis, J. Howlett, and G.-C. Rota, editors. *A History of Computing in the Twentieth Century*. Academic Press, 1980.

[423] A. R. Meyer. What is a model of the lambda calculus? *Information and Control*, 52(1), January 1982. Special issue NSF Workshop on Recursion Theoretic Aspects of Computer Science, Purdue University, May 1981.

[424] B. Meyer. *Object-oriented Software Construction*. Prentice-Hall, 1988.

[425] H. D. Mills. Mathematical foundations for structured programming. Technical report, IBM Gaithersburg, February 1972.

[426] H. D. Mills. How to write correct programs and know it. *ACM Sigplan Notices*, 10(6):363–370, June 1975.

[427] R. Milne and C. Strachey. A theory of programming language semantics, 1974. An essay submitted for the Adams Prize 1973–74.

[428] R. Milne and C. Strachey. *A Theory of Programming Language Semantics*. Chapman and Hall, 1976. Part A: Indices and Appendices, Fundamental Concepts and Mathematical Foundations.

[429] R. Milne and C. Strachey. *A Theory of Programming Language Semantics*. Chapman and Hall, 1976. Part B: Standard Semantics, Store Semantics and Stack Semantics.

[430] R. Milner. The difficulty of verifying a program with unnatural data representation. Technical Report 3, Computation Services Dept., University College of Swansea, January 1969.

[431] R. Milner. Equivalences on program schemes. Memorandum 5, Department of Computer Science, University College of Swansea, March 1969.

[432] R. Milner. A formal notion of simulation between programs. Technical Report 14, Department of Computer Science, University College of Swansea, October 1970.

[433] R. Milner. An algebraic definition of simulation between programs. Technical Report CS-205, Computer Science Dept, Stanford University, February 1971.

[434] R. Milner. Program simulation: An extended formal notion. Technical Report 17, University College of Swansea, April 1971.

[435] R. Milner. Logic for computable functions description of a machine implementation. Technical Report STAN-CS-72-288, Computer Science Department, Stanford University, May 1972.

[436] R. Milner. An approach to the semantics of parallel programs. In *Proceedings of the Convegno di Informatica Teorica*, pages 285–302, 1973.

[437] R. Milner. *A Calculus for Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer-Verlag, 1980.

[438] R. Milner. On relating synchrony and asynchrony. Technical Report CSR-75-80, University of Edinburgh, Department of Computer Science, December 1980.

[439] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[440] J. Misra and K. M. Chandy. Proofs of networks of processes. *IEEE Transactions on Software Engineering*, 7:417–426, 1981.

[441] E. Moggi. *The Partial Lambda-Calculus*. PhD thesis, Department of Computer Science, University of Edinburgh, August 1988. Published as TR CST-53-88.

[442] B. Möller. Formal derivation of pointer algorithms. In M. Broy, editor, *Informatik und Mathematik*, pages 419–440. Springer-Verlag, 1991.

[443] B. Q. Monahan. *Data Type Proofs using Edinburgh LCF*. PhD thesis, University of Edinburgh, 1984. Published as TR CST-34-85.

[444] F. L. Morris. Criteria for semantical analysis, July 1970. Manuscript.

[445] F. L. Morris. The next 700 formal language descriptions. Manuscipt, 1970.

[446] F. L. Morris. *Correctness of Translations of Programming Languages – An Algebraic Approach*. PhD thesis, Computer Science Department, Stanford University, August 1972. Printed as STAN-CS-72-303.

[447] F. L. Morris. Advice on stucturing compilers and proving them correct. In *ACM Symposium on Principles of Programming Languages*, pages 144–152. ACM, 1973.

[448] F. L. Morris and C. B. Jones. An early program proof by Alan Turing. *Annals of the History of Computing*, 6:139–143, April 1984.

[449] J. H. Morris. Lambda-calculus models of programming languages. Technical Report MAC-TR-57 (Thesis), MIT, December 1968.

