

Cover Page



Universiteit Leiden



The handle <http://hdl.handle.net/1887/45620> holds various files of this Leiden University dissertation

Author: Nobakht, Behrooz

Title: Actors at work

Issue Date: 2016-12-15

Bibliography

- [1] Parker Abercrombie and Murat Karaorman. “jContractor: Bytecode Instrumentation Techniques for Implementing Design by Contract in Java”. In: *Electr. Notes Theor. Comput. Sci.* 70.4 (2002), pp. 55–79 (cit. on p. 92).
- [2] L. Aceto, M. Cimini, A. Ingólfssdóttir, et al. “Modelling and Simulation of Asynchronous Real-Time Systems using Timed Rebeca”. In: *FOCLASA*. 2011, pp. 1–19 (cit. on p. 42).
- [3] G. Agha, I. Mason, S. Smith, and C. Talcott. “A Foundation for Actor Computation”. In: *Journal of Functional Programming* 7 (1997), pp. 1–72 (cit. on pp. 3, 48, 59).
- [4] Gul Agha. “Actors: a model of concurrent computation in distributed systems”. PhD thesis. MIT, 1986 (cit. on pp. 3, 26).
- [5] Gul Agha. “Online Efficient Predictive Safety Analysis of Multithreaded Programs”. In: () (cit. on p. 4).
- [6] Gul Agha. “The Structure and Semantics of Actor Languages”. In: *Proc. the REX Workshop*. 1990, pp. 1–59 (cit. on pp. 48, 59).
- [7] Elvira Albert, Puri Arenas, Antonio Flores-Montoya, et al. “SACO: Static Analyzer for Concurrent Objects”. In: *Tools and Algorithms for the Construction and Analysis of Systems - 20th International Conference, TACAS 2014, ETAPS 2014*. 2014, pp. 562–567 (cit. on p. 5).
- [8] Chris Allan, Pavel Avgustinov, Aske Simon Christensen, et al. “Adding trace matching with free variables to AspectJ”. In: *Proceedings of the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications. OOPSLA '05*. 2005, pp. 345–364 (cit. on p. 92).
- [9] R. Alur and P. Madhusudan. “Adding nesting structure to words”. In: *J. ACM* 56 (3 2009), 16:1–16:43 (cit. on p. 94).
- [10] Rajeev Alur and David L Dill. “A theory of timed automata”. In: *Theoretical computer science* 126.2 (1994), pp. 183–235 (cit. on pp. 96, 97, 104).
- [11] G.R. Andrews. *Foundations of Multithreaded, Parallel, and Distributed Programming*. Addison-Wesley, 2000 (cit. on p. 16).
- [12] Joe Armstrong. “Erlang”. In: *Communications of ACM* 53.9 (2010), pp. 68–75 (cit. on pp. 48, 49).
- [13] Joe Armstrong. *Programming Erlang: Software for a Concurrent World*. Pragmatic Bookshelf, 2007 (cit. on pp. 9, 26, 29, 43).

- [14] Michael Baker and Martin Thompson. *LMAX Disruptor*. <http://github.com/LMAX-Exchange/disruptor>. LMAX Exchange (cit. on p. 50).
- [15] Howard Barringer, Allen Goldberg, Klaus Havelund, and Koushik Sen. “Rule-Based Runtime Verification”. In: *Verification, Model Checking, and Abstract Interpretation, 5th International Conference (VMCAI’04), Proceedings*. Ed. by Bernhard Steffen and Giorgio Levi. Vol. 2937. Lecture Notes in Computer Science. Springer, 2004, pp. 44–57 (cit. on p. 93).
- [16] Detlef Bartetzko, Clemens Fischer, Michael Möller, and Heike Wehrheim. “Jass - Java with Assertions”. In: *Electr. Notes Theor. Comput. Sci.* 55.2 (2001), pp. 103–117 (cit. on p. 92).
- [17] Gerd Behrmann, Alexandre David, and Kim G Larsen. “A tutorial on uppaal”. In: *Formal methods for the design of real-time systems*. Springer, 2004, pp. 200–236 (cit. on pp. 104, 109).
- [18] Joakim Bjørk, Frank S de Boer, Einar Broch Johnsen, Rudolf Schlatter, and S Lizeth Tapia Tarifa. “User-defined schedulers for real-time concurrent objects”. In: *Innovations in Systems and Software Engineering* 9.1 (2013), pp. 29–43 (cit. on pp. 5, 43, 96, 98).
- [19] E. Bodden. “J-lo, a tool for runtime-checking temporal assertions”. In: *Master Thesis, RWTH Aachen University* (2005) (cit. on p. 92).
- [20] Frank S. de Boer, Marcello B. Bonsangue, Andreas Gruner, and Martin Steffen. “Java Test Driver Generation from Object-Oriented Interaction Traces”. In: *Electronic Notes in Theoretical Computer Science* 243 (2009), pp. 33–47 (cit. on p. 94).
- [21] Frank S. de Boer, Dave Clarke, and Einar Broch Johnsen. “A Complete Guide to the Future”. In: *ESOP*. 2007, pp. 316–330 (cit. on p. 30).
- [22] Frank S. de Boer, Mohammad Mahdi Jaghoori, and Einar Broch Johnsen. “Dating Concurrent Objects: Real-Time Modeling and Schedulability Analysis”. In: *CONCUR 2010*. Vol. 6269. 2010, pp. 1–18 (cit. on p. 27).
- [23] Grady Booch. “Object-oriented design”. In: *ACM SIGAda Ada Letters* 1.3 (1982), pp. 64–76 (cit. on p. 3).
- [24] Gilad Bracha, Martin Odersky, David Stoutamire, and Philip Wadler. “Making the future safe for the past: Adding genericity to the Java programming language”. In: *Acm sigplan notices* 33.10 (1998), pp. 183–200 (cit. on p. 5).
- [25] Konstantinos Bratanis, Dimitris Dranidis, and Anthony J. H. Simons. “Towards Run-Time Monitoring of Web Services Conformance to Business-Level Agreements”. In: ed. by Leonardo Bottaci and Gordon Fraser. Vol. 6303. Springer, 2010, pp. 203–206 (cit. on p. 96).
- [26] Eric Bruneton, Romain Lenglet, and Thierry Coupaye. “ASM: a code manipulation tool to implement adaptable systems”. In: *Adaptable and extensible component systems* 30 (2002) (cit. on p. 10).
- [27] Richard Bubel, Antonio Flores-Montoya, and Reiner Hähnle. “Analysis of Executable Software Models”. In: *SFM 2014, Bertinoro, Italy, June 16-20, 2014, Advanced Lectures*. 2014, pp. 1–25 (cit. on p. 97).

