

Cormac Flanagan
Associate Professor
Alfred P. Sloan Research Fellow
Computer Science Department
University of California, Santa Cruz

January 2, 2008

Professional Experience

- 2005– Associate Professor, Department of Computer Science,
University of California, Santa Cruz
- 2003–05 Assistant Professor, Department of Computer Science,
University of California, Santa Cruz
- 1997–03 Principal Research Scientist, Systems Research Center,
Digital Equipment Corporation / Compaq Computer Corpora-
tion / Hewlett Packard Corporation
- 1995 Research Intern, Digital Equipment Corporation
- 1990–91 Software Engineer, Peregrine Expert Systems

Education

- Ph.D. 1997 Computer Science, Rice University.
- M.Sc. 1995 Computer Science, Rice University.
- B.Sc. 1990 Computer Science and Mathematics, University College Dublin.

Honors

- 2005-07 Alfred P. Sloan Foundation Fellowship, \$45,000.
- 2006 Ranked by CiteSeer as being in the top 0.15% of computer science
authors in terms of normalized citation counts (a proxy for re-
search impact). <http://citeseer.ist.psu.edu/allcitedn.html>.
- 2004 ACM SIGPLAN Distinguished Paper Award [27], International
Symposium on Software Testing and Analysis.
- 2003 “The Essence of Compiling with Continuations” [53] selected
as one of the 50 most influential contributions in the last twenty
years of the Programming Language Design and Implementation
Conference.

Grants and Awards

- 2007-10 Principal Investigator, National Science Foundation award CCR-0707885, “A JML Community Infrastructure – Revitalizing Tools and Documentation to Aid Formal Methods Research”, \$150,000. Part of a multi-university proposal funded at a level of \$895,000.
- 2003-07 Principal Investigator, National Science Foundation award CCR-0341179, “Checking Atomicity for Improved Multithreaded Software Reliability”, \$257,773.
- 2005-06 Principal Investigator, University of California Microelectronics Innovation and Computer Research Opportunities (MICRO) award, “Lightweight Transactions for Robust Error Handling”, \$42,174.
- 2005-06 Principal Investigator, Microsoft Phoenix Research award, “Lightweight Transactions for Robust Error Handling”, \$49,192.
- 2002-05 Irish Research Council for Science, Engineering and Technology Basic Research Award, “Automated Verification of Security Protocols”, Euro 173,000. (Collaborator)
- 1997 NSF-NATO Postdoctoral Fellowship awarded but declined by me in favor of a position at the Systems Research Center.
- 1996–97 Lodieska Stockbridge Vaughan Fellowship, Rice University.
- 1993,95,96 ACM SIGPLAN Professional Activities Committee grants.
- 1989 Scholarship in Computer Science, University College Dublin.
- 1988 Scholarship in Computer Science, University College Dublin.
- 1987 Scholarship in Computer Science, University College Dublin.

Consulting

- 2007– Consultant for Solidware Corporation.
- 2006– Invited expert on the Ecma TC39 committee responsible for the standardization of the JavaScript programming language.
- 2006– Consultant for Microsoft Corporation.
- 2006– Consultant for Mozilla Corporation.

Board Member

- 2007– Advisory Board Member, UCSC Extension, Software Engineering and Quality Program (SEQ).
- 2006–07 Advisory Board Member, UCSC Extension, Software Quality Engineering and Management Program (SQEM).

Publications

Publications are available at: <http://www.soe.ucsc.edu/~cormac>

Journal Publications

- [1] “Atomizer: A Dynamic Atomicity Checker for Multithreaded Programs”. Cormac Flanagan and Stephen N. Freund. *Science of Computer Programming*, 71 (2008), 89–109.
- [2] “Types for Atomicity: Static Checking and Inference for Java”. Cormac Flanagan, Stephen N. Freund, Marina Lifshin, and Shaz Qadeer. *ACM Transactions on Programming Languages and Systems*, to appear.
- [3] “Type Inference Against Races”. Cormac Flanagan and Stephen N. Freund. *Science of Computer Programming*, 64, 1 (September 2006), 140–165.
- [4] “Types for Safe Locking: Static Race Detection for Java”. Martin Abadi, Cormac Flanagan, and Stephen N. Freund. *ACM Transactions on Programming Languages and Systems*, 28, 2 (March 2006), 207–255.
- [5] “Modular Verification of Multithreaded Programs”. Cormac Flanagan, Stephen N. Freund, Shaz Qadeer, and Sanjit Seshia. *Theoretical Computer Science*, 338, 1-3 (June 2005), 153–183.
- [6] “Exploiting Purity for Atomicity”. Cormac Flanagan, Stephen N. Freund, and Shaz Qadeer. *IEEE Transactions on Software Engineering*, 31, 4 (April 2005), 275–291.
- [7] “Automatic Software Model Checking via Constraint Logic”. Cormac Flanagan. *Science of Computer Programming*, 50, 1 (March 2004), 253–270.