[450] J. H. Morris. A correctness proof of the Fisher-Galler algorithm using inductive assertions. In *[512]*, pages 107–124. 1972.

[451] J. H. Morris. Types are not sets. In *ACM Symposium on Principles of Programming Languages*, pages 120–124. ACM, October 1973.

[452] P. D. Mosses. The mathematical semantics of Algol 60. Technical Monograph PRG-12, Oxford University Computing Laboratory, Programming Research Group, January 1974.

[453] P. D. Mosses. *Mathematical Semantics and Compiler Generation*. PhD thesis, University of Oxford, April 1975.

[454] P. D. Mosses. SIS — semantics implementation system: Tested examples. Technical Report DAIMI MD-33, Aarhus University, Denmark, August 1979.

[455] P. D. Mosses. SIS - semantics implementation system: Reference manual and user guide. Technical Report DAIMI MD-30, Aarhus University, Denmark, August 1979.

[456] P. D. Mosses. A constructive approach to compiler correctness. In *[301]*, pages 189–210. Springer-Verlag, 1980.

[457] P. D. Mosses. *Action Semantics*. Cambridge Tracts in Theoretical Computer Science, 26. Cambridge University Press, 1992.

[458] B. Moszkowski. *Executing Temporal Logic Programs*. Cambridge University Press, 1986.

[459] R. Nakajima and T. Yuasa. *The IOTA Programming System: A Modular Programming Environment*, volume 160 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1983.

[460] P. Naur. Proof of algorithms by general snapshots. *BIT*, 6:310–316, 1966.

[461] P. Naur. Programming by action clusters. *BIT*, 9:250–258, 1969.

[462] P. Naur. An experiment on program development. *BIT*, 12:347–365, 1972.

[463] P. Naur. *Concise Survey of Computer Methods*. Studentlitteratur, Lund, 1974.

[464] P. Naur. Formalization in program development. *BIT*, 22:437–453, 1982.

[465] P. Naur and B. Randell, editors. *Software Engineering*. NATO Science Committee, 1969. Report on a conference Garmisch, Germany, 7th to 11th October 1968.

[466] E. J. Neuhold and G. Chroust. *Formal Models in Programming*. North-Holland, 1985. Proceedings of the IFIP TC2 Working Conference on The Role of Abstract Models in Information Processing. Vienna, Austria, 30 January – 1 February 1985.

[467] A. Newell and H. A. Simon. The logic theory machine. In *IRE Transactions on Information Theory IT-2*, pages 61–79, 1956.

[468] M. Nielsen, G. Plotkin, and G. Winskel. Petri nets, event structures and domains. In *[305]*, pages 266–284. Springer-Verlag, 1979.

[469] T. Nipkow. *Behavioural Implementation Concepts for Nondeterministic Data Types*. PhD thesis, University of Manchester, 1986. Reprinted as UMCS-87-5-3, May 1987.

[470] T. Nipkow. Non-deterministic data types: Models and implementations. *Acta Informatica*, 22:629–661, 1986.

[471] E. R. Olderog and C. A. R. Hoare. Specification-oriented semantics for communicating processes. *Acta Informatica*, 23:9–66, 1986.

[472] A. Ollongren. On the Vienna method for the definition of programming languages. Technical Report TR 25.120, IBM Laboratory Vienna, May 1971.

[473] S. S. Owicki. *Axiomatic Proof Techniques for Parallel Programs*. PhD thesis, Department of Computer Science, Cornell University, 1975. Published as technical report 75-251.

[474] S. S. Owicki and D. Gries. An axiomatic proof technique for parallel programs I. *Acta Informatica*, 6:319–340, 1976.

[475] J. Owlett. *A Theory of Database Schemata – Studies in Conceptual and Relational Schemata*. PhD thesis, Wolfson College, Oxford University, October 1979.