- [28] Po-Hao Chang and Gul Agha. “Towards Context-Aware Web Applications”. In: *7th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS)*. 2007, pp. 239–252 (cit. on p. 48).
- [29] Feng Chen and Grigore Rosu. “Java-MOP: A Monitoring Oriented Programming Environment for Java”. In: *Tools and Algorithms for the Construction and Analysis of Systems, 11th International Conference, Proceedings*. Ed. by Nicolas Halbwachs and Lenore D. Zuck. Vol. 3440. Lecture Notes in Computer Science. Springer, 2005, pp. 546–550 (cit. on pp. 88–90, 93).
- [30] Feng Chen and Grigore Roşu. “MOP: an efficient and generic runtime verification framework”. In: *Proceedings of the 22nd annual ACM SIGPLAN conference on Object-oriented programming systems and applications*. OOPSLA ’07. Montreal, Quebec, Canada, 2007, pp. 569–588 (cit. on pp. 88, 93).
- [31] Yuan Chen, Subu Iyer, Xue Liu, Dejan Milojicic, and Akhil Sahai. “SLA decomposition: Translating service level objectives to system level thresholds”. In: *Autonomic Computing, 2007. ICAC’07. Fourth International Conference on*. IEEE. 2007, pp. 3–3 (cit. on p. 97).
- [32] Yoonsik Cheon and Ashaveena Perumandla. “Specifying and checking method call sequences of Java programs”. In: *Software Quality Journal* 15 (1 Mar. 2007), pp. 7–25 (cit. on p. 92).
- [33] Elaine Cheong, Edward A. Lee, and Yang Zhao. “Viptos: a graphical development and simulation environment for tinyOS-based wireless sensor networks”. In: *Proc. Embedded net. sensor sys., SenSys 2005*. 2005, pp. 302–302 (cit. on p. 48).
- [34] Ben Christensen. *RxJava: Reactive Functional Programming in Java*. <http://github.com/Netflix/RxJava/wiki>. Netflix (cit. on p. 50).
- [35] Sylvan Clebsch and Sophia Drossopoulou. “Fully concurrent garbage collection of actors on many-core machines”. In: 2013, pp. 553–570 (cit. on p. 9).
- [36] Parallel Universe Co. *Quasar*. <https://github.com/puniverse/quasar> (cit. on p. 9).
- [37] Andrew Coles, Amanda Jane Coles, Allan Clark, and Stephen Gilmore. “Cost-Sensitive Concurrent Planning Under Duration Uncertainty for Service-Level Agreements.” In: *ICAPS*. 2011 (cit. on p. 98).
- [38] Marco Comuzzi, Constantinos Kotsokalis, George Spanoudakis, and Ramin Yahyapour. “Establishing and monitoring SLAs in complex service based systems”. In: *Web Services, 2009. ICWS 2009. IEEE International Conference on*. IEEE. 2009, pp. 783–790 (cit. on p. 97).
- [39] *Concurrent Haskell*. http://en.wikipedia.org/wiki/Concurrent_Haskell (cit. on p. 9).
- [40] *Concurrenty in Rust Language*. <https://doc.rust-lang.org/book/concurrency.html> (cit. on p. 9).
- [41] Melvin E Conway. “Design of a separable transition-diagram compiler”. In: *Communications of the ACM* 6.7 (1963), pp. 396–408 (cit. on p. 4).
- [42] “Coordination Models and Languages”. In: vol. 4467. Springer Berlin / Heidelberg, 2007. Chap. Actors That Unify Threads and Events, pp. 171–190 (cit. on pp. 26, 43).