- [8] “DrScheme: A Programming Environment for Scheme”. Robert Bruce Findler, John Clements, Cormac Flanagan, Matthew Flatt, Shriram Krishnamurthi, Paul Steckler, and Matthias Felleisen. *Journal of Functional Programming*, 12, 2, (March 2002), 159–182.
- [9] “Annotation Inference for Modular Checkers”. Cormac Flanagan, Rajeev Joshi, and K. Rustan M. Leino. *Information Processing Letters*, 77, 2–4 (February 2001), 97–108.
- [10] “Componential Set-Based Analysis”. Cormac Flanagan and Matthias Felleisen. *ACM Transactions on Programming Languages and Systems*, 21, 2 (March 1999) 370–416.
- [11] “The Semantics of Future and an Application”. Cormac Flanagan and Matthias Felleisen. *Journal of Functional Programming*, 9, 1 (January 1999), 1–31.

Refereed Conference and Workshop Publications

- [12] “Status Report: Specifying JavaScript with ML”. David Herman and Cormac Flanagan. *ACM SIGPLAN Workshop on ML*, (October 2007).
- [13] “Cartesian Partial-Order Reduction”. Guy Gueta, Cormac Flanagan, Eran Yahav, and Mooly Sagiv. *13th International SPIN Workshop on Model Checking Software*, (July 2007).
- [14] “Space Efficient Gradual Typing”. David Herman, Aaron Tomb, and Cormac Flanagan. *Eighth Symposium on Trends in Functional Programming*, (April 2007).
- [15] “Unifying Hybrid Types and Contracts”. Jessica Gronski and Cormac Flanagan. *Eighth Symposium on Trends in Functional Programming*, (April 2007).
- [16] “Type Reconstruction for General Refinement Types”. Kenneth Knowles and Cormac Flanagan. *Programming Languages and Systems, 16th European Symposium on Programming, ESOP 2007*, Springer-Verlag (April 2007).
- [17] “Sage: Hybrid Checking for Flexible Specifications”. Jessica Gronski, Kenneth Knowles, Aaron Tomb, Stephen N. Freund, and Cormac Flanagan. *Workshop on Scheme and Functional Programming* (September 2006).

- [18] “Dynamic Architecture Extraction”. Cormac Flanagan and Stephen N. Freund. *Workshop on Formal Approaches to Testing and Runtime Verification* (August 2006).
- [19] “Hybrid Type Checking”. Cormac Flanagan. *Proceedings of the 33rd ACM Symposium on Principles of Programming Languages* (January 2006).
- [20] “Hybrid Types, Invariants, and Refinements for Imperative Objects”. Cormac Flanagan, Stephen N. Freund, and Aaron Tomb. *International Workshop on Foundations and Developments of Object-Oriented Languages* (January 2006).
- [21] “Automatic Synchronization Correction”. Cormac Flanagan and Stephen N. Freund. *Workshop on Synchronization and Concurrency in Object-Oriented Languages* (October 2005).
- [22] “Extending JML for Modular Specification and Verification of Multi-Threaded Programs”. Edwin Rodriguez, Matthew Dwyer, Cormac Flanagan, John Hatcliff, Gary T. Leavens, and Robby. *European Conference on Object Oriented Programming* (July 2005).
- [23] “Automatic Type Inference via Partial Evaluation”. Aaron Tomb and Cormac Flanagan. *Principles and Practice of Declarative Programming* (July 2005).
- [24] “Type Inference for Atomicity”. Cormac Flanagan and Stephen N. Freund. *Proceedings of the ACM SIGPLAN Workshop on Types in Language Design and Implementation, TLDI 2005*, (January 2005).
- [25] “Dynamic Partial-Order Reduction for Model Checking Software”. Cormac Flanagan and Patrice Godefroid. *Proceedings of the 32nd ACM Symposium on Principles of Programming Languages* (January 2005).
- [26] “Type Inference Against Races”. Cormac Flanagan and Stephen N. Freund. *Static Analysis, 11th International Symposium, SAS 2004*, Springer-Verlag (August 2004).
- [27] “Exploiting Purity for Atomicity”. Cormac Flanagan, Stephen N. Freund, and Shaz Qadeer. *Proceedings of the ACM International Symposium on Software Testing and Analysis, ISSTA 2004* (July 2004), 221–231. This paper received an ACM SIGPLAN Distinguished Paper Award.