[476] J. A. Painter. Semantic correctness of a compiler for an Algol-like language. Technical Report AI Memo 44, Computer Science Department, Stanford University, March 1967.

[477] D. Park. Some semantics for data structures. In D. Michie, editor, *Machine Intelligence, 3*, pages 351–371. Edinburgh University Press, 1968.

[478] D. Park. Fixpoint induction and proofs of program properties. In B. Meltzer and D. Michie, editors, *Machine Intelligence, 5*, pages 59–78. Edinburgh University Press, 1969.

[479] D. Park. Notes on a formalism for reasoning about schemas, November 1970. Manuscript.

[480] D. Park. A theorem on loops, using Scott's induction principle, November 1970. Manuscript.

[481] D. Park. The Y-combinator in Scott's lambda-calculus model. Symposium on Theory of Programming, University of Warwick. Unpublished, 1970.

[482] D. Park. On the semantics of fair parallelism. In *[50]*, pages 504–526. 1980.

[483] D. Park. Concurrency and automata on infinite sequences. In *Theoretical Computer Science, 5th GI-Conference, Karlsruhe, March, 1981*, number 104 in Lecture Notes in Computer Science, pages 167–183. Springer-Verlag, 1981.

[484] M. S. Paterson. *Equivalence Problems in a Model of Computation*. PhD thesis, University of Cambridge, 1967.

[485] M. S. Paterson. Program schemata. In D. Michie, editor, *Machine Intelligence, 3*, pages 19–31. Edinburgh University Press, 1968.

[486] L. C. Paulson. *Logic and Computation: Interactive Proof with Cambridge LCF*. Cambridge University Press, 1987.

[487] J. S. Pedersen. VDM in three generations of Ada formal descriptions. In *[49]*, pages 33–48, 1987.

[488] C. A. Petri. *Kommunikation mit Automaten*. PhD thesis, University of Darmstadt, 1962.

[489] C. A. Petri. Nichtsequentielle prozesse. Technical Report ISF-76-6, GMD, Bonn, 1976.

[490] C. A. Petri. Non-sequential processes. Technical Report ISF-77-05, GMD, Bonn, 1977. Translation of ISF-76-6.

[491] G. D. Plotkin. A powerdomain construction. *SIAM Journal on Computing*, 5:452–487, September 1976.

[492] G. D. Plotkin. Dijkstra's predicate transformers and Smyth's power domains. In *[50]*, pages 527–553. 1980.

[493] G. D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, Aarhus University, 1981.

[494] G. D. Plotkin. An operational semantics for CSP. Technical Report CSR-114-82, Department of Computer Science, University of Edinburgh, May 1982.

[495] A. Pnueli. The temporal logic of programs. In *Proceedings of Eighteenth Foundations of Computer Science*, pages 46–57, 1977.

[496] A. Pnueli. The temporal semantics of concurrent programs, 1977. Tel-Aviv University.

[497] A. Pnueli. The temporal semantics of concurrent programs. In *[305]*, pages 1–20. Springer-Verlag, 1979.

[498] A. Pnueli. The temporal semantics of concurrent programs. *Theoretical Computer Science*, 13:45–60, 1981.

[499] W. Polak. *Compiler Specification and Verification*, volume 124 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1981.

[500] D. Prawitz. *Natural Deduction: A Proof-Theoretical Study*. Almqvist and Wisell, Stockholm, 1965.

[501] B. Randell. *The Origins of Digital Computers: Selected Papers*. Springer-Verlag, second edition, 1975.

[502] J. C. Reynolds. Definitional interpreters for higher-order programming languages. In *Proceedings of the 1972 ACM Annual Conference*, pages 717–740, New York, 1972. ACM.

[503] J. C. Reynolds. On the relation between direct and continuation semantics. In *Proceedings of the 2nd Colloqium on Automata, Languages and Programming, Saarbrucken, July 29–Aug2, 1974*, volume 14 of *Lecture Notes in Computer Science*, pages 141–156. Springer-Verlag, 1974.

