# Bibliography


[1] F. B. Abreu and R. Carapuca. Candidate metrics for object-oriented software wihin a taxonomy framework. *Journal of Systems and Software*, 26(1):87–96, Jan. 1994.

[2] V. R. Basili. *Tutorial on models and metrics for software management and engineering*. IEEE Press, 1980.

[3] V. R. Basili, L. Briand, and W. L. Melo. A validation of object-oriented design metrics as quality indicators. *IEEE Transactions on Software Engineering*, 22(10):751–761, Oct. 1996.

[4] V. R. Basili and A. Turner. Iterative enhancement, a practical technique for software development. *IEEE Transactions on Software Engineering*, SE-1(4), Dec. 1975.

[5] V. R. Basili and D. M. Weiss. Evaluation of a software requirements document by analysis of change data. In *5th Int. Conf. on Software Engineering*, pages 314–323. IEEE, 1981.

[6] L. Bass, P. Clements, and Rick Kazman. *Software Architecture in Practice, Second Edition*. Addison-Wesley Professional, 2003.

[7] K. Beck. *Extreme Programming Explained*. Addison-Wesley, 2000.

[8] K. Beck. *Test Driven Development: by Example*. Addison-Wesley Professional, 2002.

[9] R.V. Binder. *Testing Object-Oriented Systems—Model, Patterns, and Tools*. Addison-Wesley, 1999.

[10] B. Boehm. Software engineering. *IEEE Transactions on Computers*, 25(12), Dec. 1976.

[11] B. Boehm. *Tutorial: software risk management*. IEEE Computer Society, 1989.

[12] B. W. Boehm. *Software Engineering Economics*. Prentice Hall, Englewood Cliffs, NJ, 1981.

[13] B. W. Boehm. Software engineering economics. *IEEE Transactions on Software Engineering*, 10(1):135–152, Jan. 1984.

[14] B. W. Boehm. Improving software productivity. *IEEE Computer*, pages 43–57, Sept. 1987.

[15] G. Booch. *Object-Oriented Analysis and Design*. The Benjamin/Cummings Publishing Company, 1994.

[16] F. Brooks. *The Mytical Man Month*. Addison-Wesley, 1975.

[17] N. Brown. Industrial-strength management strategies. *IEEE Software*, July 1996.

[18] R.N. Charette. *Software Engineering Risk Analysis and Management*. McGraw Hill, 1989.

[19] R.N. Charette. Large-scale project management is risk management. *IEEE Software*, July 1996.

[20] E. Chen. Program complexity and programmer productivity. *IEEE Transactions on Software Engineering*, SE-4:187–194, May 1978.

[21] S. R. Chidamber and C. F. Kemerer. A metrics suite for object-oriented design. *IEEE Transactions on Software Engineering*, 20(6):476–493, June 1994.

[22] T.S. Chow. Testing software design modeled by finite state machines. *IEEE Transactions on Software Engineering*, SE-4(3):178–187, 1978.

[23] P. Clements, F. Bachmann, L. Bass, D. Garlan, J. Ivers, R. Little, R. Nord, and J. Stafford. *Documenting Software Architectures: Views and Beyond*. Addison-Wesley, 2003.

[24] A. Cockburn. *Writing Effective Use Cases*. Addison-Wesley, 2001.

[25] D.M. Cohen, S.R. Dalal, M.L. Fredman, and G.C. Patton. The AETG system: An approach to testing based on combinatorial design. *IEEE Transactions on Software Engineering*, 23(7):437–443, 1997.

[26] S. D. Conte, H. E. Dunsmore, and V. Y. Shen. *Software Engineering Metrics and Models*. The Benjamin/Cummings Publishing Company, 1986.

[27] J. S. Davis. Identification of errors in software requirements through use of automated requirements tools. *Information and Software Technology*, 31(9):472–476, Nov. 1989.

[28] T. DeMarco. *Structured Analysis and System Specification*. Yourdon Press, 1979.

[29] L. Dobrica and E. Niemela. A survey on software architecture analysis methods. *IEEE Transactions on Software Engineering*, 28(7):638–653, 2002.