- [28] “Verifying Commit-Atomicity Using Model-Checking”. Cormac Flanagan. *11th International SPIN Workshop on Model Checking Software*, (Susanne Graf and Laurent Mounier, eds.), Springer-Verlag (April 2004), 252–266.
- [29] “Software Model Checking via Iterative Abstraction Refinement Constraint Logic Queries”. Cormac Flanagan. *Workshop on Constraint Programming and Constraints for Verification*, (April 2004).
- [30] “Atomizer: A Dynamic Atomicity Checker for Multithreaded Programs”. Cormac Flanagan and Stephen N. Freund. *Proceedings of the 31st ACM Symposium on Principles of Programming Languages* (January 2004), 256–267.
- [31] “Theorem Proving using Lazy Proof Explication”. Cormac Flanagan, Rajeev Joshi, Xinming Ou, and James B. Saxe. *Computer Aided Verification, 15th International Conference, CAV 2003*, (Warren A. Hunt Jr. and Fabio Somenzi, eds.), Springer-Verlag (July 2003), 355–367.
- [32] “A Type and Effect System for Atomicity”. Cormac Flanagan and Shaz Qadeer. *Proceedings of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation 2003*, (June 2003), 338–349.
- [33] “Transactions for Software Model Checking”. Cormac Flanagan and Shaz Qadeer. *Workshop on Software Model Checking, SoftMC 2003*, *Electronic Notes in Theoretical Computer Science*, Volume 89, 3, 2003.
- [34] “Thread-Modular Model Checking”. Cormac Flanagan and Shaz Qadeer. *SPIN Model Checking Software, 10th International SPIN Workshop*, (Thomas Ball and Sriram K. Rajamani, eds.), Springer-Verlag (May 2003), 213–224.
- [35] “Automatic Software Model Checking using CLP”. Cormac Flanagan. *Programming Languages and Systems, 12th European Symposium on Programming, ESOP 2003*, (Pierpaolo Degano, ed.), Springer-Verlag (April 2003), 189–203.
- [36] “Types for Atomicity”. Cormac Flanagan and Shaz Qadeer. *Proceedings of the ACM SIGPLAN Workshop on Types in Language Design and Implementation, TLDI 2003*, (January 2003), 1–12.
- [37] “A Modular Checker for Multithreaded Programs”. Cormac Flanagan, Shaz Qadeer, and Sanjit A. Seshia. *Computer Aided Verification*,

14th International Conference, CAV 2002, (Ed Brinksma and Kim Guldstrand Larsen, eds.), Springer-Verlag (July 2002), 180–194.

- [38] “Extended Static Checking for Java”. Cormac Flanagan, K. Rustan M. Leino, Mark Lillibridge, Greg Nelson, James B. Saxe, and Raymie Stata. *Proceedings of the 2002 ACM Conference on Programming Language Design and Implementation, PLDI 2002*, SIGPLAN Notices 37, 5, (June 2002), 234–245.
- [39] “Thread-Modular Verification for Shared-Memory Programs”. Cormac Flanagan, Stephen N. Freund, and Shaz Qadeer. *11th European Symposium on Programming, ESOP 2002*, (Daniel Le Métayer, ed.), Springer-Verlag (April 2002), 262–277.
- [40] “Predicate Abstraction for Software Verification”. Cormac Flanagan and Shaz Qadeer. *Conference Record of POPL 2002: The 29th Symposium on Principles of Programming Languages*, (January 2002), 191–202.
- [41] “Detecting Race Conditions in Large Programs”. Cormac Flanagan and Stephen N. Freund. *Proceedings of the 2001 ACM Workshop on Program Analysis For Software Tools and Engineering, PASTE 2001* (June 2001), 90–96.
- [42] “Houdini, an Annotation Assistant for ESC/Java”. Cormac Flanagan and K. Rustan M. Leino. *ME 2001: Formal Methods for Increasing Software Productivity, International Symposium of Formal Methods Europe*, (José Nuno Oliveira and Pamela Zave, eds.), Springer-Verlag (March 2001), 500–517.
- [43] “Avoiding Exponential Explosion: Generating Compact Verification Conditions”. Cormac Flanagan and James B. Saxe. *Conference Record of POPL 2001: The 28th ACM Symposium on Principles of Programming Languages*, ACM SIGPLAN Notices 36, 3, (January 2001), 193–205.
- [44] “Type-Based Race Detection for Java”. Cormac Flanagan and Stephen N. Freund. *Proceedings of the 2000 ACM SIGPLAN Conference on Programming Language Design and Implementation*, SIGPLAN Notices 35, 5 (June 2000), 219–232.
- [45] “Object Types against Races”. Cormac Flanagan and Martin Abadi. *CONCUR '99: Concurrency Theory, 10th International Conference*,