[504] J. C. Reynolds. *The Craft of Programming*. Prentice Hall International, 1981.

[505] J. C. Reynolds. Idealized Algol and its specification logic. Technical Report 1-81, School of Computer and Information Science, Syracuse University, July 1981.

[506] J. A. Robinson. Theorem-proving on a computer. *Journal of the ACM*, 10:163–174, 1963.

[507] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12:23–41, 1965.

[508] J. A. Robinson. Logic and logic programming. *Communications of the ACM*, 35(3):40–65, March 1992.

[509] A. W. Roscoe and C. A. R. Hoare. Laws of occam programming. Technical Report PRG-53, Oxford University Computing Laboratory, Programming Research Group, 1986.

[510] G. Rozenberg. *Advances in Petri-nets*, volume 188 of *Lecture Notes in Computer Science*. Springer-Verlag, 1985.

[511] B. Russell. On an equivalence between continuation and stack semantics. *Acta Informatica*, 8:113–123, 1977.

[512] R. Rustin. *Formal Semantics of Programming Languages*. Prentice-Hall, 1972. Courant Computer Science Symposium 2, September 14-16, 1970.

[513] J. D. Rutledge. On Ianov's program schemata. *Journal of the ACM*, 11:1–9, January 1964.

[514] J. G. Sanderson. The lambda calculus, lattice theory and reflexive domains, 1973. Mathematical Institute Lecture Notes, University of Oxford.

[515] D. T. Sannella. *Semantics, Implementation and Pragmatics of Clear, A program Specification Language*. PhD thesis, Department of Computer Science, University of Edinburgh, July 1982.

[516] R. W. Scheifler. A donotational semantics of CLU. Technical Report MIT/LCS/TR-201, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Mass., May 1978.

[517] D. A. Schmidt. Denotational semantics as a programming language. Technical Report CSR-100-82, Department of Computer Science, University of Edinburgh, January 1982.

[518] D. A. Schmidt. *Denotational Semantics: a Methodology for Language Development*. Allyn & Bacon, 1986.

[519] C. S. Scholten. An axiomatic basis for non-deterministic programs. Technical Report CD.R/73/200/172, MIG Data Systems, November 1973.

[520] J. S. Schwartz. *Sematics of Partial Correctness Formalisms*. PhD thesis, Syracuse University, December 1974.

[521] J. S. Schwartz. Denotational semantics of parallelism. In *[305]*, pages 191–202. Springer-Verlag, 1979.

[522] J. T. Schwartz. Semantic definition methods and the evolution of programming languages. In *[512]*, pages 1–24. 1972.

[523] D. Scott. Existence and description in formal logic. In R. Schoenman, editor, *Bertrand Russell, Philosopher of the Century*, pages 181–200. Allen and Unwin, 1967.

[524] D. Scott. Some definitional suggestions for automata theory. *Journal of Computer and System Sciences*, 1(2):187–212, August 1967.

[525] D. Scott. A construction of a model for the λ calculus. Manuscript, November 1969.

[526] D. Scott. Models for the λ calculus. Manuscript – Draft, December 1969.

[527] D. Scott. A type-theoretical alternative to CUCH, ISWIM, OWHY. Typed script – Oxford, October 1969.

[528] D. Scott. The lattice of flow diagrams. Technical Report PRG-3, Oxford University Computing Laboratory, Programming Research Group, November 1970.

[529] D. Scott. Outline of a mathematical theory of computation. Technical Report PRG-2, Oxford University Computing Laboratory, Programming Research Group, November 1970.

[530] D. Scott. Continuous lattices. Technical Report PRG-7, Oxford University Computing Laboratory, Programming Research Group, August 1971.

[531] D. Scott. The lattice of flow diagrams. In *[183]*, pages 311–366. Springer-Verlag, 1971.