- [43] Fábio Corrêa. “Actors in a new “highly parallel” world”. In: *Proc. Warm Up Workshop for ACM/IEEE ICSE 2010*. WUP '09. ACM, 2009, pp. 21–24 (cit. on pp. 26, 43).
- [44] Ole-Johan Dahl, Bjørn Myhrhaug, and Kristen Nygaard. “Some Features of the SIMULA 67 Language”. In: *Proceedings of the Second Conference on Applications of Simulations*. New York, New York, USA: Winter Simulation Conference, 1968, pp. 29–31 (cit. on pp. 3, 4).
- [45] Markus Dahm. “Byte code engineering”. In: *JIT'99*. Springer, 1999, pp. 267–277 (cit. on pp. 6, 10).
- [46] Frank S De Boer, Dave Clarke, and Einar Broch Johnsen. “A complete guide to the future”. In: *Programming Languages and Systems*. Springer, 2007, pp. 316–330 (cit. on p. 4).
- [47] Crystal Chang Din, Richard Bubel, and Reiner Hähnle. “KeY-ABS: A Deductive Verification Tool for the Concurrent Modelling Language ABS”. In: *Automated Deduction - CADE-25 - 25th International Conference on Automated Deduction*. 2015, pp. 517–526 (cit. on p. 5).
- [48] *Eclipse Debug Platform*. <http://www.eclipse.org/eclipse/debug/> (cit. on p. 94).
- [49] J. Eker, J.W. Janneck, E.A. Lee, et al. “Taming heterogeneity - the Ptolemy approach”. In: *Proceedings of the IEEE 91.1 ()*, pp. 127–144 (cit. on p. 43).
- [50] *Elixir Agents*. <http://elixir-lang.org/getting-started/mix-otp/agent.html> (cit. on p. 9).
- [51] *Erlang Concurrent Programming*. <http://www.erlang.org/course/concurrent-programming> (cit. on p. 9).
- [52] Kiko Fernandez-Reyes, Einar Broch Johnsen, Ka I Pun, et al. “Parallel Objects for Multicores: A Glimpse at the Parallel Language ENCORE”. In: *Formal Methods for Multicore Programming, SFM 9104* (2015), p. 1 (cit. on p. 6).
- [53] Elena Fersman, Pavel Krcal, Paul Pettersson, and Wang Yi. “Task automata: Schedulability, decidability and undecidability”. In: *Information and Computation* 205.8 (2007), pp. 1149–1172 (cit. on pp. 10, 96, 97, 104, 108).
- [54] Elena Fersman, Leonid Mokrushin, Paul Pettersson, and Wang Yi. “Schedulability Analysis Using Two Clocks”. In: vol. 2619. *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2003, pp. 224–239 (cit. on p. 44).
- [55] Martin Fowler. *LMAX Architecture*. <http://martinfowler.com/articles/lmax.html>. Martin Fowler. July 2011 (cit. on p. 50).
- [56] Gamma, Erich and Helm, Richard and Johnson, Ralph and Vlissides, John. “Design Patterns: Abstraction and Reuse of Object-Oriented Design”. In: *ECOOP '93 – Object-Oriented Programming*. Vol. 707. *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 1993, pp. 406–431 (cit. on p. 47).
- [57] Elena Giachino, Carlo A. Grazia, Cosimo Laneve, Michael Lienhardt, and Peter Y. H. Wong. “Deadlock Analysis of Concurrent Objects: Theory and Practice”. In: *IFM*. 2013, pp. 394–411 (cit. on pp. 5, 63).
- [58] Stephen Gilmore, László Gönczy, Nora Koch, et al. “Non-functional properties in the model-driven development of service-oriented systems”. In: *Software and Systems Modeling* 10.3 (2011), pp. 287–311 (cit. on p. 98).

