References

1. JUnit - Testing Resource for eXtreme Programming. `http://www.junit.org`. Access date April 5th, 2004.
2. *IEEE Computer*, volume 32. July 1999.
3. J. Aagedal and E. Ecklund. Modelling QoS: Towards a UML profile. In *Proceedings of UML 2002*, Dresden, Germany, Oct. 2002.
4. A. Abdurazik and J. Offutt. Using UML collaboration diagrams for static checking and test generation. In *UML'00*, pages 383–395, York, UK, Oct. 2000.
5. Exception handling for a 21st century programming language proceedings. *ACM SIGAda Ada Letters*, XXI(3), 2001.
6. H. Agrawal, J. R. Horgan, E. W. Krauser, and S. A. London. Incremental regression testing. In *Proceedings of the Conference on Software Maintenance*, pages 348–357, Washington, Sept. 1993.
7. J. Aidemark, J. Vinter, P. Folkesson, and J. Karlsson. GOOFI: Generic object-oriented fault injection tool. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN 2001)*, Gothenburg, Sweden, 2001.
8. R. Alexander and J. Offutt. Criteria for testing polymorphic relationships. In *Int. Symp. on Softw. Reliability Eng.*, pages 15–23, 2000.
9. R. Allen and D. Garlan. A formal basis for architectural connection. *ACM Trans. on Software Engineering and Methodology*, 6(3), April 1997.
10. P. America, H. Obbink, R. van Ommering, and F. van der Linden. Copam: A component-oriented platform architecting method family for product family engineering. In *Proceedings of the First Software Product Lines Conference, Software Product Lines, Experience and Research Directions*, pages 167 – 180, Boston, 2000. Kluwer Academic Publishers.
11. Apple Computer. *MacAppII Programmer's Guide*. Apple Computer, 1989.
12. J. Arlat, M. Aguera, L. Amat, Y. Crouzet, J.-C. Fabre, J.-C. Laprie, E. Martin, and D. Powell. Fault injection for dependability validation: A methodology and some applications. *IEEE Transactions on Software Engineering*, 16(2):166–182, 1990.
13. C. Atkinson and H.-G. Groß. Built-in contract testing in model-driven, component-based development. In *Proceedings of Workshop on Component-Based Development Processes*, April 2002.

14. F. Bachman, L. Bass, C. Buhman, S. Cornella-Dorda, F. Long, J. Robert, R. Seacord, and K. Wallnau. Volume II: Technical concepts of component-based software engineering. Technical Report CMU/SEI-2000-TR-008, Carnegie Mellon University, Software Engineering Institute, 2000.
15. S. Balsamo, P. Inverardi, and C. Mangano. An approach to performance evaluation of software architectures. In *Proceedings of the First International Workshop on Software and Performance*, pages 178–190, 1998.
16. A. Baratloo, N. Singh, and T. Tsai. Transparent run-time defense against stack smashing attacks. In *Proceedings of USENIX Annual Technical Conference*, June 2000.
17. F. Barbier. Composability for software components: An approach based on the whole-part theory. In *Proceedings of The 8th IEEE International Conference on Engineering of Complex Computer Systems, Greenbelt, USA, IEEE*, pages 101–106. Computer Society Press, 2002.
18. F. Barbier, N. Belloir, and J.-M. Bruel. Incorporation of test functionality into software components. In *Proceedings of The 2nd International Conference on COTS-Based Software Systems, Ottawa, Canada, Lecture Notes in Computer Science 2580*, pages 25–35. Springer, 2003.
19. M. Barnett, W. Grieskamp, W. Schulte, N. Tillmann, and M. Veanes. Validating use-cases with the asml test tool. In *QSIC'03*, pages 238–246, Dallas, USA, Oct. 2003.
20. V. R. Basili and B. Boehm. COTS-based systems top 10 list. *IEEE Computer*, 34(5):91–93, 2001.
21. L. Bass, C. Buhman, S. Comella-Dorda, F. Long, J. Robert, R. Seacord, and W. K. Volume i: Market assessment of component-based software engineering. Technical Report CMU/SEI-2000-TR-008, ESC-TR-2000-007, Carnegie Mellon University, Software Engineering Institute, 2000.
22. L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practice*. Addison-Wesley, Massachusetts, Reading, 1998.
23. K. Beck and E. Gamma. Test infected: Programmers love writing tests. *Java Report*, 3(7):37–50, 1998.
24. B. Beizer. *Software Testing Techniques*. Van Nostrand Reinhold, New York, 2nd edition, 1990.
25. S. Bernardi, S. Donatelli, and J. Merseguer. From UML sequence diagrams and statecharts to analysable Petri Nets models. In *Proceedings of the 3rd International Workshop on Software and Performance(WOSP02)*, pages 35–45, 2002.
26. M. Bertoa and A. Vallecillo. Quality attributes for cots components. In *ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering, Malaga, Spain*, 2002.
27. A. Bertolino, F. Corradini, P. Inverardi, and H. Muccini. Deriving test plans from architectural descriptions. In *Proceedings of the 22nd international conference on Software engineering*, pages 220–229. ACM Press, 2000.
28. A. Bertolino, E. Marchetti, and R. Mirandola. Real-time UML-based performance engineering to aid manager's decisions in multi-project planning. In *Proceedings of the 3rd International Workshop on Software and Performance (WOSP-02)*, pages 251–261, New York, July 24–26 2002. ACM Press.
29. A. Bertolino, E. Marchetti, and A. Polini. Integrating "components" to test software components. In *Proceedings of 1st International Workshop on Testing*

and Analysis of Component Software at ETAPS 2003, April 13th 2003. Warsaw - Poland.
30. A. Bertolino and R. Mirandola. Modeling and analysis of non-functional properties in component-based systems. In *Proceedings of 1st International Workshop on Testing and Analysis of Component Software at ETAPS 2003*, April 13th 2003. Warsaw - Poland.
31. A. Bertolino and A. Polini. Re-thinking the development process of component-based software. In *Procediings ECBS 2002 Workshop On CBSE, Composing Systems From Components*, April 2002.
32. A. Bertolino and A. Polini. WCT: a wrapper for component testing. In *Proceedings of International Workshop Fidji'2002*, volume 2604 of *LNCS*, pages 141–151, Luxembourg, November 28-29 2002.
33. A. Bertolino and A. Polini. A framework for component deployment testing. In *Proceedings of 25th International Conference on Software Engineering*, pages 221–231, May 2003.
34. S. Beydeda and V. Gruhn. Integrating white- and black-box techniques for class-level testing object-oriented prototypes. In *Proceedings of the Software Engineering and Applications Conference*, pages 23–28, Nov. 2000.
35. S. Beydeda and V. Gruhn. An integrated testing technique for component-based software. In *Proceedings of the AICCSA ACS/IEEE International Conference on Computer Systems and Applications*, June 2001.
36. S. Beydeda and V. Gruhn. Merging components and testing tools: The self-testing cots components (stecc) strategy. In *Proceedings of the 29th EUROMICRO Conference (EUROMICRO'03)*, pages 107–114, Belek-Antalya, Turkey, Sept. 2003.
37. S. Beydeda and V. Gruhn. State of art in testing components. In *QSIC'03*, pages 146–153, Dallas, USA, Oct. 2003. IEEE Computer Society.
38. S. Beydeda, V. Gruhn, and M. Stachorski. A graphical representation of classes for integrated black- and white-box testing. In *International Conference on Software Maintenance (ICSM)*, pages 706–715. IEEE Computer Society Press, 2001.
39. R. V. Binder. *Testing Object-Oriented Systems: Models, Patterns, and Tools*. Addison-Wesley, 2000.
40. D. Binkley. Semantics guided regression test cost reduction. *IEEE Transactions on Software Engineering*, 23(8):498–516, Aug. 1997.
41. G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, 1999.
42. G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Software Development Process*. Addison-Wesley, 1999.
43. J. Bosch. *Design and use of software architectures: adopting and evolving a product-line approach*. Addison-Wesley, Harlow, 2000.
44. J. Bosch, P. Molin, M. Mattsson, P. Bengtsson, and M. Fayad. Framework problems and experiences. In M. Fayad, D. Schmidt, and R. Johnson, editors, *Object-Oriented Application Frameworks*, pages 55–82. John-Wiley, 1999.
45. L. C. Briand, V. R. Basili, and C. Hetmanski. Developing interpretable models for optimized set reduction for identifying high-risk software components. *IEEE Transactions on Software Engineering*, 19(11):1028–1034, 1993.
46. P. Broadwell, N. Sastry, and J. Traupman. FIG: A prototype tool for online verification of recovery mechanisms. In *ACM ICS SHAMAN Workshop*, Ney York, NY, June 2002.

47. A. W. Brown and K. C. Wallnau. The current state of cbse. *IEEE Software*, 15(5):37–46, 1998.
48. A. W. Brown and K. C. Wallnau. The current state of CBSE. *IEEE Software*, 15(5):37–46, 1998.
49. M. Broy, A. Deimel, J. Henn, K. Koskimies, F. Plasil, G. Pomberger, W. Pree, M. Stal, and C. Szyperski. What characterizes a (software) component? *Software – Concepts and Tools*, pages 49–56, 1998.
50. A. Brucker and B. Wolff. Checking OCL Constraints in Distributed Systems Using J2EE/EJB. Technical Report 157, Albert-Ludwigs-Universität Freiburg, `http://www.brucker.ch/bibliography/download/2001/tr01.pdf`, July 2001.
51. M. Buchi and W. Weck. The greybox approach: When blackbox specifications hide too much. Technical Report TUCS TR No. 297, Turku Centre for Computer Science, 1999. `http://www.tucs.abo.fi/`.
52. T. A. Budd. *Mutation Analysis: Ideas, Examples, Problems and Prospects*, pages 129–149. Computer Program Testing. North Holland, 1981. Chandrasekaran and Radicchi (eds.).
53. T. A. Budd and D. Angluin. Two notions of correctness and their relation to testing. *Acta Informatica*, 18:31–45, 1982.
54. G. A. Bundell, G. Lee, J. Morris, K. Parker, and P. Lam. A software component verification tool. In *Proceedings of International Conference on Software Methods and Tools (SMT2000)*, pages 137–146, Wollongong, Australia, November 6-10 2000.
55. U. Buy, C. Ghezzi, A. Orso, M. Pezze, and M. Valsasna. A framework for testing object-oriented components. In *International ICSE Workshop Testing Distributed Component-Based Systems (Los Angeles, USA)*, 1999.
56. C. Atkinson et al. *Component-based Product Line Engineering with UML*. Addison-Wesley, 2001.
57. C. Atkinson et al. *Component-based Product Line Engineering with UML*. Addison-Wesley, London, 2002.
58. D. Carney and F. Long. What do you mean by COTS? – finally, a useful answer. *IEEE Software*, 17(2), 2000.
59. J. Carreira, H. Madeira, and J. Silva. Xception: A technique for the experimental evaluation of dependability in modern computers. *Software Engineering*, 24(2):125–136, 1998.
60. A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 19(3):332–383, Aug 2001.
61. A. Cechich and M. Polo. Black-box Evaluation of COTS Components using Aspects and Metadata. In *Proceedings of the 4th International Conference on Product Focused Software Process Improvement, Springer-Verlag LNCS 2559*, pages 494–508, 2002.
62. A. Cechich and M. Prieto. Comparing Visual Component Composition Environments. In *Proceedings of the XXII International Conference of the Chilean Computer Science Society, IEEE Computer Society Press*, pages 217–225, 2002.
63. CERT. CERT Advisory. CA-2001-33, multiple vulnerabilities in wu-ftpd. `http://ftp.wu-ftpd.org/pub/wu-ftpd-attic/cert.org/CA-2001-33`.
64. J. Cheesman and J. Daniels. *UML Components - a Simple Process for Specifying Component-Based Software*. Addison-Wesley, 2000.