[532] D. Scott. Lattice theory, data types and semantics. In *[512]*, pages 65–106. 1972.

[533] D. Scott. Models for various type-free calculi. In P. Suppes, L. Henkin, A. Joja, and Gr.C. Moisil, editors, *Studies in Logic and Foundations of Mathematics Vol. 74 (Proc. of the 4th International Congress for Logic, Methodology and Philosophy of Science, Bucharest, 1971)*, pages 158–187. North Holland Publishing Company, 1973.

[534] D. Scott. A simplified construction for λ calculus models. Manuscript, April 1973.

[535] D. Scott. Data types as lattices. Technical Report PRG-5, Oxford University Programming Research Group, September 1976. Reprinted from the SIAM Journal on Computing, Volume 5, 1976, pp. 522–587; manuscript version dated 1972.

[536] D. Scott. Lectures on a mathematical theory of computation. Technical Report PRG 19, Oxford University Computing Lab, Programming Research Group, May 1981.

[537] D. Scott. Lectures on a mathematical theory of computation. In *[83]*, pages 145–292. 1982.

[538] D. S. Scott. Identity and existence in intuitionistic logic. In M. P. Fourman, C. J. Mulvey, and D. S. Scott, editors, *Applications of Sheaves. Proceedings, Durham 1979*, volume 753 of *Lecture Notes in Mathematics*, pages 660–696. Springer-Verlag, 1979.

[539] R. Sethi and A. Tang. Constructing call-by-value continuation semantics. *Journal of the ACM*, 27,3:580–597, 1980.

[540] H. A. Simon. *Models of my life*. Basic Books, 1992.

[541] R. L. Sites. *Proving that Computer Programs Terminate Cleanly*. PhD thesis, Computer Science Department, Stanford University, 1974. Printed as STAN-CS-74-418.

[542] R. L. Sites. Some thoughts on proving clean termination of programs. Technical Report STAN-CS-74-417, Computer Science Department, Stanford University, May 1974.

[543] M. B. Smyth. Powerdomains. Technical report, Department of Computer Science, University of Warwick, May 1976.

[544] M. B. Smyth and G. D. Plotkin. The category-theoretic solution of recursive domain equations. Technical Report CSR-102-82, Department of Computer Science, University of Edinburgh, February 1982.

[545] S. Sokołowski. Partial correctness: The term-wise approach. *Science of Computer Programming*, 4:141–157, 1984.

[546] N. Soundararajan. A proof technique for parallel programs. *Theoretical Computer Science*, 31:13–29, 1984.

[547] T. B. Steel. *Formal Language Description Languages for Computer Programming*. North-Holland, 1966.

[548] C. Stirling. A compositional reformulation of Owicki-Gries's partial correctness logic for a concurrent while language. In L. Kott, editor, *Automata, Languages and Programming: ICALP'86*, volume 226 of *Lecture Notes in Computer Science*, pages 407–415. Springer-Verlag, 1986.

[549] C. Stirling. A generalisation of Owicki-Gries's Hoare logic for a concurrent while language. *Theoretical Computer Science*, 58:347–359, 1988.

[550] K. Stølen. *Development of Parallel Programs on Shared Data-Structures*. PhD thesis, Manchester University, 1990. Also published as technical report UMCS-91-1-1.

[551] J. E. Stoy. The Scott-Strachey approach to the mathematical semantics of programming languages. Course notes of MIT subject 6.971, Fall 1973. Limited preliminary edition. From MIT Project MAC., December 1974.

[552] J. E. Stoy. *Denotational Semantics: The Scott-Strachey Approach to Programming Language Theory*. MIT Press, 1977.

[553] J. E. Stoy. Foundations of denotational semantics. In *[50]*, pages 43–99. 1980.

[554] J. E. Stoy. The congruence of two programming language definitions. *Theoretical Computer Science*, 13:151–174, 1981.