- (Jos C. M. Baeten and Sjouke Mauw, eds.), Springer-Verlag (August 1999), 288–303.
- [46] “Types for Safe Locking”. Cormac Flanagan and Martin Abadi. *Programming Languages and Systems, 8th European Symposium on Programming, ESOP’99*, (S. Doaitse Swierstra, ed.), Springer-Verlag (March 1999), 91–108.
 - [47] “A New Way of Debugging Lisp Programs”. Cormac Flanagan and Matthias Felleisen. *Proceedings of the Conference on the 40th Anniversary of Lisp: Lisp in the Mainstream*, (November 1998).
 - [48] “DrScheme: A Pedagogic Programming Environment for Scheme”. Robert Bruce Findler, Cormac Flanagan, Matthew Flatt, Shriram Krishnamurthi, and Matthias Felleisen. *Programming Languages: Implementations, Logics, and Programs, 9th International Symposium*, (Hugh Glaser and Pieter H. Hartel and Herbert Kuchen, eds.), Springer-Verlag (September 1997), 369–388.
 - [49] “Componential Set-Based Analysis”. Cormac Flanagan and Matthias Felleisen. *Proceedings of the ACM SIGPLAN ’97 Conference on Programming Language Design and Implementation, PLDI ’97*, SIGPLAN Notices 32, 5 (June 1997), 235–248.
 - [50] “Catching Bugs in the Web of Program Invariants”. Cormac Flanagan, Matthew Flatt, Shriram Krishnamurthi, Stephanie Weirich, and Matthias Felleisen. *Proceedings of the ACM SIGPLAN’96 Conference on Programming Language Design and Implementation, PLDI ’96*, SIGPLAN Notices 31, 5 (May 1996), 23–32.
 - [51] “pHluid: The Design of a Parallel Functional Language Implementation on Workstations”. Cormac Flanagan and Rishiyur S. Nikhil. *Proceedings of the 1996 ACM SIGPLAN International Conference on Functional Programming, ICFP ’96*, SIGPLAN Notices 31, 6 (May 1996), 169–179.
 - [52] “The Semantics of Future and its use in Program Optimizations”. Cormac Flanagan and Matthias Felleisen. *Conference Record of the 22nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, (January 1995), 209–220.
 - [53] “The Essence of Compiling with Continuations”. Cormac Flanagan, Amr Sabry, Bruce F. Duba, and Matthias Felleisen. *Proceedings of the ACM SIGPLAN’93 Conference on Programming Language Design*

and Implementation, *PLDI '93*, SIGPLAN Notices 28, 6 (June 1993), 237–247.

Invited Papers

- [54] “Retrospective: The Essence of Compiling with Continuations”. Cormac Flanagan, Amr Sabry, Bruce F. Duba, and Matthias Felleisen. *20 Years of the ACM SIGPLAN Conference on Programming Language Design and Implementation 1979-1999, A Selection*, SIGPLAN Notices 39(4): 502–514, April 2004. This paper is a retrospective on [53].
- [55] “Atomizer: A Dynamic Atomicity Checker for Multithreaded Programs (Summary)”. Cormac Flanagan and Stephen N. Freund. *Parallel and Distributed Systems: Testing and Debugging Workshop, part of 18th International Parallel and Distributed Processing Symposium, IPDPS 2004* (April 2004). This invited paper is a summary of [30].