- [59] *Go Statement Specification*. https://golang.org/ref/spec#Go_statements (cit. on p. 9).
- [60] Brian Goetz. *JSR 335, Lambda Expressions for the Java Programming Language*. <http://jcp.org/en/jsr/detail?id=335>. Oracle. Mar. 2013 (cit. on pp. 6, 53, 54).
- [61] Brian Goetz. *Lambda: A Peek Under The Hood*. JAX London. Oracle. 2012 (cit. on p. 55).
- [62] Brian Goetz. *Lambda Expression Translation in Java 8*. <http://cr.openjdk.java.net/~briangoetz/lambda/lambda-translation.html>. Oracle. Mar. 2013 (cit. on pp. 10, 54).
- [63] James Gosling. *The Java language specification*. Addison-Wesley Professional, 2000 (cit. on p. 5).
- [64] James Gosling, Bill Joy, Guy Steele, Gilad Bracha, and Alex Buckley. *The Java Language Specification, Java SE 8 Edition*. <https://docs.oracle.com/javase/specs/jls/se8/html/index.html>. 2015 (cit. on p. 6).
- [65] S. de Gouw, J. Vinju, and F. S. de Boer. “Prototyping a tool environment for run-time assertion checking in JML with Communication Histories”. In: FTFJP ’10. ACM, 2010, 6:1–6:7 (cit. on p. 92).
- [66] Scott F. Smith Gul A. Agha Ian A. Mason and Carolyn L. Talcott. “A foundation for actor computation”. In: *Journal of Functional Programming* 7 (1997), pp. 1–72 (cit. on p. 29).
- [67] Reiner Hähnle, Michiel Helvensteijn, Einar Broch Johnsen, et al. “HATS Abstract Behavioral Specification: The Architectural View”. In: *FMCO*. 2011, pp. 109–132 (cit. on pp. 5, 63).
- [68] Philipp Haller. “On the integration of the actor model in mainstream technologies: the Scala perspective”. In: *Proceedings of the 2nd edition on Programming systems, languages and applications based on actors, agents, and decentralized control abstractions*. ACM. 2012, pp. 1–6 (cit. on p. 50).
- [69] Philipp Haller and Martin Odersky. “Scala Actors: Unifying thread-based and event-based programming”. In: *Theoretical Computer Science* 410.2-3 (2009), pp. 202–220 (cit. on pp. 9, 26, 29, 43, 48, 49).
- [70] K. Havelund and G. Rosu. “Monitoring Java programs with Java PathExplorer”. In: *Runtime Verification (in connection with CAV ’01)* 55.2 of ENTCS (2001), pp. 200–217 (cit. on p. 93).
- [71] Johannes Henkel and Amer Diwan. “Discovering algebraic specifications from Java classes”. In: *ECOOP 2003–Object-Oriented Programming*. Springer, 2003, pp. 431–456 (cit. on p. 5).
- [72] Maurice P. Herlihy and Jeannette M. Wing. “Linearizability: A Correctness Condition for Concurrent Objects”. In: *ACM Trans. Program. Lang. Syst.* 12.3 (July 1990), pp. 463–492 (cit. on p. 4).
- [73] Carl Hewitt. “Procedural Embedding of knowledge in Planner”. In: *Proc. the 2nd International Joint Conference on Artificial Intelligence*. 1971, pp. 167–184 (cit. on p. 48).

- [74] Carl Hewitt. “What Is Commitment? Physical, Organizational, and Social (Revised)”. In: *Proc. Coordination, Organizations, Institutions, and Norms in Agent Systems II*. LNCS Series. Springer, 2007, pp. 293–307 (cit. on p. 48).
- [75] C. A. R. Hoare. “Monitors: An Operating System Structuring Concept”. In: *Commun. ACM* 17.10 (1974), pp. 549–557 (cit. on p. 4).
- [76] Giles Hogben and Alain Pannetrat. “Mutant Apples: A Critical Examination of Cloud SLA Availability Definitions”. In: *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*. Vol. 1. IEEE. 2013, pp. 379–386 (cit. on pp. 97, 109).
- [77] Inzinger, Christian and Hummer, Waldemar and Satzger, Benjamin and Leitner, Philipp and Dustdar, Shahram. “Generic event-based monitoring and adaptation methodology for heterogeneous distributed systems”. In: *Software – Practice and Experience* (2014) (cit. on p. 97).
- [78] Mohammad Mahdi Jaghoori. “Composing Real-Time Concurrent Objects Refinement, Compatibility and Schedulability”. In: *Fundamentals of Software Engineering*. Springer Berlin Heidelberg, 2012, pp. 96–111 (cit. on p. 104).
- [79] Mohammad Mahdi Jaghoori. “Time at your service: schedulability analysis of real-time and distributed services”. PhD thesis. Leiden University, 2010 (cit. on pp. 96, 104–107, 109).
- [80] Mohammad Mahdi Jaghoori, Frank S. de Boer, Tom Chothia, and Marjan Sirjani. “Schedulability of asynchronous real-time concurrent objects”. In: *J. Log. Algebr. Program.* 78.5 (2009), pp. 402–416 (cit. on p. 63).
- [81] Mohammad Mahdi Jaghoori, Ali Movaghar, and Marjan Sirjani. “Modere: the model-checking engine of Rebeca”. In: *Proc. 21st ACM Symposium on Applied Computing*. 2006, pp. 1810–1815 (cit. on p. 28).
- [82] *Java 5 Annotations*. <http://java.sun.com/j2se/1.5.0/docs/guide/language/annotations.html> (cit. on pp. 73, 77).
- [83] JCP. *RTSJ v1 JSR 1*. <http://jcp.org/en/jsr/detail?id=1>. 1998 (cit. on pp. 29, 44).
- [84] JCP. *RTSJ v1.1 JSR 282*. <http://jcp.org/en/jsr/detail?id=282>. 2005 (cit. on pp. 29, 44).
- [85] Einar Broch Johnsen. “History-Based Specification and Verification of Scalable Concurrent and Distributed Systems”. In: *Formal Methods and Software Engineering: 17th International Conference on Formal Engineering Methods, ICFEM 2015, Paris, France, November 3-5, 2015, Proceedings*. Vol. 9407. Springer. 2015, p. 217 (cit. on p. 4).
- [86] Einar Broch Johnsen, Reiner Hähnle, Jan Schäfer, Rudolf Schlatter, and Martin Steffen. “ABS: A Core Language for Abstract Behavioral Specification”. In: *Formal Methods for Components and Objects*. Vol. 6957. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, pp. 142–164 (cit. on pp. 5, 42, 43).
- [87] Einar Broch Johnsen, Reiner Hähnle, Jan Schäfer, Rudolf Schlatter, and Martin Steffen. “ABS: A core language for abstract behavioral specification”. In: *Formal Methods for Components and Objects*. Springer. 2012, pp. 142–164 (cit. on pp. 5, 7, 10, 49, 52, 97, 100).