[555] J. E. Stoy. Semantic models. In M. Broy and G. Schmidt, editors, *Theoretical Foundations of Programming Methodology*, pages 293–328. D. Reidel, 1982.

[556] J. E. Stoy. Some mathematical aspects of fuctional programming. In *Functional Programming and its Applications : CREST-ITG–Advanced course proceedings*, pages 217–252. Cambridge University Press, 1982.

[557] J. E. Stoy. The development of programming language semantics. In *[466]*, pages 229–230, 1985.

[558] J. E. Stoy and C. Strachey. OS6 – an operating system for a small computer. Technical Report PRG-8, Oxford University Computing Laboratory, Programming Research Group, May 1972.

[559] C. Strachey. Systems analysis and programming. *Scientific American*, 215(3):112–124, September 1966.

[560] C. Strachey. The varieties of programming language. Technical Monograph PRG-10, Oxford University Computing Lab, March 1973.

[561] C. Strachey. How can we put programming theory into practice? In *Software '73. Loughborough, July 9–11 1973*, pages 115–122. Transcripta Books, 1974.

[562] C. Strachey and D. Scott. Mathematical semantics for two simple languages, August 1970. Paper read at Princeton.

[563] C. Strachey and J. E. Stoy. The text of OSPub. Technical Report PRG-9(t), Oxford University Computing Laboratory, Programming Research Group, July 1972.

[564] C. Strachey and J. E. Stoy. The text of OSPub, (Commentary). Technical Report PRG-9(c), Oxford University Computing Laboratory, Programming Research Group, July 1972.

[565] C. Strachey and C. P. Wadsworth. Continuations – a mathematical semantics for handling jumps. Monograph PRG-11, Oxford University Computing Laboratory, Programming Research Group, January 1974.

[566] H. R. Strong. Algebraically generalized recursive function theory. *IBM Journal of Research and Development*, pages 465–475, November 1968.

[567] N. Suzuki. Automatic program verification ii: Verifying programs by algebraic and logical reduction. Technical Report STAN-CS-74-473, Computer Science Department, Stanford University, December 1974.

[568] A. Tarlecki. A language of specified programs. *Science of Computer Programming*, 5:59–81, 1985.

[569] A. H. Taub, editor. *John von Neumann: Collected Works*, volume V: Design of Computers, Theory of Automata and Numerical Analysis. Pergamon Press, 1963.

[570] R. D. Tennent. The denotational semantics of programming languages. *Communications of the ACM*, 19:437–453, 1976.

[571] R. D. Tennent. *Principles of Programming Languages*. Prentice-Hall International, 1981.

[572] R. D. Tennent. Semantic analysis of specification logic. Technical Report ECS-LFCS-86-5, LFCS, Department of Computer Science, University of Edinburgh, June 1986.

[573] J. W. Thatcher, E. G. Wagner, and J. B. Wright. More on advice on structuring compilers and proving them correct. Technical Report RC-7588, IBM Research Division, New York, April 1979.

[574] J. W. Thatcher, E. G. Wagner, and J. B. Wright. More advice on structuring compilers and proving them correct. In *[301]*, pages 165–188. Springer-Verlag, 1980.

[575] B. A. Trakhtenbrot, J. Y. Halpern, and A. R. Meyer. From denotational to operational and axiomatic semantics for Algol-like langauges: An overview. Technical Report RJ 4105, IBM Research Division, New York, November 1983.

[576] A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, Series 2*, 42:230–265, 1936. Correction published: ibid, 43:544–546, 1937.

[577] A. M. Turing. Checking a large routine. In *Report of a Conference on High Speed Automatic Calculating Machines*, pages 67–69. University Mathematical Laboratory, Cambridge, June 1949.

[578] W. M. Turski. ALGOL 68 revisited twelve years later or from AAD to ADA. In J. W. de Bakker and J. C. van Vliet, editors, *Algorithmic Languages*, pages 417–431. IFIP, North-Holland, 1981.