65. H. Y. Chen, T. H. Tse, F. T. Chan, and T. Y. Chen. In black and white: an integrated approach to class-level testing of object-oriented programs. *ACM Transactions on Software Engineering and Methodology*, 7(3):250–295, 1998.
66. H. Y. Chen, T. H. Tse, and T. Y. Chen. Taccle: a methodology for object-oriented software testing at the class and cluster levels. *ACM Transactions on Software Engineering and Methodology*, 10(1), 2001.
67. J. C. Cherniavsky and C. H. Smith. A recursion theoretic approach to program testing. *IEEE Transaction on Software Engineering*, 13(7):777–784, July 1987.
68. J. Choi. Aspect-Oriented Programming with Enterprise JavaBeans. In *Proceedings of the Fourth International Enterprise Distributed Object Computing Conference*, pages 252–262, 2000.
69. P. Clements, R. Kazman, and M. Klein. *Evaluating Software Architecture: Methods and Case Studies*. Addison-Wesley, 2002.
70. P. Clements and L. Northrop. *Software Product Lines: Practices and Patterns*. Addison-Wesley, Massachusetts, Boston, 2002.
71. Component+ Consortium. Built-in testing for component-based development, technical report d.3. `http://www.component-plus.org`, 2001.
72. K. Compton, Y. Gurevich, J. Huggins, and W. Shen. An automatic verification tool for UML, 2000.
73. J. C. Corbett, M. B. Dwyer, J. Hatcliff, and Robby. A language framework for expressing checkable properties of dynamic software. In *Proceedings of the* 7^{th} *SPIN Workshop*, volume 1885 of *LNCS*, August 2000.
74. V. Cortellessa and R. Mirandola. PRIMA-UML: a performance validation incremental methodology on early UML diagrams. *Science of Computer Programming*, 44:101–129, 2002.
75. F. Cristian. Exception handling and tolerance of software faults. In M. Lyu, editor, *Software Fault Tolerance*, pages 81–107. Wiley, 1995.
76. I. Crnkovic, B. Hnich, T. Jonsson, and Z. Kiziltan. Specification, implementation, and deployment of components. *Communications of the ACM*, 45(10):35–40, 2002.
77. I. Crnkovic and M. Larsson. A case study: demands on component-based development. In *ICSE'2000*, pages 22–30, Limerick, Ireland, June 2000.
78. I. Crnkovic and M. Larsson. Component-based software engineering – new paradigm of software development. 2001.
79. I. Crnkovic and M. Larsson, editors. *Building Reliable Component-Based Software System*. Artech House Publisher, 2002.
80. I. Crnkovic, H. Schmidt, J. Stafford, and K. Wallnau, editors. *The Journal of Systems and Software - Special issue on CBSE*, volume 65. 2003.
81. I. Crnkovic, H. Schmidt, J. Stafford, and K. C. Wallnau, editors. *6th ICSE workshop on CBSE: Automated Reasoning and Prediction*. May 2002. Orlando, Florida - USA.
82. D. Crocker. The verified design-by-contract paradigm. In *Safety-Critical Systems Symposium*, Meriden, Warwickshire, UK, February 2004.
83. G. Cugola, E. D. Nitto, and A. Fuggetta. The JEDI event-based infrastructure and its application to the development of the OPSS WFMS. *IEEE Transactions on Software Engineerings*, 27(9):827–850, Sept. 2001.
84. D. Garlan and S.Khersonsky and J.S. Kim. Model checking publish-subscribe systems. In *Proceedings of the* 10^{th} *SPIN Workshop*, volume 2648 of *LNCS*, May 2003.

85. W. Damm and D. Harel. LSCs: Breathing life into message sequence charts. *Formal Methods in System Design*, 19(1):45–80, 2001.
86. L. Davis, R. Gamble, and J. Payton. The impact of component architectures on interoperability. *The Journal of Systems and Software*, 61:31–45, 2002.
87. M. Davis and W. E. Metric space-based test-data adequacy criteria. *The Computer Journal*, 13(1):17–24, Feb. 1988.
88. S. Dawson, F. Jahanian, T. Mitton, and T.-L. Tung. Testing of fault-tolerant and real-time distributed systems via protocol fault injektion. In *Symposium on Fault-Tolerant Computing*, pages 404–414, 1996.
89. M. de Miguel, J. Ruiz, and M. Garcia. QoS-aware component frameworks. In *Proceedings of Tenth International Workshop on Quality of Service*, 2002.
90. J. C. Dean. Timing the testing of COTS software products. In *International ICSE Workshop Testing Distributed Component-Based Systems*, 1999.
91. M. E. Delamaro, J. C. Maldonado, and A. P. Mathur. Interface mutation: an approach to integration testing. *IEEE Transactions on Software Engineering*, 27(3):228–247, March 2001.
92. L. G. DeMichiel. Enterprise javabeans specification, version 2.1. Technical report, Sun Microsystems, 2002.
93. R. A. DeMillo, R. J. Lipton, and F. G. Sayward. Hints on test data selection: Help for the practising programmer. *IEEE Computer*, 11(4):34–41, April 1978.
94. J. P. DeVale and P. Koopman. Robust software – no more excuses. In *Proceedings of the International Conference on Dependable Systems and Networks*, June 2002.
95. L. K. Dillon and Y. S. Ramakrishna. Generating oracles from your favorite temporal logic specifications. In *Proceedings of the Fourth ACM SIGSOFT Symposium on the Foundations of Software Engineering*, volume 21 of *ACM Software Engineering Notes*, pages 106–117, New York, Oct.16–18 1996. ACM Press.
96. L. K. Dillon and Q. Yu. Oracles for checking temporal properties of concurrent systems. In *Proceedings of the ACM SIGSOFT '94 Symposium on the Foundations of Software Engineering*, pages 140–153, Dec. 1994.
97. L. Dobrica and E. Niemelä. A strategy for analyzing product line software architectures. Technical report, VTT Publications 427, VTT Technical Research Centre of Finland, Espoo, 2000.
98. L. Dobrica and E. Niemelä. Using uml notation extensions to model variability in product line architectures. In *ICSE, International workshop on Software Variability Management, Portland, USA*, pages 8–13, 2003.
99. F. Doucet, S. Shukla, and R. Gupta. Typing abstractions and management in a component framework. In *Proceedings of the Design Automation Conference*, Asia-South Pacific, January 2003.
100. D. D'Souza and A. C. Wills. *Objects, Components and Frameworks with UML: The Catalysis Approach.* Addison Wesley, Reading, MA, 1999.
101. J. Duran and S. Ntafos. An evaluation of random testing. *IEEE Trans. on Software Eng.*, 10:438–444, 1984.
102. EC. IST-1999-20162, Component+. `http://www.component-plus.org`, 2002.
103. H. Edler and J. Hörnstein. BIT in software components, european component+, 2001.
104. A. Egyed and C. Gacek. Automatically detecting mismatches during component-based and model-based development. In *14th IEEE International*

Conference on Automated Software Engineering, Florida, USA, pages 191–198, 1999.
105. N. S. Eickelmann and D. J. Richardson. An evaluation of software test environment architectures. In *Proceedings of the 18th international conference on Software engineering*, pages 353–364. IEEE Computer Society, 1996.
106. S. Elbaum, D. Gable, and G. Rothermel. Understanding and measuring the sources of variation in the prioritization of regression test suites. In *Proceedings of the Seventh International Software Metrics Symposium (METRICS 2001)*, pages 169–179, Apr. 2001.
107. S. Elbaum, A. Malishevsky, and G. Rothermel. Prioritizing test cases for regression testing. In *Proceedings of the ACM International Symposium on Software Testing and Analysis*, pages 102–112, Aug. 2000.
108. W. Emmerich. *Engineering Distributed Objects*. John-Wiley & Sons, 2000.
109. W. Emmerich. Software engineering and middleware. In *Proceedings of the 22th International Conference on Software Engineering (ICSE-00)*, pages 117–132. ACM Press, 2000.
110. W. Emmerich. Distributed component technologies and their software engineering implications. In *Proceedings of the 24th International Conference on Software Engineering (ICSE-02)*, pages 537–546. ACM Press, 2002.
111. W. Emmerich and J. Skene. Model driven performance analysis of enterprise information systems. In *Proceedings of the International Workshop on Testing and Analysis of Component-Based Systems(TACOS'03)*, 2003.
112. D. Engler, B. Chelf, A. Chou, and S. Hallem. Checking system rules using system-specific, programmer-written compiler extensions. In *Proceedings of the Fourth Symposium on Operating Systems Design and Implementation*, San Diego, CA, Oct. 2000.
113. D. Engler, D. Y. Chen, S. Hallem, A. Chou, and B. Chelf. Bugs as deviant behavior: a general approach to inferring errors in systems code. In *Proceedings of the eighteenth ACM symposium on Operating systems principles*, pages 57–72. ACM Press, 2001.
114. ETSI - Telecom Standards. The testing and test control notation: Core language, es 201 873-1, v.2.2.1, Oct. 2002.
115. ETSI - Telecom Standards. The ttcn-3 control interfaces (tci), es 201 873-6, v.1.0, Mar. 2002.
116. ETSI - Telecom Standards. The ttcn-3 run-time interface (tri), es 201 873-5, v.1.0, Oct. 2002.
117. P. T. Eugster, P. A. Felber, R. Guerraoui, and A. Kermarrec. The many faces of publish/subscribe. *ACM Computing Surveys*, 35(2):114–131, 2003.
118. H. Expert. `http://www.hugin.com`, 2001.
119. J.-C. Fabre, M. Rodriguez, J. Arlat, and J.-M. Sizun. Building dependable cots microkernel-based systems using mafalda. In *Proceedings of the Pacific Rim International Symposium on Dependable Computing (PRDC'00)*, pages 85–94, Los Angeles, California, Dec. 2000.
120. P. Felber. Transparent parallelization of java applications. In *Proceedings of the International Symposium on Distributed Objects and Applications (DOA'03)*, Nov. 2003.
121. N. E. Fenton and M. Neil. A critique of software defect prediction models. *IEEE Transactions on Software Engineering*, 25(5):675–689, 1999.