- [88] Einar Broch Johnsen and Olaf Owe. “An Asynchronous Communication Model for Distributed Concurrent Objects”. In: *Software and Systems Modeling* 6.1 (2007), pp. 39–58 (cit. on pp. 5, 16, 17, 29).
- [89] Einar Broch Johnsen, Rudolf Schlatte, and Silvia Lizeth Tapia Tarifa. “Modeling resource-aware virtualized applications for the cloud in Real-Time ABS”. In: *Formal Methods and Software Engineering*. Springer, 2012, pp. 71–86 (cit. on pp. 5, 63, 98, 100, 104).
- [90] *JPDA Reference Home Page*. <http://java.sun.com/javase/technologies/core/toolsapis/jpda/> (cit. on pp. 69, 77).
- [91] *JSR 166: Java concurrency utilities*. <http://www.jcp.org/jsr/detail/166.jsp> (cit. on pp. 5, 6, 10, 56, 59).
- [92] *JUnit Test Framework*. <http://www.junit.org/> (cit. on p. 83).
- [93] *JUnit Tutorial in Eclipse*. <http://www.vogella.com/articles/JUnit/article.html> (cit. on p. 83).
- [94] Rajesh K. Karmani, Amin Shali, and Gul Agha. “Actor frameworks for the JVM platform: a comparative analysis”. In: *Proc. Principles and Practice of Prog. in Java (PPPJ’09)*. ACM, 2009, pp. 11–20 (cit. on pp. 3, 5, 9, 26, 43, 48, 50, 58).
- [95] Alan Kay. *Alan Kay*. <http://c2.com/cgi/wiki?AlanKayOnMessaging> (cit. on p. 3).
- [96] Alan Kay. *Dr. Alan Kay on the Meaning of Object-Oriented Programming*. http://userpage.fu-berlin.de/~ram/pub/pub_jf47ht81Ht/doc_kay_oop_en (cit. on p. 3).
- [97] Keven T Kearney, Francesco Torelli, and Constantinos Kotsokalis. “SLA*: An abstract syntax for Service Level Agreements”. In: *Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on*. IEEE, 2010, pp. 217–224 (cit. on p. 110).
- [98] Alexander Keller and Heiko Ludwig. “The WSLA framework: Specifying and monitoring service level agreements for web services”. In: *Journal of Network and Systems Management* 11.1 (2003), pp. 57–81 (cit. on p. 97).
- [99] Gregor Kiczales, Erik Hilsdale, Jim Hugunin, et al. “Getting started with ASPECTJ”. In: *Commun. ACM* 44 (10 Oct. 2001), pp. 59–65 (cit. on pp. 71, 89, 92).
- [100] Moonzoo Kim, Mahesh Viswanathan, Sampath Kannan, Insup Lee, and Oleg Sokolsky. “Java-MaC: A Run-Time Assurance Approach for Java Programs”. In: *Form. Methods Syst. Des.* 24 (2 Mar. 2004), pp. 129–155 (cit. on p. 93).
- [101] D. E. Knuth. “Semantics of Context-Free Languages”. In: *Mathematical Systems Theory* 2.2 (1968), pp. 127–145 (cit. on p. 92).
- [102] Donald Knuth. *The Art of Computer Programming*. Vol. Volume 1: Fundamental Algorithms. Addison-Wesley, 1968 (cit. on p. 4).
- [103] Kim Guldstrand Larsen, Fredrik Larsson, Paul Pettersson, and Wang Yi. “Efficient verification of real-time systems: compact data structure and state-space reduction”. In: *Real-Time Systems Symposium, 1997. Proceedings., The 18th IEEE*. IEEE, 1997, pp. 14–24 (cit. on p. 104).