Collections

- [56] “Proceedings of the 2004 ACM SIGPLAN-SIGSOFT Workshop on Program Analysis For Software Tools and Engineering, PASTE 2004”. Cormac Flanagan and Andreas Zeller. ACM 2004.

Dissertation

- [57] “Componential Set-Based Analysis”. Cormac Flanagan. Doctoral Thesis, Rice University (July 1997).

Abstracts

- [58] “Type-Based Race Detection for Java (summary)”. Cormac Flanagan and Stephen N. Freund. *15th Annual IEEE Symposium on Logic in Computer Science* (June 2000). This paper is a summary of [44].

Technical Reports

- [59] “Modular Verification of Multithreaded Programs”. Cormac Flanagan, Stephen N. Freund, Shaz Qadeer, and Sanjit A. Seshia. Williams College Technical Note 04-08, Nov. 2004.
- [60] “An Explicating Theorem Prover for Quantified Formulas”. Cormac Flanagan, Rajeev Joshi, and James B. Saxe. Hewlett-Packard Labs Technical Report HPL-2004-199, Nov. 2004.

- [61] “Type Inference Against Races (extended version)”. Cormac Flanagan and Stephen N. Freund. Williams College Technical Note 04-06, Sept. 2004.
- [62] “Exploiting Purity for Atomicity (extended version)”. Cormac Flanagan, Stephen N. Freund, and Shaz Qadeer. Williams College Technical Note 04-05, July 2004.
- [63] “Partial Type And Effect Inference for Rcc/Java in NP-Complete”. Cormac Flanagan and Stephen N. Freund. Williams College Technical Note 04-01, Feb. 2004.
- [64] “Thread-Modular Verification For Shared-Memory Programs”. Cormac Flanagan, Stephen Freund, and Shaz Qadeer. Systems Research Center Technical Note SRC-TN-2001-003, Nov. 2001.
- [65] “Houdini, an Annotation Assistant for ESC/Java”. Cormac Flanagan and K. Rustan M. Leino. Systems Research Center Technical Note SRC-TN-2000-003, Dec. 2000.
- [66] “Modular and Polymorphic Set-Based Analysis: Theory and Practice”. Cormac Flanagan and Matthias Felleisen. Rice University Department of Computer Science Technical Report TR96-266, Jan. 1996.
- [67] “Set Based Analysis for Full Scheme and Its Use in Soft-Typing”. Cormac Flanagan and Matthias Felleisen. Rice University Department of Computer Science Technical Report TR95-254, Oct. 1995.
- [68] “Well-Founded Touch Optimization of Futures”. Cormac Flanagan and Matthias Felleisen. Rice University Department of Computer Science Technical Report TR94-239, Oct. 1994.
- [69] “The Semantics of Future”. Cormac Flanagan and Matthias Felleisen. Rice University Department of Computer Science Technical Report TR94-238, Feb. 1994.
- [70] “PLT MrSpidey: Static Debugger Manual”. Cormac Flanagan. March, 1997. Available at <http://www.plt-scheme.org/software/mrspidey/docs.html>.

Patents

- [1] “Method and Apparatus For Automatically Inferring Annotations”. Cormac Flanagan and K. Rustan M. Leino. U.S. Patent 7,120,902, (2006).

- [2] “Method and Apparatus For Organizing Warning Messages”. Cormac Flanagan and K. Rustan M. Leino. U.S. Patent 6,978,443, (2005).
- [3] “Method and Apparatus For Verifying Data Local To A Single Thread”. Cormac Flanagan and Stephen N. Freund. U.S. Patent 6,817,009, (2004).
- [4] “System and Method for Dynamic Detecting Unchecked Error Condition Values in Computer Programs”. Cormac Flanagan and Mike Burrows. U.S. Patent 6,378,087, (2002).
- [5] “System and Method for Lexing and Parsing Program Annotations”. Raymond Paul Stata, Cormac Flanagan, K. Rustan M. Leino, Mark Lillibridge, and James B. Saxe. U.S. Patent 6,353,925, (2002).
- [6] “System and Method for Statically Detecting Potential Race Conditions in Multithreaded Computer Programs”. Cormac Flanagan and Andrew Bernard. U.S. Patent 6,353,371, (2002).

Patent Applications

- [1] “Method and Apparatus For Automatically Inferring Annotations for an Extended Static Checker”. Cormac Flanagan and K. Rustan M. Leino. Filed December 4, 2001.
- [2] “Case-reduced Verification Condition Generation System and Method by Use of Dynamic Single Assumption and Assigning Labels to Variables at Control Join Points”. Cormac Flanagan, James B. Saxe, and Greg Nelson. Filed July 16, 2001.