122. N. E. Fenton and N. Ohlsson. Quantitative analysis of faults and failures in a complex software system. *IEEE Transactions on Software Engineering*, 26(8):797–814, 2000.
123. C. Fetzer and K. Högstedt. Self*: A component based data-flow oriented framework for pervasive dependability. In *Eighth IEEE International Workshop on Object-oriented Real-time Dependable Systems (WORDS 2003)*, Jan. 2003.
124. C. Fetzer, K. Högstedt, and P. Felber. Automatic detection and masking of non-atomic exception handling. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN'03)*, June 2003.
125. C. Fetzer and Z. Xiao. Detecting heap smashing attacks through fault containment wrappers. In *Proceedings of the 20th IEEE Symposium on Reliable Distributed Systems*, Oct. 2001.
126. C. Fetzer and Z. Xiao. An automated approach to increasing the robustness of C libraries. In *Proceedings of the International Conference on Dependable Systems and Networks*, June 2002.
127. C. Fetzer and Z. Xiao. A flexible generator architecture for improving software dependability. In *Proceedings of the International Symposium on Software Reliability Engineering*, Nov. 2002.
128. C. Fetzer and Z. Xiao. Healers: A toolkit for enhancing the robustness and security of existing applications. In *Proceedings of the International Conference on Dependable Systems and Networks*, June 2003.
129. C. Floyd. A systematic look at prototyping. In R. Budde, K. Kuhlenkamp, L. Mathiassen, and H. Züllighoven, editors, *Approaches to Prototyping*, pages 1–18. Springer Verlag, 1984.
130. P. Frankl and J. Weyuker. An applicable family of data flow testing criteria. *IEEE Transactions on Software Engineering*, 14(10):1483–1498, Oct. 1988.
131. P. G. Frankl, R. G. Hamlet, B. Littlewood, and L. Strigini. Evaluating testing methods by delivered reliability. *IEEE Transactions on Software Engineering*, 24(8):586–601, 1998.
132. P. G. Frankl and J. E. Weyuker. A formal analysis of the fault-detecting ability of testing methods. *IEEE Transactions on Software Engineering*, 19(3):202–213, March 1993.
133. G. Froehlich, H. Hoover, L. Liu, and P. Sorenson. Hooking into object-oriented application frameworks. In *Proc. of 1997 Int. Conf. on Software Engineering*, pages 141–151. ACM, 1997.
134. S. Fujiwara, G. v. Bochmann, F. Khendek, M. Amalou, and A. Ghedamsi. Test selection based on finite state models. *IEEE Transactions on Software Engineering*, 17(6):591–603, June 1991.
135. E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns Elements of Reusable Object-Oriented Software*. Addison Wesley, 1995.
136. J. Gao, K. Gupta, S. Gupta, and S. Shim. On building testable software components. In *COTS-Based Software Systems (ICCBCC)*, volume 2255 of *LNCS*, pages 108–121. Springer Verlag, 2002.
137. J. Gao, E. Y. Zhu, and S. Shim. Monitoring software components and component-based software. In *Computer Software and Applications Conference (COMPSAC)*, pages 403–412. IEEE Computer Society Press, 2000.
138. J. Gao, E. Y. Zhu, and S. Shim. Tracking software components. *Journal of Object-Oriented Programming*, 14(4):13–22, 2001.

139. D. Garlan, R. Allen, and J. Ockerbloom. Architectural mismatch or why it's hard to build systems out of existing parts. In *17th International Conference on Software Engineering*, pages 179–185. ACM Press, Washington, Seatle, 1995.
140. D. Garlan, R. Allen, and J. Ockerbloom. Architectural mismatch: Why reuse is so hard. *IEEE Software*, 12(6), November 1995.
141. D. Garlan and M. Shaw. *Software Architecture: Perspective on an Emerging Discipline*. Prentice-Hall, 1996.
142. S. Ghosh and A. P. Mathur. Issues in testing distributed component-based systems. In *International ICSE Workshop Testing Distributed Component-Based Systems*, 1999.
143. S. Gnesi, D. Latella, and M. Massink. Model checking UML statecharts diagrams using JACK. In *Proceedings of the 4^{th} IEEE International Symposium on High Assuarance Systems Enginering (HASE)*, pages 46–55. IEEE Press, 1999.
144. S. S. Gokhale and M. R. Lyu. Regression tree modeling for the prediction of software quality. In *Proceedings of the Third ISSAT International Conference on Reliability and Quality in Design*, pages 31–36, Anaheim, California, Mar. 1997.
145. O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.
146. J. Goodenough. Exception handling: issues and a proposed notation. *Communications of the ACM*, 18(12):683–696, 1975.
147. J. B. Goodenough and S. L. Gerhart. Toward a theory of test data selection. *IEEE Transactions on Software Engineering*, 3, June 1975.
148. I. Gorton and A. Liu. Software component quality assessment in practice: successes and practical impediments. In *Proceedings of the 24th International Conference on Software Engineering (ICSE-02)*, pages 555–558, New York, 2002. ACM Press.
149. M. Goto. CINT C/C++ interpreter. `http://root.cern.ch/root/Cint.html`.
150. M. Goulao and F. B. e Abreu. The quest for software components quality. In *Proceedings of the 26th Annual International Computer Software and Applications Conference (COMPSAC'02)*, pages 313–318, Oxford,England, Aug. 2002.
151. F. Griffel. *Componentware: Konzepte und Techniken eines Softwareparadigmas*. dpunkt Verlag, 1998.
152. M. L. Griss. Software reuse architecture, process, and organization for business success. In *Proceedings of the Eighth Israeli Conference on Computer Systems and Software Engineering*, pages 86–98, Dan Accadia, Herzliya, June 1997.
153. H. Gross, C. Atkinson, and F. Barbier. Component integration through built-in contract testing. In *Component-Based Software Quality: Methods and Techniques*, volume LNCS 2693, pages 159–183, 2003.
154. H.-G. Gross. Testing and the UML – a perfect fit. Technical Report 110.03/E, Fraunhofer Institute for Experimental Software Engineering, Oct. 2003.
155. H.-G. Gross, C. Atkinson, and F. Barbier. Component integration through built-in contract testing. In Cechich, Piattini, and Vallcillo, editors, *Component-based Software Quality, Lecture Notes in Computer Science, Vol 2693*, Heidelberg, 2003. Springer.
156. H.-G. Gross, C. Atkinson, F. Barbier, N. Belloir, and M. Bruel. Built-in contract testing for component-based development. In Barbier, editor, *Business Component-Based Software Engineering*. Kluwer, 2003.

157. H.-G. Gross and N. Mayer. Built-in contract testing in component integration testing. *Electronic Notes in Theoretical Computer Science*, 82(6), 2003.
158. V. Gruhn and A. Thiel. *Komponentenmodelle: DCOM, JavaBeans, Enterprise JavaBeans, CORBA*. Addison-Wesley, 2000.
159. J. Grundy. Aspect-Oriented requirement Engineering for Component-Based Software Systems. In *Proceedings of the 4th IEEE International Symposium on Requirements Engineering*, pages 84–91, 1999.
160. J. Grundy. Multi-Perspective Specification, Design, and Implementation of Software Components using Aspects. *International Journal of Software Engineering and Knowledge Engineering*, 10(6):713–734, 2000.
161. C. A. Gunter and D. S. Scott. *Semantic domains*, volume B: Formal Models and Semantics of *Handbook of Theoretical Computer Science*, pages 633–674. The MIT Press/Elsevier, 1990. (Ed.) J. van Leeuwen.
162. D. Hamlet, D. Mason, and D. Woit. Theory of software reliability based on components. In *Proceedings of the 23rd international conference on Software engineering*, pages 361–370. IEEE Computer Society, 2001.
163. D. Hamlet and R. Taylor. Partition testing does not inspire confidence. *IEEE Trans. on Software Eng.*, 16(12):1402–1411, 1990.
164. S. Han, K. Shin, and H. Rosenberg. DOCTOR: An integrated software fault injection environment for distributed real-time systems. In *Proceedings of the International Computer Performance and Dependability Symposium (IPDS'95)*, pages 204–213, Erlangen, Germany, Apr. 1995.
165. R. A. Haraty, N. Mansour, and B. Daou. Regression testing of database applications. In *Proceedings of the 2001 ACM symposium on Applied computing*, pages 285–289. ACM Press, 2001.
166. M. Hardy. Cots components in software development. In *Computer Science Discipline Seminar Conference (CSCI 3901), University of Minnesota, Minnesota, USA*, 2000.
167. D. Harel. Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8:231–274, 1987.
168. D. Harel and E. Gery. Executable object modeling with statecharts. *IEEE Computer*, 30(7):31–42, 1997.
169. M. J. Harrold. Testing: A roadmap. In *The Future of Software Engineering (special volume of the proceedings of the International Conference on Software Engineering (ICSE))*, pages 63–72. ACM Press, 2000.
170. M. J. Harrold, D. Liang, and S. Sinha. An approach to analyzing and testing component-based systems. In *International ICSE Workshop Testing Distributed Component-Based Systems*, 1999.
171. M. J. Harrold, A. Orso, D. Rosenblum, G. R. G., M. L. Soff, and H. Do. Using Component Metadata to Support the Regression Testing of Component-Based Software. Technical Report GIT-CC-01-38, College of Computing, Georgia Institute of Technology, 2001.
172. M. J. Harrold and M. L. Soffa. Selecting and using data for integration testing. *IEEE Software*, pages 58–65, March 1991.
173. J. Hatcliff, W. Deng, M. Dwyer, G. Jung, and V. Ranganath. Cadena: An integrated development, analysys, and verification environment for component-based systems. In *Proceedings of the 25^{th} International Conference on Software Engineering*, pages 160–172, May 2003.
174. M. Hayden. *The Ensemble System*. PhD thesis, cornell, Jan. 1998.