- [104] Doug Lea. “A Java fork/join framework”. In: *ACM Java Grande*. 2000, pp. 36–43 (cit. on p. 5).
- [105] Doug Lea. *Concurrent programming in Java - design principles and patterns*. Addison-Wesley-Longman, 1997 (cit. on p. 4).
- [106] G. T. Leavens, A. L. Baker, and C. Ruby. “Preliminary design of JML: a behavioral interface specification language for Java”. In: *ACM SIGSOFT Software Engineering Notes* 31 (3 May 2006), pp. 1–38 (cit. on p. 91).
- [107] Edward A. Lee, Xiaojun Liu, and Stephen Neuendorffer. “Classes and inheritance in actor-oriented design”. In: *ACM Transactions in Embedded Computing Systems* 8.4 (2009) (cit. on p. 48).
- [108] Edward A. Lee, Stephen Neuendorffer, and Michael J. Wirthlin. “Actor-Oriented Design of Embedded Hardware and Software Systems”. In: *Journal of Circuits, Systems, and Computers* 12.3 (2003), pp. 231–260 (cit. on pp. 43, 48).
- [109] Xavier Leroy. “Java bytecode verification: algorithms and formalizations”. In: *Journal of Automated Reasoning* 30.3-4 (2003), pp. 235–269 (cit. on pp. 6, 10).
- [110] Xavier Leroy. “Java bytecode verification: an overview”. In: *Computer aided verification*. Springer. 2001, pp. 265–285 (cit. on pp. 6, 10).
- [111] Jed Liu, Aaron Kimball, and Andrew C. Myers. “Interruptible Iterators”. In: (2006) (cit. on p. 6).
- [112] Xavier Logean, Falk Dietrich, Hayk Karamyan, and Shawn Koppenhöfer. “Run-time monitoring of distributed applications”. In: *Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing*. Middleware ’98. 1998, pp. 459–474 (cit. on p. 96).
- [113] K Lundin. “Inside the Erlang VM, focusing on SMP”. http://erlang.org/euc/08/euc_smp.pdf. Nov. 2008 (cit. on pp. 26, 43).
- [114] Khaled Mahbub, George Spanoudakis, and Theodoris Tsigkritis. “Translation of SLAs into monitoring specifications”. In: *Service Level Agreements for Cloud Computing*. Springer, 2011, pp. 79–101 (cit. on p. 97).
- [115] Cláudio Maia, Luís Nogueira, and Luís Miguel Pinho. “Combining RTSJ with Fork/Join: a priority-based model”. In: *Proceedings of the 9th International Workshop on Java Technologies for Real-Time and Embedded Systems*. JTRES ’11. York, United Kingdom: ACM, 2011, pp. 82–86 (cit. on p. 43).
- [116] Michael Martin, Benjamin Livshits, and Monica S. Lam. “Finding application errors and security flaws using PQL: a program query language”. In: *Proceedings of the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*. OOPSLA ’05. 2005, pp. 365–383 (cit. on p. 92).
- [117] Nicholas D Matsakis and Felix S Klock II. “The rust language”. In: *Proceedings of the 2014 ACM SIGAda annual conference on High integrity language technology*. ACM. 2014, pp. 103–104 (cit. on p. 9).
- [118] *Maven JUnit Integration with Surefire plugin*. <http://maven.apache.org/surefire/maven-surefire-plugin/examples/junit.html> (cit. on p. 83).
- [119] Meyer, B. “Applying “design by contract””. In: *Computer* 25.10 (Oct. 1992), pp. 40–51 (cit. on p. 48).