[30] J. Eder, G. Kappel, and M. Schrefl. Coupling and cohesion in object-oriented systems. Technical report, University of Klagenfurt, 1994.

[31] M. E. Fagan. Design and code inspections to reduce errors in program development. *IBM System Journal*, (3):182–211, 1976.

[32] M. E. Fagan. Advances in software inspections. *IEEE Transactions on Software Engineering*, 12(7):744–751, July 1986.

[33] W. Farr. Software reliability modeling survey. In M. R. Lyu, editor, *Software Reliability Engineering*, pages 71–117. McGraw Hill and IEEE Computer Society, 1996.

[34] S. I. Feldman. Make—a program for maintaining computer programs. *Software Practice and Experience*, 9(3):255–265, March 1979.

[35] M. Fowler. *UML Distilled—A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley Professional, 2003.

[36] M. Fowler, K. Beck, J. Brant, W. Opdyke, and D. Roberts. *Refactoring: Improving the Design of Existing Code*. Addison-Wesley, 1999.

[37] D. P. Freedman and G. M. Weinberg. *Handbook of Walkthroughs, Inspections, and Technical Reviews—Evaluating Programs, Projects, and Products*. Dorset House, 1990.

[38] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns—Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1995.

[39] T. Gilb and D. Graham. *Software Inspection*. Addison-Wesley, 1993.

[40] H. Gomma and D. B. H. Scott. Prototyping as a tool in the specification of user requirements. In *Fifth Int. Conf. on Software Engineering*, pages 333–341, 1981.

[41] J. Goodenough and S. L. Gerhart. Towards a theory of test data selection. *IEEE Transactions on Software Engineering*, SE-1:156–173, 1975.

[42] S. E. Goodman and S. T. Hedetniemi. *Introduction to the Design and Analysis of Algorithms*. McGraw-Hill, 1977.

[43] R. Grady and D. Caswell. *Software Metrics: Establishing a Company-wide Program*. Prentice Hall, 1987.

[44] R. B. Grady and T. V. Slack. Key lessons learned in achieving widespread inspection use. *IEEE Software*, pages 48–57, July 1994.

[45] E.M. Hall. *Managing Risk: Methods for Software Development and Enhancement*. Addison-Wesley, 1998.

[46] M. Halstead. *Elements of Software Science*. Elsevier North-Holland, 1977.

[47] W. Harrison, K. Magel, R. Kluczny, and A. DeKock. Applying software complexity metrics to program maintenance. *IEEE Computer*, pages 65–79, Sept. 1982.

[48] S. Henry and D. Kafura. Software structure metrics based on information flow. *IEEE Transactions on Software Engineering*, 7(5):510–518, 1981.

[49] S. Henry and D. Kafura. The evaluation of software systems' structures using quantitative software metrics. *Software Practice and Experience*, 14(6):561–573, June 1984.

[50] C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(3):335–355, 1969.

[51] IBM-Rational. Rational unified process best practices for software development teams. Technical report, IBMwebsite, 1993.

[52] IEEE. IEEE standard glossary of software engineering terminology. Technical report, 1990.

[53] IEEE. IEEE recommended practice for software requirements specifications. Technical report, 1998.

[54] IEEE. IEEE recommended practice for architectural description of software-intensive systems. Technical Report 1471-2000, 2000.

[55] International Standards Organization. Software engineering—product quality. part 1: Quality model. Technical Report ISO9126-1, 2001.

[56] I. Jacobson. *Object-oriented Software Engineering—A Use Case Driven Approach*. Addison-Wesley, 1992.

[57] P. Jalote. *CMM in Practice—Processes for Executing Software Projects at Infosys*. Addison-Wesley, 1999.

[58] P. Jalote. *Software Project Management in Practice*. Addison-Wesley, 2002.

[59] P. Jalote, A. Palit, and P. Kurien. The timeboxing process model for iterative software development. In *Advances in Computers, Vol. 62*, pages 67–103. Academic Press, 2004.

[60] P. Jalote, A. Palit, P. Kurien, and V. T. Peethamber. Timeboxing: A process model for iterative software development. *The Journal of Systems and Software*, 70:117–127.

[61] S.H. Kan. *Metrics and Models in Software Quality Engineering*. Addison-Wesley, 1995.