175. S. Heiler. Semantic interoperability. *ACM Computing Surveys*, 27(2):271–279, 1995.
176. G. T. Heineman and W. T. C. (ed.). *Component-Based Software Engineering: Putting the Pieces Together*. Addison-Wesley, Reading, MA, 2001.
177. R. Helm, I. Holland, and D. Gangopadhyay. Contracts: Specifying behavioral compositions in object-oriented systems. In *OOPSLA-ECOOP*, pages 169–180, 1990.
178. D. Hemer and P. Lindsay. Supporting component-based reuse in CARE. In *Proceedings of the 25th Australasian conference on Computer science - Volume 4*, Melbourne, Victoria, Australia, 2002.
179. D. Hoffman and P. Strooper. Graph-based class testing. *The Australian Computer Journal*, 26(4):158–163, 1994.
180. D. Hoffman and P. Strooper. The testgraph methodology: Automated testing collection classes. *Journal of Object Oriented Programming*, 8(7):35–41, 1995.
181. G. Holzmann. *Design and Validation of Network Protocols*. Prentice Hall, 1991.
182. G. Holzmann. The model checker SPIN. *IEEE Transactions on Software Engineering*, 23(5):279–295, May 1997.
183. H. S. Hong, Y. R. Kwon, and S. D. Cha. Testing of object-oriented programs based on finite state machines. In *Second Asia-Pacific Software Engineering Conference (Brisbane, Australia)*, pages 234–241. IEEE Computer Society Press, 1995.
184. Hong Kong Productivity Council. `http://www.hkpc.org/itd/servic11.htm`, 2000.
185. J. Hopkins. Component primer. *C. ACM*, 43(10):27–30, Oct. 2000.
186. J. Hörnestein and H. Edler. Test reuse in cbse using built-in tests. In *Proceedings of the 9th IEEE Conference and Workshops on Engineering of Computer-Based Systems. Workshop on Component-based Software Engineering*, 2002.
187. C. Horstmann. *Mastering Object-Oriented Design in C++*. Wiley, 1995.
188. W. E. Howden. Reliability of the path analysis testing strategy. *IEEE Transaction on Software Engineering*, 2(9):208–215, Sept. 1976.
189. Y.-W. Huang, S.-K. Huang, T.-P. Lin, and C.-H. Tsai. Web application security assessment by fault injection and behavior monitoring. In *Proceedings of the twelfth international conference on World Wide Web*, pages 148–159. ACM Press, 2003.
190. D. Hybertson. A uniform component modeling space. *Informatica*, 25(4), November 2001.
191. D. Hybertson. Strengthening the modeling foundation of the mda. In *Workshop in Software Model Engineering at UML 2002*, `http://www.metamodel.com/wisme-2002/papers/hybertson.pdf`, October 2002.
192. IBM. `http://www4.ibm.com/software/ad/sanfrancisco`, 2000.
193. F. IGD. Rin system specification. Technical report, Fraunhofer Institut für Graphische Datenverarbeitung, Darmstadt, Germany, 2004.
194. Institute of Electrical and Electronics Engineers. *IEEE 610.12-1990: IEEE Standard Glossary of Software Engineering Terminology*, 1990.
195. Institution of Electrical and Electronics Engineers. Ieee std 1063-1987, ieee standard for software user documentation, 1987.
196. Institution of Electrical and Electronics Engineers. Ieee std 1016.1-1993, ieee guide to software design descriptions, 1993.

197. Institution of Electrical and Electronics Engineers. Ieee std 829-1998, ieee standard for software test documentation, 1998.
198. Institution of Electrical and Electronics Engineers. Ieee std 829-1998, ieee standard for software test documentation, 2002.
199. International Organization for Standardadization. ISO/IEC (1991) information technology - software product evaluation - quality characteristics and guidelines for their use.
200. International Organization for Standardization. *ISO 8402: Quality management and quality assurance – Vocabulary*, 1994.
201. International Organization for Standardization. Basic reference model of open distributed processing, 1995.
202. International Organization for Standardization. *ISO/IEC 14598-5: Information technology – Software product evaluation – Part 5: Process for evaluators*, 1998.
203. International Organization for Standardization. *ISO/IEC 14598-1: Information technology – Software product evaluation – Part 1: General overview*, 1999.
204. International Organization for Standardization. *ISO/IEC 14598-4: Software engineering – Product evaluation – Part 4: Process for acquirers*, 1999.
205. International Organization for Standardization. *ISO/IEC 14598-2: Software engineering – Product evaluation – Part 2: Planning and management*, 2000.
206. International Organization for Standardization. *ISO/IEC 14598-3: Software engineering – Product evaluation – Part 3: Process for developers*, 2000.
207. International Organization for Standardization. *ISO/IEC 14598-5: Software engineering – Product evaluation – Part 6: Documentation of evaluation modules*, 2001.
208. International Organization for Standardization. *ISO/IEC 9126-1: Software engineering – Product quality – Part 1: Quality model*, 2001.
209. P. Inverardi, H. Muccini, and P. Pelliccione. Automated check of architectural models consistency using SPIN. In *Proceedings of the 16th IEEE International Conference on Automated Software Engineering conference (ASE)*, pages 349–349, 2001.
210. H. Ishikawa, Y. Ogata, K. Adachi, and T. Nakajima. Requirements for a component framework of future ubiquitous computing. In *Proceedings of the IEEE Workshop on Software Technologies for Future Embedded Systems (WSTFES'03)*, 2003.
211. I. Jacobson, M. Christerson, P. Jonsson, and G. Overgaard. *Object-Oriented Software Engineering: A Use Case Driven Approach*. Addison-Wesley, Reading, MA, 1992.
212. M. Jazayeri, A. Ran, and F. van der Linden. *Software Architecture for Product Families*. Addison-Wesley, Boston, 2000.
213. T. Jeon, H. W. Seung, and S. Lee. Embedding built-in tests in hot spots of an object-oriented framework. *SIGPLAN Not.*, 37(8):25–34, 2002.
214. Z. Jin and J. Offutt. Integration testing based on software couplings. In *COMPSAC'95*, pages 13–23, Gaithersburg, Maryland, June 1995.
215. R. Johnson and B. Foote. Designing reusable classes. *Journal of OOP*, 1:26–49, 1988.
216. N. G. K. E. Emam, S. Benlarbi and S. N. Rai. Comparing case-based reasoning classifiers for predicting high risk software components. *The Journal of Systems and Software*, 55(3):301–320, 2001.

217. P. Kallio and T. Ihme. Evolution of the use and risks of the commercial software components. In *28th Euromicro Conference, Dortmund, DE*, pages 55 – 61, 2002.
218. S. H. Kan. *Metrics and Models in Software Quality Engineering (Second Edition)*. Addison-Wesley, Reading, MA, 2003.
219. G. Kanawati, N. Kanawati, and J. Abraham. FERRARI: A tool for the validation of system dependability properties. In *Proceedings of the 22nd International Symposium on Fault Tolerant Computing (FTCS-22)*, pages 336–344, Boston, Massachusetts, 1992. IEEE.
220. W. Kao and R. Iyer. A user-oriented synthetic workload generator. In *12th International Conference on Distributed Computing Systems (ICDCS '92)*, pages 270–277. IEEE Computer Society Press, June 1992.
221. N. Kaveh and W. Emmerich. Deadlock detection in distributed object systems. In *Proceedings of the joint 8^{th} European Software Engineering Conference (ESEC) and 9^{th} ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE)*, pages 44–51, 2001.
222. A. Kelkar and R. Gamble. Understanding the architectural characteristics behind middleware choices. In *1st Conference on Information Reuse and Integration, Georgia, Atlanta*, 1999.
223. R. Keshav and R. Gamble. Towards a taxonomy of architecture integration strategies. In *3rd International Software Architecture Workshop, Florida, Orlando*, 1998.
224. A. A. Keshlaf and K. Hashim. A model and prototype tool to manage software risks. In *Proceedings of the First Asia-Pacific Conference on Quality Software*, pages 297–305, Kowloon, Hong Kong, Oct. 2000.
225. G. Kiczales and J. des Rivières. *The Art of the Metaobject Protocol*. MIT Press, 1991.
226. G. Kiczales, J. Lamping, A. Menhdhekar, C. Maeda, C. Lopes, J.-M. Loingtier, and J. Irwin. Aspect-oriented programming. In M. Akşit and S. Matsuoka, editors, *Proceedings European Conference on Object-Oriented Programming*, volume 1241, pages 220–242. Springer-Verlag, Berlin, Heidelberg, and New York, 1997.
227. K. N. King and A. J. Offutt. A fortran language system for mutation-based software testing. *Software–Practice and Experience*, 21(7):685–718, July 1991.
228. S. Kirani and W. Tsai. Method sequence specification and verification of classes. *Journal of Object-Oriented Programming*, pages 28–38, Oct. 1994.
229. KLGroup. `http://www.klgroup.com`, 2001.
230. J. Kontio. A case study in applying a systematic method for cots selection. In *18th International Conference on Software Engineering, Berlin, Germany*, pages 201 – 209. IEEE Computer Society Press, 1996.
231. P. Koopman and J. DeVale. The exception handling effectiveness of POSIX operating systems. *IEEE Transactions on Software Engineering*, 26(9):837–848, Sep 2000.
232. B. Korel. Black-box understanding of COTS components. In *Proceedings: Seventh International Workshop on Program Comprehension*, pages 92–99. IEEE Computer Society Press, 1999.
233. M. Koutlis, P. Kourouniotis, K. Kyrimis, and N. Renieri. Inter-component communication as a vehicle towards end-user modeling. In *ICSE Workshop on Component-Based Software Engineering*, 1998.