Invited Talks

- 2007 Colloquium Speaker, Max Plank Institute for Software Systems, Saarbrücken, Germany.
- 2005 Distinguished Seminar Speaker, IBM T. J. Watson Research Center, Hawthorne, New York.
Invited to the Workshop on Construction and Analysis of Safe, Secure and Interoperable Smart devices (CASSIS), Nice, France.
- 2004 Invited Speaker, Workshop on Parallel and Distributed Systems: Testing and Debugging (PADTAD), Santa Fe, New Mexico.
Invited Speaker, Workshop on Constraint Programming and Constraints for Verification (CP+CV), Barcelona, Spain.

2003 Computer Science Colloquium, San Jose State University.

2002 Massachusetts Institute of Technology.
University of California at Berkeley.

Invited Tutorials

2006 “Static Analysis for Concurrency”, Summer School on Language-Based Techniques for Concurrent and Distributed Software. Eugene, Oregon, July 12-21, 2006.

Other Tutorials (by submission)

2005 “Atomicity for Reliable Concurrent Software”, Conference on Programming Language Design and Implementation. Chicago, Illinois, June 11, 2005.

External Professional Activities

2008 Reviewer for ACM SIGPLAN Outstanding Doctoral Dissertation Award.

Member of Proposal Review Panel, National Science Foundation.

Program Committee: Workshop on Specification and Verification of Component Based Systems (SAVCBS’08).

Program Committee: International Workshop on Foundations and Developments of Object-Oriented Languages (FOOL/WOOD’08).

Program Committee: Workshop on Parallel and Distributed Systems: Testing and Debugging.

Program Committee: ML Workshop.

2007 Program Committee: Workshop on Specification And Verification of Component-Based Systems.

Program Committee: The Seventh Workshop on Runtime Verification.

Program Committee: Workshop on Parallel and Distributed Systems: Testing and Debugging.

External Reviewer: ETH Zurich Research Commission.

External Reviewer: UC Microelectronics Program.

2005-07 Member of Steering Committee: ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering.

- 2006 Member of Proposal Review Panel, National Science Foundation.
 External Reviewer: University of California Micro Program.
 Program Committee: Workshop on Memory Systems Correctness and Performance.
 Posters Selection Committee: ACM SIGSOFT Symposium on Foundations of Software Engineering (FSE).
 Program Committee: Workshop on Formal Aspects of Testing and Runtime Verification.
 Program Committee: Workshop on Multithreading in Hardware and Software: Formal Approaches to Design and Verification.
 Program Committee: Workshop on Scheme and Functional Programming.
 Program Committee: The 13th International SPIN Workshop on Model Checking of Software.
 Program Committee: Workshop on Parallel and Distributed Systems: Testing and Debugging.
 External Reviewer: UC Microelectronics Program.
- 2005 Program Committee: Workshop on Specification and Verification of Component-Based Systems.
 Program Committee: The Fifth Workshop on Runtime Verification.
 Program Committee: Workshop on Evaluation of Software Defect Detection Tools.
 Program Committee: Workshop on Formal Techniques for Java-like Programs.
 Program Committee: The ACM Conference on Programming Language Design and Implementation (PLDI).
 Program Committee: The 12th International SPIN Workshop on Model Checking of Software.
 Program Committee: The ACM Workshop on Types in Language Design and Implementation (TLDI).
 Program Committee: The 3rd Workshop on Parallel and Distributed Systems: Testing and Debugging.
- 2004-05 **Steering Committee Chair:** ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering.
- 2004 Program Committee: Fourth Workshop on Runtime Verification.

Program Committee Co-Chair: ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering.

- 2003 Program Committee: The Thirtieth ACM Symposium on Principles of Programming Languages (POPL).
- 2002 Program Committee: The ACM Workshop on Program Analysis For Software Tools and Engineering (PASTE).
- 2001 Program Committee: The ACM Conference on Programming Language Design and Implementation (PLDI).