- [120] B. Meyer. *Object-Oriented Software Construction*, 2nd edition. Prentice Hall, New Jersey, 2000 (cit. on pp. 3, 92).
- [121] Microsoft. *Reactive Extensions*. <https://rx.codeplex.com/>. Microsoft (cit. on p. 50).
- [122] MOP: Monitoring-oriented programming. <http://fsl.cs.uiuc.edu/index.php/MOP> (cit. on pp. 88, 93).
- [123] Brian Nielsen and Gul Agha. "Semantics for an Actor-Based Real-Time Language". In: *In Fourth International Workshop on Parallel and Distributed Real-Time Systems*. 1996 (cit. on p. 42).
- [124] Behrooz Nobakht, Frank S de Boer, and Mohammad Mahdi Jaghoori. "The Future of a Missed Deadline". In: *Coordination Models and Languages*. Springer. 2013, pp. 181–195 (cit. on pp. 96, 105).
- [125] Behrooz Nobakht, Frank S. de Boer, Mohammad Mahdi Jaghoori, and Rudolf Schlatte. "Programming and deployment of active objects with application-level scheduling". In: *Proceedings of the 27th Annual ACM Symposium on Applied Computing*. SAC '12. ACM, 2012, pp. 1883–1888 (cit. on pp. 42, 59, 105).
- [126] Nathaniel Nystrom, Michael R Clarkson, and Andrew C Myers. "Polyglot: An extensible compiler framework for Java". In: *Compiler Construction*. Springer. 2003, pp. 138–152 (cit. on p. 5).
- [127] ObjectWeb. ASM. <http://asm.ow2.org/>. ObjectWeb (cit. on p. 10).
- [128] Martin Odersky, Philippe Altherr, Vincent Cremet, et al. *The Scala language specification*. 2004 (cit. on p. 6).
- [129] Martin Odersky and Philip Wadler. "Pizza into Java: Translating theory into practice". In: *Proceedings of the 24th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. ACM. 1997, pp. 146–159 (cit. on p. 5).
- [130] Terence Parr. ANTLR. <http://antlr.org/> (cit. on p. 38).
- [131] Michael Philippsen. "A survey of concurrent object-oriented languages". In: *Concurrency - Practice and Experience* 12.10 (2000), pp. 917–980 (cit. on pp. 3, 8).
- [132] Lee Pike. *Copilot: A stream language for generating hard real-time C code*. <http://leepike.github.com/Copilot/> (cit. on p. 94).
- [133] Gordon D Plotkin. "The origins of structural operational semantics". In: *The Journal of Logic and Algebraic Programming* 60-61 (2004), pp. 3–15 (cit. on pp. 5, 30).
- [134] *Pony Language*. <http://www.ponylang.org/>. 2014–2016 (cit. on p. 9).
- [135] Franco Raimondi, James Skene, and Wolfgang Emmerich. "Efficient online monitoring of web-service SLAs". In: *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*. ACM. 2008, pp. 170–180 (cit. on p. 97).
- [136] Henrique Rebêlo, Ricardo Massa Ferreira Lima, Márcio Cornélio, and Sérgio Soares. "A JML Compiler Based on AspectJ". In: *Proceedings of the First International Conference on Software Testing, Verification, and Validation (ICST'08)*. IEEE Computer Society, 2008, pp. 541–544 (cit. on p. 92).

- [137] Mike Rettig. *Jetlang Library*. <http://code.google.com/p/jetlang/>. 2008 (cit. on pp. 26, 43).
- [138] John Rose. *JSR 292: Supporting Dynamically Typed Languages on the Java Platform*. <http://jcp.org/en/jsr/detail?id=292>. Oracle. July 2011 (cit. on pp. 6, 55, 59).
- [139] Jan Schäfer. *A programming model and language for concurrent and distributed object-oriented systems*. Verlag Dr. Hut, 2010 (cit. on p. 6).
- [140] Jan Schäfer and Arnd Poetzsch-Heffter. “JCoBox: Generalizing Active Objects to Concurrent Components”. In: *ECOOP*. 2010 (cit. on p. 6).
- [141] Aleksey Shipilev. *JMH: Java Microbenchmark Harness*. <http://openjdk.java.net/projects/code-tools/jmh/>. Oracle (cit. on p. 62).
- [142] Jim Shore. “Fail Fast”. In: *IEEE Software* 21.5 (2004), pp. 21–25 (cit. on p. 69).
- [143] Marjan Sirjani. “Formal specification and verification of concurrent and reactive systems”. PhD thesis. Citeseer, 2004 (cit. on p. 5).
- [144] Marjan Sirjani. “Rebeca: Theory, applications, and tools”. In: *Formal Methods for Components and Objects*. Springer. 2007, pp. 102–126 (cit. on p. 5).
- [145] Marjan Sirjani and Ali Movaghar. “Simulation in Rebeca”. In: *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications-Volume 2*. CSREA Press. 2002, pp. 923–926 (cit. on p. 5).
- [146] Sriram Srinivasan. *Kilim*. <http://www.malhar.net/sriram/kilim/> (cit. on p. 9).
- [147] Sriram Srinivasan and Alan Mycroft. “Kilim: Isolation-typed actors for Java”. In: *ECOOP 2008–Object-Oriented Programming*. Springer, 2008, pp. 104–128 (cit. on pp. 9, 26, 29, 43, 50).
- [148] Lukas Stadler, Christian Wimmer, Thomas Würthinger, Hanspeter Mössenböck, and John Rose. “Lazy Continuations for Java Virtual Machines”. In: *PPPJ ’09*. ACM, 2009 (cit. on p. 6).
- [149] Lukas Stadler, Thomas Würthinger, and Christian Wimmer. “Efficient Coroutines for the Java Platform”. In: *PPPJ ’10*. ACM, 2010 (cit. on p. 6).
- [150] TIOBE. *Programming Languages Index*. http://www.tiobe.com/tiobe_index. Apr. 2016 (cit. on p. 5).
- [151] Typesafe. *Akka*. <http://akka.io/>. Typesafe. 2010 (cit. on pp. 9, 43, 50, 61).
- [152] Jose Valim. *Elixir*. <http://elixir-lang.org/>. Elixir (cit. on pp. 9, 49).
- [153] Raja Vallée-Rai, Phong Co, Etienne Gagnon, et al. “Soot-a Java bytecode optimization framework”. In: *Proceedings of the 1999 conference of the Centre for Advanced Studies on Collaborative research*. IBM Press. 1999, p. 13 (cit. on p. 10).
- [154] Carlos A Varela, Gul Agha, Wei-Jen Wang, et al. “The SALSA Programming Language 1.1.2 Release Tutorial”. In: *Dept. of Computer Science, RPI, Tech. Rep* (2007), pp. 07–12 (cit. on p. 50).
- [155] Carlos Varela and Gul Agha. “Programming dynamically reconfigurable open systems with SALSA”. In: *SIGPLAN Not.* 36 (12 Dec. 2001), pp. 20–34 (cit. on pp. 26, 29, 43).