234. W. Kozaczynski and G. Booch. Component-based software engineering. *IEEE Software*, 15(5):34–36, 1998.
235. R. Kramer. iContract - The Java Design by Contract Tool. In *Proceedings of Technology of Object-Oriented Languages and Systems*, pages 295–307, Santa Barbara, California, August 03-07 1998.
236. N. Kranitis, A. Paschalis, D. Gizopoulos, and Y. Zorian. Effective software self-test methodology for processor cores. In *Proceedings of the conference on Design, automation and test in Europe*, page 592. IEEE Computer Society, 2002.
237. N. Kropp, P. K. Jr., and D. Siewiorek. Automated robustness testing of off-the-shelf software components. In *Proceedings of the Symposium on Fault-Tolerant Computing (FTCS)*, pages 230–239, 1998.
238. A. Krstic, W. C. Lai, K. T. Cheng, L. Chen, and S. Dey. Embedded software-based self-testing for soc design. In *Proceedings of the 39th conference on Design automation*, pages 355–360. ACM Press, 2002.
239. P. Kruchten. *The Rational Unified Process: An Introduction.* Addison Wesley Longman, 2000.
240. P. Lago and M. Matinlassi. The wise approach to architect wireless services. In *Proceedings of the 4th International Conference in Product Focused Software Process Improvement, PROFES2002*, pages 367 – 382, Berlin, Heidelberg, 2002. Springer.
241. D. Lea. Collections. `http://gee.cs.oswego.edu/dl/classes/collections`.
242. X. Leroy. The Objective Caml system release 2.03, June 2001.
243. J. Lilius and I. Paltor. vUML: a tool for verifying UML models. In *Proceedings of the 14^{th} IEEE International Conference on Automated Software Engineering (ASE)*, pages 255–258, October 1999.
244. C. Lin, A. Avritzer, E. Weyuker, and L. Sai-Lai. Issues in interoperability and performance verification in a multi-orb telecommunications environment. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN 2000)*, pages 567–575, 2000.
245. J.-L. Lions. Ariane 5, flight 501 failure, report by the inquiry board. `http://java.sun.com/people/jag/Ariane5.html`, 1996.
246. M. Lippert and C. V. Lopes. A study on exception detection and handling using aspect-oriented programming. In *Proceedings of the 22nd international conference on Software engineering*, pages 418–427. ACM Press, 2000.
247. Y. Liu, I. Gorton, A. Liu, N. Jiang, and S. Chen. Designing a test suite for empirically-based middleware performance prediction. In *Fortieth International Conference on Technology of Object-Oriented Languages and Systems (TOOLS Pacific 2002)*, Sydney, Australia, 2002. ACS.
248. D. Luckham, J. Vera, and S. Meldal. Three concepts of system architecture. Technical Report CSL–TR–95–67, Computer Systems Lab, Stanford University, July 1995.
249. C. Lüer and D. Rosenblum. WREN - An Environment for Component-Based Development. In *Proceedings of the Joint 8th European Software Engineering Conference and 9th ACM Sigsoft International Symposium on the Foundations of Software Engineering (FSE-9)*, pages 207–217, 2001.
250. R. Lutz and G. Gannod. Analysis of a software product line architecture: an experience report. *Journal of Systems and Software*, 66(3):253 – 267, 2003.
251. M. R. Lyu, editor. *Handbook of Software Reliability Engineering.* McGraw-Hill, New York, 1996.

252. M. R. Lyu. Software reliability theory. In J. J. Marciniak, editor, *Encyclopedia of Software Engineering*. Wiley, New York, 2001.
253. M. R. Lyu, J. S. Yu, E. Keramidas, and S. R. Dalal. Armor: Analyzer for reducing module operational risk. In *Proceedings of Twenty-Fifth International Symposium on Fault-Tolerant Computing (FTCS-25)*, pages 137–142, Pasadena, California, June 1995.
254. J. Magee, N. Dulay, S. Eisenbach, and J. Kramer. Specifying distributed software architectures. In *Proceedings 5th European Software Engineering Conference (ESEC 95)*, pages 137–153, Sitges, Spain, 1995.
255. E. Martins, C. Toyota, and R. Yanagawa. Constructing Self-Testable Software Components. In *Proceedings of the 2001 International Conference on Dependable Systems and Networks*, pages 151–160, 2001.
256. M. Matinlassi and E. Niemelä. The impact of maintainability on component-based software systems. In *Proceedings of the 29th Euromicro Conference, Antalya, Turkey*, 2003.
257. M. Matinlassi, E. Niemelä, and L. Dobrica. Quality-driven architecture design and quality analysis method, a revolutionary initiation approach to a product line architecture. Technical report, VTT Technical Research Centre of Finland, Espoo, 2002.
258. P. M. Maurer. Components: What if they gave a revolution and nobody came? *IEEE Computer*, 33(6):28–34, June 2000.
259. R. Maxion and R. Olszewski. Eliminating exception handling errors with dependability cases: a comparative, empirical study. *IEEE Transactions on Software Engineering*, 26(9):888–906, 2000.
260. T. McCabe, L. Dreyer, A. Dunn, and A. Watson. Testing an object-oriented application. *J. of the Quality Assurance Institute*, 8(4):21–27, 1994.
261. D. McIllroy. Mass produced software components. In P. Naur and B. Randall, editors, *Software Engineering: Report on a Conference by the NATO Science Committee*, pages 138–155, 1969.
262. M. D. McIlroy. Mass produced software components. In *NATO Software Engineering Conference*, pages 138–155, 1968.
263. K. McMillan. *Symbolic Model Checking*. Kluwer Academic, 1993.
264. N. Medvidovic, E. Dashofy, and R. Taylor. On the role of middleware in architecture-based software development. *International Journal of Software Engineering and Knowledge Engineering*, 13(4), 2003.
265. N. Medvidovic, P. Oreizy, J. E. Robbins, and R. N. Taylor. Using object-oriented typing to support architectural design in the c2 style. In *Proceedings of 4th Symposium on the Fundation of Software Engineering (FSE4)*, pages 24–32. ACM Press, October 1996. San Francisco, California (USA).
266. N. Medvidovic, D. Rosenblum, D. Redmiles, and J. Robbins. Modeling software architectures in the unified modeling language. *ACM Transcations on Software Engineering and Methodology*, 11(1):2–57, January 2002.
267. N. Mehta, N. Medvidovic, and S. Phadke. Towards a taxonomy of software connectors. In *Proceedings of the 22nd International Conference on Software Engineering (ICSE-00)*, pages 178–187. ACM Press, 2000.
268. P. Melliar-Smith and B. Randell. Software reliability: The role of programmed exception handling. In *Proceedings of the ACM conference on Language Design for Reliable Software*, pages 95–100, 1977.
269. S. Mellor and S. Balcer. *Executable UML - A Foundation for Model-Driven Architecture*. Addison-Wesley, 2002.

270. A. M. Memon, M. E. Pollack, and M. L. Soffa. Automated test oracles for GUIs. In *Proceedings of the ACM SIGSOFT 8th International Symposium on the Foundations of Software Engineering (FSE-8)*, pages 30–39, NY, Nov. 8–10 2000.
271. A. M. Memon and M. L. Soffa. Regression testing of GUIs. In *Proceedings of the 9th European software engineering conference held jointly with 10th ACM SIGSOFT international symposium on Foundations of software engineering*, pages 118–127. ACM Press, 2003.
272. A. M. Memon, M. L. Soffa, and M. E. Pollack. Coverage criteria for GUI testing. In *Proceedings of the 8th European Software Engineering Conference (ESEC) and 9th ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE-9)*, pages 256–267, Sept. 2001.
273. P. Merle. CORBA 3.0 new components chapters. Technical report, TC Document ptc/2001-11-03, Object Management Group, 2001.
274. Metamata. `http://www.metamata.com`, 2001.
275. B. Meyer. Applying design by contract. *IEEE Computer*, 25(10):40–51, October 1992.
276. B. Meyer. *Object-Oriented Software Construction*. Prentice Hall, second edition edition, 1997.
277. B. Meyer. *Object-oriented Software Construction*. Prentice-Hall, Upper Saddle River, 1999.
278. B. Meyer. The grand challenge of trusted components. In *Proceedings of the 25th International Conf. on Software engineering*, May 2003.
279. B. Meyer, C. Mingins, and H. Schmidt. Trusted components for the software industry. `http://trusted-components.org/documents/tc_original_paper.html`.
280. B. Meyers and P. Oberndorf. *Managing Software Acquisition: Open Systems and COTS Products*. Addison-Wesley, New York, 2001.
281. Microsoft Corporation. Microsoft .NET framework. `http://www.microsoft.com/net/`.
282. Microsoft Corporation. .Net resources. `http://www.microsoft.com/net/`. Access date April 5th, 2004.
283. Microsoft Corporation. The component object model specification. `http://www.microsoft.com/COM/resources/COM1598D.ZIP`, 1995.
284. Microsoft Corporation. `http://www.microsoft.com/isapi`, 2000.
285. H. D. Mills. Top-Down Programming in Large Systems. In R. Ruskin, editor, *Debugging Techniques in Large Systems*. Prentice Hall, 1971.
286. S. Mitchell, A. Burns, and A. Wellings. Mopping up exceptions. *ACM SIGAda Ada Letters*, XXI(3):80–92, 2001.
287. R. T. Mittermeir, A. Bollin, H. Pozewaunig, and D. Rauner-Reithmayer. Goal-driven combination of software comprehension approaches for component based development. In *Proceedings of the 2001 symposium on Software reusability*, pages 95–102. ACM Press, 2001.
288. L. J. Morell. *A Theory of Error-based Testing*. PhD thesis, University of Maryland, Department of Computer Science, 1984.
289. M. Morisio, C. Seaman, A. Parra, V. Basili, S. Kraft, and S. Condon. Investigating and improving a COTS-based software development process. In *International Conference on Software Engineering (ICSE)*, pages 32–41. ACM Press, 2000.