[579] M. H. van Emden and R. A. Kowalski. The semantics of predicate logic as a programming language. *Journal of the ACM*, 23:733–742, 1976.

[580] A. van Wijngaarden. Numerical analysis as an independent science. *BIT*, 6:66–81, 1966. (Text of 1964 talk).

[581] A. van Wijngaarden, M. Sintzoff, B. J. Mailloux, C. H. Lindsey, J. E. L. Peck, L. G. L. T. Meertens, C. H. A. Koster, and R. G. Fisker. *Revised report on the Algorithmic Language ALGOL 68*. Mathematical Centre Tracts 50. Mathematisch Centrum, Amsterdam, 1976.

[582] A. van Wijngaarden B. J. Mailloux, J. E. L. Peck, and C. H. A. Koster. *Report on the Algorithmic Language ALGOL 68*. Mathematisch Centrum, Amsterdam, October 1969. Second printing , MR 101.

[583] C. P. Wadsworth. *Semantics and Pragmatics of the Lambda-Calculus*. PhD thesis, Programming Research Group, University of Oxford, September 1971.

[584] C. P. Wadsworth. The relation between computational and denotational properties for Scott's D_∞-models of the lambda-calculus. *SIAM Journal on Computing*, 5(3):488–521, September 1976.

[585] C. P. Wadsworth. Approximate reduction and lambda-calculus models. *SIAM Journal on Computing*, 7(3):337–356, August 1978.

[586] R. J. Waldinger. *Constructing Programs Automatically using Theorem Proving*. PhD thesis, Carnegie-Mellon University, 1969.

[587] M. Wand. Specifications, models and implementations of data abstractions. Technical Report 88, Indiana University, Computer Science Department, March 1980.

[588] M. Wand. Deriving target code as a representation of continuation semantics. *ACM Transactions on Programming Languages and Systems*, 4:496–517, July 1982.

[589] H. Wang. Towards mechanical mathematics. *IBM Journal of Research and Development*, 4:2–22, 1960.

[590] P. Wegner. Programming language semantics. In *[512]*, pages 149–248. 1972.

[591] R. L. Wexelblat, editor. *History of Programming Languages*. Academic Press, 1981.

[592] M. V. Wilkes. Constraint-type statements in programming languages. *Communications of the ACM*, 7:587–588, October 1964.

[593] M. V. Wilkes. *Memoirs of a Computer Pioneer*. MIT Press, 1985.

[594] W. T. Wilner. Formal semantics definition using sythesized and inherited attributes. In *[512]*, pages 25–40. 1972.

[595] J. Winkowski. Behaviours of concurrent systems. *Theoretical Computer Science*, 11:39–60, September 1980.

[596] N. Wirth. On certain basic concepts of programming languages. Technical Report CS 65, Computer Science Department, Stanford University, May 1967.

[597] N. Wirth. The design of a pascal compiler. *Software — Practice and Experience*, 1:309–333, 1971.

[598] N. Wirth. *Systematic Programming: An Introduction*. Prentice-Hall, 1973.

[599] N. Wirth. *Algorithms + Data Structures = Programs*. Prentice-Hall, 1976.

[600] Y I. Yanov. The logical schemes of algorithms. *Problems of Cybernetics*, 1:82–140, 1960.

[601] H. Zemanek. Semiotics and programming languages. *Communications of the ACM*, 9:139–143, 1966.

[602] H. Zemanek. Formalization, history, present and future, 1974. Paper for Newcastle IBM Seminar.

[603] H. Zemanek. Formal definition: The hard way. In *[466]*, pages 411–418, 1985.

[604] S. N. Zilles. Abstract specifications for data types. Technical Report 11, M.I.T. Progress Report, 1974.

[605] K. Zuse. *Der Computer: Mein Lebenswerk*. Springer-Verlag, 1984.