- [156] Wirfs-Brock, Rebecca J. and Johnson, Ralph E. “Surveying Current Research in Object-oriented Design”. In: *Commun. ACM* 33.9 (1990), pp. 104–124 (cit. on p. 47).
- [157] Peter Y. H. Wong, Richard Bubel, Frank S. de Boer, et al. “Testing abstract behavioral specifications”. In: *STTT* 17.1 (2015), pp. 107–119 (cit. on pp. 5, 97).
- [158] Jim Woodcock, Ana Cavalcanti, John Fitzgerald, Simon Foster, and Peter Gorm Larsen. “Contracts in CML”. In: *Leveraging Applications of Formal Methods, Verification and Validation. Specialized Techniques and Applications*. Springer, 2014, pp. 54–73 (cit. on p. 98).
- [159] Ingrid Chieh Yu, Einar Broch Johnsen, and Olaf Owe. “Type-Safe Runtime Class Upgrades in Creol”. In: *Proc. FMOODS’06*. Vol. 4037. LNCS. Springer-Verlag, June 2006, pp. 202–217 (cit. on p. 17).
- [160] Alexandros Zerzelidis and Andy Wellings. “A framework for flexible scheduling in the RTSJ”. In: *ACM Trans. Embed. Comput. Syst.* 10.1 (2010), 3:1–3:44 (cit. on p. 44).

List of Figures

1.1	General Architecture	8
2.1	<i>Crisp</i> Architecture: Structural Overview	20
2.2	New MethodInvocation	20
2.3	Policy-based selection of a MethodInvocation	21
2.4	Execution of a MethodInvocation	23
2.5	Increasing parallelism in <i>Crisp</i> for Prime Sieve	25
2.6	Utilizing both CPUs with Prime Sieve in <i>Crisp</i>	25
3.1	A kernel version of the real-time programming language	31
3.2	Life cycle for remote data processing	33
4.1	Architecture of Actor API in Java 8	58
4.2	Benchmarking comparison of ABS API and Akka	62
5.1	JMSeq Execution Architecture	68
5.2	Examples of Method Call Sequence Specification	71
5.3	Method Sequence Specification Grammar	71
5.4	Overview of JMSeq's process to verify a program	78
5.5	Software Architecture for Method Sequence Specification	80
5.6	Life cycle for data processing requests from customers	86
5.7	Comparison of measurements using JMSeq at Fredhopper	87
5.8	Comparison of different techniques with JMSeq	91
6.1	Task Automata for $\alpha(s, \tau, t_c)$ and $\beta(s, \tau)$	105
6.2	Task automata to execute verify_α and verify_β	106
6.3	M_{A_s} : Timed Automaton to execute task type <code>allocate</code> in M_P	106
6.4	M_{D_s} : Timed Automaton to execute task type <code>deallocate</code> in M_P	106
6.5	An example behavior for M_E	107
6.6	Evolving $\alpha(s, \tau, t_c)$ with different τ	110

List of Tables

1.1	Actor Model Support in Programming Languages	9
1.2	Actor programming libraries in Java	9
1.3	Actors at Work – Thesis Organization	10
1.4	Actors at Work – Conference and Journal Publications	11
2.1	Thread stack allocated for different executions	25
2.2	Number of live threads and total threads created for different runs of parallel prime sieve	26
2.3	Overview of evaluation of challenges	28
3.1	Evaluation Results	42

Colophon

This thesis was typeset with $\text{\LaTeX}$ 2_ε. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at <http://cleanthesis.der-ric.de/>.