290. M. Morisio, C. B. Seaman, V. R. Basili, A. T. Parra, S. E. Kraft, and S. E. Condon. COTS-based software development: Processes and open issues. *The Journal of Systems and Software*, 61(3):189–199, 2002.
291. M. Morisio and M. Torchiano. Definition and classification of COTS: A proposal. In *COTS-Based Software Systems (ICCBSS)*, volume 2255 of *LNCS*, pages 165–175. Springer Verlag, 2002.
292. J. Morris, P. Lam, G. Lee, K. Parker, and G. A. Bundell. Determining component reliability using a testing index. In *Proceedings of the twenty-fifth Australasian conference on Computer science*, pages 167–176. Australian Computer Society, Inc., 2002.
293. J. Morris, G. Lee, K. Parker, G. A. Bundell, and C. P. Lam. Software component certification. *IEEE Computer*, 34(9):30–36, September 2001.
294. M. Morrison. *Presenting JavaBeans: SunSITE India.* Virtual Library, 1997.
295. C. J. Mueller and B. Korel. Automated evaluation of COTS components. In *International Workshop on Automated Program Analysis, Testing and Verification (Limerick, Ireland)*, 2000.
296. R. Mukherjee, J. Jain, K. Takayama, and M. Fujita. Automatic partitioning for efficient combinatorial verification. In *Proceedings of the 2000 conference on Asia South Pacific design automation*, pages 67–72. ACM Press, 2000.
297. J. Munson and T. Khoshgoftaar. The detection of fault-prone programs. *IEEE Transactions on Software Engineering*, 18(5):423–433, 1992.
298. J. D. Musa. *Software Reliability Engineering.* McGraw-Hill, New York, 1998.
299. G. J. Myers. *The Art of Software Testing.* John Wiley and Sons, New York, 1979.
300. National Coordination Office for Information Technology Research and Development. January: High confidence software and systems research needs. `http://www.ccic.gov/iwg/hcss.html`, 2001.
301. National Product Line Asset Center. Nplace, 2002.
302. C. Ncube and N. Maiden. Cots software selection: The need to make tradeoffs between system requirements, architectures and cots/components. In *ICSE 2000 Workshop on Continuing Collaborations for successful COTS Development, Limerick, Ireland*, 2000.
303. S. H. News. Oops! Linux bug escapes early. `http://www.securityfocus.com/news/293`.
304. E. Niemelä and M. Holappa. Experiences with the use of corba. In *Proceedings of the 24th EUROMICRO Conference, Västerås, SE*, pages 989 – 996. IEEE Computer Society, 1998.
305. E. Niemelä and T. Ihme. Product line software engineering of embedded systems. *ACM SIGSOFT Software Engineering Notes*, 26:118 – 125, 2001.
306. E. Niemelä, M. Matinlassi, and P. Lago. Architecture-centric approach to wireless service engineering. *IEC, Annual Review of Communications*, 56:875 – 889, 2003.
307. E. Niemelä, H. Perunka, and T. Korpipää. A software bus as a platform for a family of distributed embedded system products. In F. Linden, editor, *Development and Evolution of Software Architectures for Product Families*, pages 14–23. Springer, 1998.
308. J. Q. Ning, K. Miriyala, and W. Kozaczynski. An architecture-driven, business-specific, and component-based approach to software engineering. In *Proceedings of Third International Conference on Software Reuse: Advances in Software Reusability*, pages 84–93, Rio De Janeiro, Brazil, Nov. 1994.

309. Object Management Group. CORBA Component Model specifications. http://www.omg.org/technology/documents/formal/components.htm. Access date April 5th, 2004.
310. Object Management Group. http://www.omg.org/corba/whatiscorba.html, 2000.
311. Object Management Group. CORBA component model v3, formal/2002-06-65. http://www.omg.org/technology/documents/formal/components.htm, 2002.
312. Object Management Group. Corba components. http://www.omg.org/cgi-bin/doc?formal/02-06-65.pdf, 2002.
313. Object Management Group. XMI: XML metadata interchange v.1.2. http://www.omg.org/, 2002.
314. Object Management Group. Adtf, 2nd revised submission on unified modeling language: Superstructure. version 2.0, 2003.
315. Object Management Group. Adtf: The uml 2.0 testing profile, July 2003.
316. Object Management Group. Omg unified modeling language specification, version 1.5. http://www.omg.org/technology/documents/formal/uml.htm, 2003.
317. S. W. O'Malley and L. L. Peterson. A dynamic network architecture. *ACM Transactions on Computer Systems*, 10(2):110–143, May 1992.
318. A. K. Onoma, W.-T. Tsai, M. Poonawala, and H. Suganuma. Regression testing in an industrial environment. *Commun. ACM*, 41(5):81–86, 1998.
319. A. Orso. Component Metadata for Software Engineering Tasks. In *Proceedings of the Second International Workshop on Engineering Dsitributed Objects, Springer-Verlag LNCS 1999*, pages 126–140, 2000.
320. A. Orso, M. J. Harrold, and D. Rosenblum. Component metadata for software engineering tasks. In W. Emmerich and S. Tai, editors, *Proceeedings of International Conference on Engineering Distributed Objects 2000*, LNCS 1999, pages 129–144, 2000.
321. A. Orso, M. J. Harrold, D. Rosenblum, G. Rothermel, M. L. Soffa, and H. Doo. Using component metadata to support the regression testing of component-based software. In *Proceedings of the International Conference on Software Maintenance (ICSM2001)*, pages 716–725, Florence, Italy, November 6-10 2001.
322. A. Orso, M. J. Harrold, and D. S. Rosenblum. Component metadata for software engineering tasks. In *Revised Papers from the Second International Workshop on Engineering Distributed Objects*, pages 129–144. Springer-Verlag, 2001.
323. T. J. Ostrand and M. J. Balcer. The category-partition method for specifying and generating functional tests. *Communications of the ACM, CACM*, 31(6):676–686, June 1988.
324. R. Paige and J. Ostroff. A proposal for a lightweight rigorous uml-based development method for reliable systems. In *Proceedings of Workshop on Practical UML-Based Rigorous Development Methods 2001 (co-located with UML 2001)*, Toronto, Canada, October 2001.
325. H. D. Pandi, B. G. Ryder, and W. Landi. Interprocedural def-use associations in c programs. In *Proc. of TAV4*, pages 139–153, Oct. 1991.
326. A. S. Parrish and S. H. Zweben. Analysis and refinement of software test data adequacy properties. *IEEE Transactions on Software Engineering*, 17(6):565–581, June 1991.
327. A. S. Parrish and S. H. Zweben. Clarifying some fundamental concepts in software testing. *IEEE Transactions on Software Engineering*, 19(7):742–746, July 1993.

328. B. Perens. Electricfence. `ftp://ftp.perens.com/pub/ElectricFence/`.
329. D. Petriu, C. Shousha, and A. Jalnapurkar. Architecture-based performance analysis applied to a telecommunication system. *IEEE Transactions on Software Engineering*, 26(11):1049–1065, 2000.
330. M. Pezzè and M. Young. *Software Testing and Analysis: Process, Principles, and Techniques*. John Wiley, 2004.
331. A. Pnueli. The temporal logic of programs. In *Proceedings of 18th IEEE Symposium Foundations of Computer Science(FOCS)*, pages 46–57, October 1977.
332. M. Polo. Automating Testing of Java Programs using Reflection. In *Proceedings of the ICSE 2nd International Workshop on Automated Program Analysis, Testing, and Verification, WAPATV*, 2001.
333. R. Pooley. Using UML to derive stocastic process algebra models. In *Proceedings of the 15th UK Performance Engineering Workshop (UKPEW)*, pages 23–34, 1999.
334. A. A. Porter and R. W. Selby. Empirically guided software development using metric-based classification trees. *IEEE Software*, 7(2):46–53, 1990.
335. G. Pour. Component-based software development approach: New opportunities and challenges. In *Proceedings of Technology of Object-Oriented Languages Tools 26*, pages 375–383, Santa Barbara, California, Aug. 1998.
336. G. Pour. Enterprise javabeans, javabeans & xml expanding the possibilities for web-based enterprise application development. In *Proceedings of Technology of Object-Oriented Languages and Systems*, pages 282–291, Nancy, France, June 1999.
337. G. Pour, M. Griss, and J. Favaro. Making the transition to component-based enterprise software development: Overcoming the obstacles - patterns for success. In *Proceedings of Technology of Object-Oriented Languages and systems*, pages 419–419, Nancy, France, June 1999.
338. A. Purhonen, E. Niemelä, and M. Matinlassi. Viewpoints of dsp software and service architectures. *Journal of Systems and Software*, 69(1–2):57–73, 2004.
339. C. Rajaraman and M. R. Lyu. Reliability and maintainability related software coupling metrics in C++ programs. In *Proceedings 3rd IEEE International Symposium on Software Reliability Engineering (ISSRE'92)*, pages 303–311, North Carolina, USA, Oct. 1992.
340. C. Rajaraman and M. R. Lyu. Some coupling measures for C++ programs. In *Proceedings of TOOLS USA 92 Conference*, pages 225–234, Santa Barbara, California, Aug. 1992.
341. B. Randell and J. Xu. The evolution of the recovery block concept. In M. Lyu, editor, *Software Fault Tolerance*, pages 1–21. Wiley, 1995.
342. T. Ravichandran and M. A. Rothenberger. Software reuse strategies and component markets. *Communications of the ACM*, 46(8):109–114, 2003.
343. D. Richardson and A. Wolf. Software testing at the architectural level. In *Proc. of 2nd International Software Architecture Workshop*, pages 68–71, San Francisco, California, Oct. 1996. ACM Press.
344. D. J. Richardson. Taos: Testing with analysis and oracle support. In *Proceedings of the 1994 international symposium on Software testing and analysis*, pages 138–153. ACM Press, 1994.
345. D. J. Richardson, S. Leif-Aha, and T. O. OMalley. Specification-based Test Oracles for Reactive Systems. In *Proceedings of the 14th International Conference on Software Engineering*, pages 105–118, May 1992.

346. E. H. Riedemann. *Testmethoden für sequentiell und nebenläufige Software-Systeme*. B. G. Teubner, 1997.
347. Robby, M. B. Dwyer, and J. Hatcliff. Bogor: an extensible and highly-modular software model checking framework. In *Proceedings of the joint 9th European Software Engineering Conference (ESEC) and 10th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE)*, pages 267–276, 2003.
348. A. Romanovsky, C. Dony, J. L. Knudsen, and A. Tripathi, editors. *Advances in Exception Handling Techniques*. Springer Verlag, 2001.
349. D. Rosenblum. Challenges in exploiting architectural models for software testing. In *Proceedings of the International Workshop on the Role of Software Architecture in Testing and Analysis (ROSATEA)*, 1998.
350. D. S. Rosenblum. Adequate testing of component-based software. Technical Report 97-34, University of California, Department of Information and Computer Science, 1997.
351. G. Rothermel and M. J. Harrold. A safe, efficient regression test selection technique. *ACM Transactions on Software Engineering and Methodology*, 6(2):173–210, 1997.
352. G. Rothermel, M. J. Harrold, and J. Dedhia. Regression test selection for C++ software. *Software Testing, Verification and Reliability*, 10(2):77–109, 2000.
353. A. Rountev, A. Milanova, and B. Ryder. Fragment class analysis for testing of polymorphism in java software. In *Int. Conf. on Softw. Eng.*, pages 210–220, 2003.
354. J. Rumbaugh, I. Jacobson, and G. Booch. *The Unified Modeling Language Reference Manual*. Addison Wesley Lognman, 1999.
355. S. Bodoff et al. *The J2EE Tutorial*. Addison-Wesley, 2002.
356. F. Salles, M. Rodriguez, J.-C. Fabre, and J. Arlat. Metakernels and fault containment wrappers. In *Proceedings of the 29th International Symposium on Fault-Tolerant Computing*, June 1999.
357. P. Santos, T. Ritter, and M. Born. Rapid engineering of collaborative and adaptive multimedia systems on top of corba components. In K. Irmscher, editor, *Kommunikation in Verteilten Systemen*. VDE, Offenbach, 2003.
358. S. Savage, M. Burrows, G. Nelson, P. Sobalvarro, and T. Anderson. Eraser: a dynamic data race detector for multi-threaded programs. In *Proceedings of the sixteenth ACM symposium on Operating systems principles*, pages 27–37. ACM Press, 1997.
359. T. Schäfer, A. Knapp, and S. Merz. Model checking UML state machines and collaborations. *Electronic Notes in Theoretical Computer Science*, 55(3):13 pages, 2001.
360. C. H. Schmauch. *ISO9000 for Software Developers*. ASQC Quality Press, Milwaukee, Wisconsin, 1994.
361. R. Seacord and L. Wrage. Replaceable components and the service provider interface. Technical report, Software Engineering Institute, 2002.
362. S. Sedigh-Ali, A. Ghafoor, and R. A. Paul. Metrics and models for cost and quality of component-based software. In *Proceedings of the Sixth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'03)*, pages 149–155, Hokkaido, Japan, May 2003.
363. Z. Segall, D. Vrsalovic, D. Siewiorek, D. Yaskin, J. Kownacki, J. Barton, D. Rancey, A. Robinson, and T. Lin. FIAT — fault injection based automated testing environment. In *Proceedings of the 18th International Sympo-*