[62] T. Korson and J. D. Gregor. Understanding object-oriented: A unifying paradigm. *Communications of the ACM*, 33(9):40–60, Sept. 1990.

[63] P. Kruchten. *The Rational Unified Process*. Addison-Wesley, 1999.

[64] W. Lie and S. Henry. Object-oriented metrics that predict maintainability. *Journal of Systems and Software*, 23(2):111–122, 1993.

[65] B. Liskov. Data abstraction and hierarchy. *SIGPLAN Notices*, 23(5), May 1988.

[66] B. Meyer. *Object Oriented Software Construction*. Prentice Hall, 1988.

[67] J. D. Musa, A. Iannino, and K. Okumoto. *Software Reliability—Measurement, Prediction, Application*. McGraw Hill, 1987.

[68] G. Myers. *The Art of Software Testing*. Wiley-Interscience, New York, 1979.

[69] J. Offutt, S. Liu, A. Abdurazik, and P. Ammann. Generating test data from state-based specifications. *The Journal of Software Testing, Verification, and Reliability*, 13(1):25–53, March 2003.

[70] M.S. Phadke. Planning efficient software tests. *Crosstalk*, Oct 1997.

[71] L. H. Putnam. A general empirical solution to the macro software sizing and estimation problem. *IEEE Transactions on Software Engineering*, SE-4:345–361, July 1978.

[72] L. H. Putnam and W. Myers. *Industrial Strength Software: Effective Management Using Measurement*. IEEE Computer Society, 1997.

[73] S. Rapps and E. J. Weyuker. Selecting software test data using data flow information. *IEEE Transactions on Software Engineering*, 11(4):367–375, Apr. 1985.

[74] W. W. Royce. Managing the development of large software systems. In *Proc. 9th Int. Conf. on Software Engineering (ICSE-9); originally in IEEE Wescon, Aug 1970*, pages 328–338. IEEE, 1987.

[75] SEI (Software Engineering Institute). *The Capability Maturity Model: Guidelines for Improving the Software Process*. Addison-Wesley, 1995.

[76] M. Shaw and D. Garlan. *Software Architecture: Perspectives on an Emerging Discipline*. Prentice Hall, 1996.

[77] M. D. Smith and D. J. Robson. Object oriented programming: The problems of validation. *Proc. of 6th Int. IEEE Conference on Software Maintenance*, pages 272–282, Nov. 1990.

[78] M. D. Smith and D. J. Robson. A framework for testing object-oriented programs. *Journal of Object-Oriented Programming*, pages 45–53, June 1992.

[79] W. P. Stevens, G. J. Myers, and L. Constantine. Structured design. *IBM Systems Journal*, 13(2), 1974.

[80] K. S. Trivedi. *Probability and Statistics with Reliability, Queuing, and Computer Science Applications, Second Edition*. Wiley-Interscience, 2002.

[81] C. Watson and C. Felix. A method of programming measurement and estimation. *IBM Systems Journal*, 16(1), Jan. 1977.

[82] G. M. Weinberg and E. L. Schulman. Goals and performance in computer programming. *Human Factors*, 16(1):70–77, 1974.

[83] E. F. Weller. Lessons learned from three years of inspection data. *IEEE Software*, pages 38–53, Sept. 1993.

[84] N. Wirth. Program development by stepwise refinement. *Communications of the ACM*, 14(4):221–227, April 1971.

[85] M. Woodward, M. Hennell, and D. Hedley. A measure of control flow complexity in program text. *IEEE Transactions on Software Engineering*, SE-5:45–50, Jan. 1979.

[86] R. T. Yeh and P. Zave. Specifying software requirements. *Proceedings of the IEEE*, 68(9):1077–1088, Sept. 1980.

[87] B. H. Yin and J. W. Winchester. The establishment and use of measures to evaluate the quality of designs. *Software Engineering Notes*, 3:45–52, 1978.

[88] E. Yourdon and L. Constantine. *Structured Design*. Prentice Hall, 1979.

[89] W. M. Zage and D. M. Zage. Evaluating design metrics on large-scale software. *IEEE Software*, pages 75–81, July 1993.

# Index