sium on Fault-Tolerant Computing (FTCS-18), pages 102–107, Tokyo, Japan, June 1988.
364. B. Selic, G. Gullekson, and P. Ward. *Real-Time Object-Oriented Modeling*. John Wiley, 1994.
365. B. Shannon. Java 2 platform enterprise edition specification, 1.4 - proposed final draft 2. Technical report, Sun Microsystems, 2002.
366. M. Shaw and D. Garlan. *Software Architecture. Perspectives on an Emerging Discipline*. Prentice-Hall, 1996.
367. J. Skene and W. Emmerich. A model-driven approach to non-functional analysis of software architectures. In *Proceedings of 18th IEEE International Conference on Automated Software Engineering (ASE2003)*, pages 236–239, Montreal, Canada, October 6-10 2003.
368. D. J. Smith. *Achieving Quality Software (Third Edition)*. Chapman & Hall, 1995.
369. I. Sommerville. *Software Engineering*. Addison-Wesley, sixth edition, 2001.
370. N. Soundarajan and S. Fridella. Understanding OO frameworks and applications. *Informatica*, 25:297–308, 2001.
371. M. Sparling. Lessons learned through six years of component-based development. *Communications of the ACM*, 43(10):47–53, 2000.
372. O. Spinczyk, A. Gal, and W. Schrder-Preikschat. AspectC++: an aspect-oriented extension to C++. In *Proceedings of the 40th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS Pacific 2002)*, Sydney, Australia, Feb. 2002.
373. K. Sreenivasan and A. Kleinman. On the construction of a representative synthetic workload. *Communications of the ACM*, 17(3):127–133, Mar. 1974.
374. J. Stafford and L. Wolf. Annotating Components to Support Component-Based Static Analyses of Software Systems. Technical Report CU-CS-896-99, University of Colorado at Boulder, 1999.
375. J. A. Stafford and A. L. Wolf. Annotating components to support component-based static analyses of software systems. In *Proceedings of the Grace Hopper Celebration of Women in Computing*, 2001.
376. F. Stallinger, A. Dorling, T. Rout, B. Henderson-Sellers, and B. Lefever. Software process improvement for component-based software engineering: An introduction to the oospice project. In *Proceedings of the 28th EUROMICRO Conference (EUROMICRO'02)*, pages 318–323, Dortmund, Germany, Sept. 2002.
377. B. Subraya and S. Subrahmanya. Object driven performance testing of Web applications. In *Proceedings of the First Asia-Pacific Conference on Quality Software (APAQS'00)*, 2000.
378. G. Sullivan. Aspect-Oriented Programming using Reflection and Metaobject Protocols. *Communications of the ACM*, 44(10):95–97, 2001.
379. Sun Microsystem. Java message service specification. Technical report, Sun Microsystem Technical Report.
380. Sun Microsystems. Enterprise JavaBean Technology. `http://java.sun.com/products/ejb/`. Access date April 5th, 2004.
381. Sun Microsystems. `http://developer.java.sun.com/developer`, 2000.
382. Sun Microsystems. Java management extensions, instrumentation and agent specification, version 1.2. `http://java.sun.com/products/JavaManagement`, 2002.
383. Sun Microsystems. Java 2 platform, enterprise edition (J2EE). `http://java.sun.com/j2ee/`, 2003.

384. T. Systa, Y. Ping, and H. Muller. Analyzing java software by combining metrics and program visualization. In *Proceedings of the Fourth European Software Maintenance and Reengineering*, pages 199–208, Zurich, Switzerland, Mar. 2000.
385. C. Szyperski. *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley, New York, 1998.
386. C. Szyperski, D. Gruntz, and S. Murer. *Component Software - Beyond Object-Oriented Programming*. Addison-Wesley, second edition edition, 2002.
387. A. Taulavuori. Component documentation in the context of software product lines. Technical report, VTT Publications 484, VTT Technical Research Centre of Finland, Espoo, 2002.
388. A. Taulavuori, E. Niemelä, and P. Kallio. Component documentation - a key issue in software product lines. *Journal Information and Software Technology*, 2004.
389. R. N. Taylor, D. L. Levine, and C. D. Kelly. Structural testing of concurrent programs. *IEEE Transaction on Software Engineering*, 18(3):206–215, 1992.
390. Testing Technologies. Ttcn-3 tool series. `http://www.testing-technologies.de/products`, 2004.
391. The Apache Software Foundation. BCEL: Byte Code Engineering Library. `http://jakarta.apache.org/bcel`.
392. The Apache Software Foundation. Regexp. `http://jakarta.apache.org/regexp`.
393. TIBCO. The power of now. TIBCO hawk. `www.tibco.com/solutions/`.
394. A. Tikkala and M. Matinlassi. Platform services for wireless multimedia applications: case studies. In *1st International Conference on Mobile and Ubiquitous Multimedia, Oulu, Finland*, pages 76 – 81, 2002.
395. A. Ulrich and G. Chrobok-Diening. International workshop on testing distributed component-based systems. *SIGSOFT Softw. Eng. Notes*, 24(4):43–46, 1999.
396. H. Ural and B. Yang. Modeling software for accurate data flow representation. In *ICSE'93*, pages 277–286, May 1993.
397. D. Urting, Y. Berbers, S. V. Baelen, T. Holvoet, Y. Vandewoude, and P. Rigole. A tool for component based design of embedded software. In *Proceedings of the 40th International Conf. on Tools Pacific: Objects for internet, mobile and embedded applications - Volume 10*, February 2002.
398. R. van Renesse, K. P. Birman, and S. Maffeis. Horus: A flexible group communication system. In *Commnications of the ACM*, Apr. 1996.
399. M. Vidger and J. Dean. An architectural approach to building systems from cots software components. In *Proceedings of the 22nd Annual Software Engineering Workshop*, pages 99 – 131, Greenbelt, Maryland, 1997. National Research Council.
400. M. Vigder. An architecture for cots based software systems. Technical Report NRC Report No. 41603, National Research Council of Canada, 1998.
401. M. Vigder, T. McClean, and F. Bordeleau. Evaluating cots based architectures. In *Proceedings of the Second International Conference on COTS-Based Software Systems*, pages 240 – 250, 2003.
402. J. Vincent. Built-in test vade mecum part i - a common bit architecture, version 2.0. `http://www.component-plus.org`, 2002.
403. P. Vitharana. Risks and challenges of component-based software development. *Communications of the ACM*, 46(8):67–72, 2003.

404. K.-P. Vo, Y.-M. Wang, P. Chung, and Y. Huang. Xept: a software instrumentation method for exception handling. In *Proceedings of the Eighth International Symposium on Software Reliability Engineering*, pages 60–69, Albuquerque, NM, USA, Nov 1997.
405. J. Voas. Certifying off-the-shelf software components. *IEEE Computer*, 31(6):53–59, June 1998.
406. J. Voas. COTS software: The economical choice? *IEEE Software*, 15(2):16–19, 1998.
407. J. Voas. Maintaining component-based systems. *IEEE Software*, pages 22 – 27, 1998.
408. J. Voas. Developing a usage-based software certification process. *IEEE Computer*, 33(8):32–37, August 2000.
409. J. Voas, F. Charron, G. McGraw, K. Miller, and M. Friedman. Predicting how badly "good" software can behave. *IEEE Software*, 14(4):73–83, /1997.
410. J. Voas and J. Payne. Cots software failures: Can anything be done? In *Proceedings of the First IEEE Workshop on Application Specific Software Engineering and Technology (ASSET'98)*, pages 140–145. IEEE Press, Mar. 1998.
411. J. Voas and J. Payne. Dependability certification of software components. *The Journal of Systems and Software*, 52(2-3):165–172, 2000.
412. M. Volter, A. Schmid, and E. Wolff. *Server Component Patterns: Component Infrastructures Illustrated with EJB*. John Wiley, 2002.
413. K. Wallnau. Volume iii: A technology for predictable assembly from certifiable components. Technical Report CMU/SEI-2003-TR-009, Carnegie Mellon University, Pittsburgh, PA, 2003.
414. K. Wallnau, S. Hissam, and R. Seacord. *Building Systems from Commercial Components*. Addison-Wesley, 2002.
415. Y. Wang and G. King. A European COTS Architecture with Built-in Tests. In *Proceedings of the 8th International Conference on Object-Oriented Information Systems, Springer-Verlag LNCS 2425*, pages 336–347, 2002.
416. Y. Wang, G. King, and H. Wickburg. A method for built-in tests in component-based software maintenance. In *Proceedings of the IEEE International Conference on Software Maintenance and Reengineering*, pages 186–189, 1999.
417. Y. M. Wang, O. P. Damani, and W. J. Lee. Reliability and availability issues in distributed component object model (DCOM). In *Fourth International Workshop on Community Networking Proceedings*, pages 59–63, Atlanta,Georgia, Sept. 1997.
418. C. Warner. *Evaluation of Program Testing*. IBM Data Systems Division Development Laboratories, Poughkeepsie, N.Y., 1964.
419. E. Weyuker. The evaluation of program-based software test data adequacy criteria. *Communications of the ACM*, 31(6):668–675, June 1988.
420. E. Weyuker. Testing component-based software: A cautionary tale. *IEEE Software*, 15(5):54–59, 1998.
421. E. Weyuker and F. Vokolos. Experience with performance testing of software systems: issues, an approach, and case study. *IEEE Transactions on Software Engineering*, 26(12):1147–1156, 2000.
422. E. J. Weyuker. Axiomatizing software test data adequacy. *IEEE Transactions on Software Engineering*, 12(12):1128–1138, 1986.
423. E. J. Weyuker. Testing component-based software: A cautionary tale. *IEEE Software*, 15(5):54–59, 1998.

424. J. Whaley, M. C. Martin, and M. S. Lam. Automatic extraction of object-oriented component interfaces. In *Proceedings of ISSTA 2002*, pages 218–228, July 22-24 2002. Roma, Italy.
425. C. H. Wittenberg. Progress in testing component-based software (abstract only). In *Proceedings of the International Symposium on Software Testing and Analysis*, page 178. ACM Press, 2000.
426. Y. Wu, M. Chen, and J. Offutt. UML-based integration testing for component-based software. In *Int. Conf. on COTS-Based Software Sys.*, 2003.
427. G. Xing and M. R. Lyu. Testing, reliability, and interoperability issues in the corba programming paradigm. In *Proceedings of 1999 Asia-Pacific Software Engineering Conference (APSEC'99)*, pages 530–536, Takamatsu, Japan, Dec. 1999.
428. S. Yacoub, A. Mili, C. Kaveri, and M. Dehlin. Hierarchy of cots certification criteria. In P. Donohoe, editor, *Proceedings of the First Software Product Lines Conference*, pages 397 – 411, Boston, 2000. Kluwer Academic Publishers.
429. S. M. Yacoub, B. Cukic, and H. H. Ammar. A component-based approach to reliability analysis of distributed systems. In *Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems*, pages 158–167, Lausanne, Switzerland, Oct. 1999.
430. S. M. Yacoub, B. Cukic, and H. H. Ammar. Scenario-based reliability analysis of component-based software. In *Proceedings of 10th International Symposium on Software Reliability Engineering*, pages 22–31, Boca Raton,Florida, Nov. 1999.
431. S. S. Yau and B. Xia. Object-oriented distributed component software development based on corba. In *Proceedings of COMPSAC'98*, pages 246–251, Vienna, Austria, Aug. 1998.
432. M. Young and R. N. Taylor. Rethinking the taxonomy of fault detection techniques. In *Proceedings of the 11th International Conference on Software Engineering*, pages 53–62, May 1989.
433. Y. Yu and B. W. Johnson. A bbn approach to certifying the reliability of cots software systems. In *Proceedings of Annual Reliability and Maintainability Symposium*, pages 19–24, Tampa, Florida, Jan. 2003.
434. A. Zaremski and J. Wing. Specification matching of software components. *ACM Trans. on Software Engineering and Methodology*, 6(4), October 1997.
435. W. Zhao and C. Papachristou. Testing dsp cores based on self-test programs. In *Proceedings of the conference on Design, automation and test in Europe*, pages 166–172. IEEE Computer Society, 1998.
436. H. Zhu. Axiomatic assessment of control flow based software test adequacy criteria. *Software Engineering Journal*, Sept. 1995.
437. H. Zhu. A formal analysis of the subsume relation between software test adequacy criteria. *IEEE Transactions on Software Engineering*, 22(4):248–255, April 1996.
438. H. Zhu. A note on test oracles and semantics of algebraic specifications. In *QSIC'03*, pages 91–98, Dallas, USA, Oct. 2003.
439. H. Zhu and P. Hall. Test data adequacy measurement. *Software Engineering Journal*, 8(1):21–30, Jan. 1993.
440. H. Zhu, P. A. V. Hall, and J. H. R. May. Software unit test coverage and adequacy. *ACM Computing Surveys*, 29(4):366–427, 1997.

441. H. Zhu and X. He. A theory of behaviour observation in software testing. Technical Report CMS-TR-99-05, School of Computing and Mathematical Sciences, Oxford Brookes University, Oxford, UK, Sept. 1999.
442. H. Zhu and X. He. Constructions of behaviour observation schemes in software testing. In *HASE'00*, pages 2–12, New Mexico, Nov. 2000.
443. H. Zhu and X. He. A theory of testing high-level petri nets. In *Proc. of the IFIP 16th World Computer Congress*, pages 443–450, Beijing, China, Aug. 2000.
444. H. Zhu and X. He. An observational theory of integration testing for component-based software development. In *COMPSAC'2001*, pages 363–368, Chicago, Illinois, USA, Oct. 2001.
445. H. Zhu and X. He. A methodology of testing high-level petri nets. *Information and Software Technology*, 44(8):473–489, June 2002.
446. H. Zhu, L. Jin, and D. Diaper. Application of task analysis to the validation of software requirements. In *SEKE'99*, pages 239–245, Kaiserslautern, Germany, June 1999.
447. H. Zhu, L. Jin, D. Diaper, and G. Bai. Software requirements validation via task analysis. *Journal of System and Software*, 61(2):145–169, March 2002.

Index

Abstraction, 3, 216
Abstraction level, 3
Adequacy criterion, 27
Adequacy measurement, 246
Analysis, 274, 277, 284, 288, 289, 291
Architectural mismatch, 21
Architecture, 273–276, 278, 281, 284, 287, 289–291
Aspects, 74
Attachment, 215
Automatic code generation, 195
Automation, 349
Axioms of Behavioral Observations, 246

BIT component, 198
Branch testing, 241
Built-in contract testing, 196, 197
Built-in test, 55, 59

CDT framework, 179
Class state machine, 366
COM, 10, 319
Commercial-off-the-shelf, 14, 57, 315, 364
Complete partially ordered sets (CPO sets), 242
Component, 364
Component analysis, 17
Component container, 10
Component context, 197
Component contract, 197
Component definition, 5
Component deployment testing, 172
Component framework, 9
Component from external sources, 14, 18
Component instance, 5
Component interface, 364
Component model, 8
Component Object Model, 10
Component persistence, 5
Component produced by contract, 14, 18
Component produced in-house, 14, 18
Component provider perspective, 19, 365
Component specification variation, 227
Component state machine, 366
Component stub, 265
Component type, 5
Component user perspective, 19, 365
Component-based software development, 239, 315
Component-based software system, 320
Composability, 56, 70
Configuration, 197
Contract, 197
Contract checking, 191
Contract testing, 55, 56
Controllability, 28
CORBA, 318, 364
CORBA Component Model, 10
Cost-effectiveness, 1
COTS, 364
COTS component, 14
COTS Components familiarization, 21
CSM, 366

DCOM, 319, 364
Dependability, 349
Design by Contract, 34
Design pattern, 3
Distributed Objects, 314
Distributed systems, 314

Enterprise JavaBeans, 10, 319, 368
Entity bean, 11
Exception injection, 91
Exceptions, 89
Explicit server, 199
External quality, 24
Extraction Relation Between Schemes, 248

Failure atomicity, 89
Fault detection ability, 249
Fault injection, 92, 349
Fault propagation, 29
Fault-tolerance, 349
Functional specification, 34

Glueware, 17

Hook methods, 34

Implicit server, 199
Incremental testing, 247
Independent commercial item, 14, 18
Infection, 30
Infinite test, 243
Information flow testing, 261
Input inconsistency, 27
Integration strategy, 265
  bottom-up, 265
  Top-down, 266
Integration testing, 239, 366
  Generalization of WIT methods, 263
  Heterogeneous WIT testing, 264
  Hierarchical, 261
  Higher order WIT methods, 263
  Incremental, 265
  White-box (WIT), 255
Interaction parameter testing, 258
Interaction sequence testing, 260
Interaction specification, 34
Interaction statement testing, 257
Interface probing, 27

Internal quality, 24
Introspection, 7

JavaBeans, 319
JUnit, 203

Limited exchange of information, 16
Live sequence charts, 274, 276, 281, 291
Logical state, 198

Message-driven bean, 11
Meta-data, 72, 75
Meta-information, 7, 17
Meta-object, 83
Middleware, 314, 349, 365
Model checker, 274, 284, 287, 289–291
Model coverage, 200
Model Driven Architecture, 196
Mutation score, 253
Mutation testing, 241, 245

Object-oriented framework, 33
Observation scheme, 243
Oracle, 366

Paragraph, 219
Path testing, 241
Polymorphism, 34
Prediction, 58, 60, 70
Product line, 2
Prototyping, 21, 26
Publish/subscribe, 273–277, 282, 289, 290

Quality, 23
Quality assurance, 320
Quality attribute, 25
Quality characteristic, 23
Quality in use, 25
Quality indicator, 25
Quality metric, 25
Quality prediction model, 338

Reflection, 7, 85
Reliability, 349
Reuse, 1
Reverse engineering, 27
Risk management, 26
Robustness, 349

Security, 349
Session bean, 10
Software architecture, 314
Software metrics, 328
Software performance, 314
Software performance evaluation, 314
Software performance testing, 314
Software testing, 349
Special version of a commercial item, 14, 18
SPIN, 273–275, 284, 285, 288–291
Statement coverage, 253
Statement testing, 241
Subsumption relation, 249
System configuration, 197
System under test, 201

Template methods, 33
Test adequacy criteria, 245
Test architecture, 201
Test behavior, 201
Test case, 202
Test data, 201
Test driver, 265
Test models, 200
Test specification, 201
Test validation, 202
Test verdict, 202
Tester component, 197
Testing, 23
Testing communication protocols, 241
Testing component, 199
Testing concurrent systems, 241
Testing criteria, 200
Testing interface, 197
Testing methods, 246
  Design based, 240
  Design patterns, 250
  High order WIT, 263
  Program based, 240
  Specification based, 240
Testing profile, 196
Trace, 34
Tracing, 29
TTCN-3, 196, 203, 209

UML, 58, 60, 61, 70
UML testing profile, 196, 201
UML-based testing, 200
Universe of phenomena, 242
  Case-wise statistical construction, 254
  Design patterns, 250
  Partially ordered set (poset) construction, 251
  Product construction, 252
  Set construction, 250
  Statistical construction, 253

Validation, 273–278, 281, 287–289, 291
Virtual component, 180

Wrapper, 349

Printing: Mercedes-Druck, Berlin
Binding: Stein + Lehmann, Berlin