Reviewer of Technical Papers

- 2008 International Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA); Principles of Programming Languages (POPL); Programming Language Design and implementation (PLDI); ACM Transactions on Programming Languages and Systems (TOPLAS).
- 2007 ACM Transactions on Programming Languages and Systems (TOPLAS); International Conference on Functional Programming (ICFP); Logical Methods in Computer Science; 16th EACSL Annual Conference on Computer Science and Logic; ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages and Applications (OOPSLA); ACM Transactions on Software Engineering and Methodology (TOSEM); Programming Language Design and Implementation (PLDI).
- 2006 ACM Transactions on Software Engineering and Methodology (TOSEM); ACM Transactions on Programming Languages and Systems (TOPLAS); International Journal on Software Tools for Technology Transfer (STTT); Computer-Aided Verification (CAV); Principles of Programming Languages (POPL); Programming Language Design and Implementation (PLDI).
- 2005 ACM Transactions on Programming Languages and Systems (TOPLAS); Journal of Object Technology (JOT); Transactions on Software Engineering (TSE); Science of Computer Programming (SCP); European Symposium on Programming (ESOP); Foundations of Object-Oriented Languages (FOOL); Principles and Practice of Parellel Programming (PPoPP); Principles of Programming Languages (POPL); Computer-Aided Verification (CAV); Tools and Algorithms for the Construction and Analysis of Systems (TACAS); International Conference of Functional Programming (ICFP).

2004 ACM Transactions on Programming Languages and Systems (TOPLAS);
Journal of Computer Security (JCS); International Symposium on
Software Testing and Analysis (ISSTA); Tools and Algorithms for
the Construction and Analysis of Systems (TACAS); Static Analysis
Symposium (SAS); Journal on Formal Aspects of Computing;
International Conference of Functional Programming (ICFP).

Teaching

University courses 2007-08

			Enrolled	Co-taught	%Eval Retd
Fall	CMPS 203	Programming Languages	31	no	(not available)
Fall	CMPS 297A	Individual Study (graduate)	2	no	-
Fall	CMPS 299B	Thesis Research (graduate)	1	no	-

University courses 2006-07

			Enrolled	Co-taught	%Eval Retd
Fall	(on sabbatical leave)				
Fall	CMPS 297B	Individual Study (graduate)	1	no	-
Fall	CMPS 297B	Individual Study (graduate)	1	no	-
Fall	CMPS 299B	Thesis Research (graduate)	1	no	-
Winter	CMPS 112	Comparative Programming Languages	25	no	84%
Winter	CMPS 297A	Individual Study (graduate)	1	no	-
Spring	CMPS 253	Advanced Programming Languages	12	no	92%
Spring	CMPS 296	Masters Project	1	no	-
Spring	CMPS 297A	Individual Study (graduate)	1	no	-
Spring	CMPS 297B	Individual Study (graduate)	1	no	-

University courses 2005-06

			Enrolled	Co-taught	%Eval Retd
Fall	CMPS 203	Programming Languages	27	no	78%
Fall	CMPS 297A	Individual Study (graduate)	2	no	-
Fall	CMPS 299A	Thesis Research (graduate)	1	no	-
Winter	CMPS 115	Software Engineering Methodology	24	no	71%
Winter	CMPS 296	Masters Project	1	no	-
Winter	CMPS 297A	Individual Study (graduate)	2	no	-
Winter	CMPS 299A	Thesis Research (graduate)	1	no	-
Spring	CMPS 12A	Introduction to Programming	72	no	85%
Spring	CMPS 12L01	Introduction to Programming Lab	35	no	-
Spring	CMPS 12L02	Introduction to Programming Lab	33	no	-

Spring	CMPS 280G	Seminar in Software Engineering	5	no	60%
Spring	CMPS 297A	Individual Study (graduate)	1	no	-
Spring	CMPS 297B	Individual Study (graduate)	1	no	-
Spring	CMPS 299A	Thesis Research (graduate)	1	no	-
Summer	CMPS 297A	Individual Study/Research	1	no	-

University courses 2004-05

			Enrolled	Co-taught	%Eval Retd
Fall	CMPS 203	Programming Languages	28	no	86%
Fall	CMPS 280G	Seminar in Software Engineering	10	no	40%
Fall	CMPS 297A	Individual Study (graduate)	2	no	-
Winter	CMPS 115	Software Engineering Methodology	31	no	39%
Winter	CMPS 297A	Individual Study (graduate)	1	no	-
Spring	CMPS 290G	Topics in Software Engineering	8	no	88%
Summer	CMPS 297A	Individual Study	1	no	-

University courses 2003-04

			Enrolled	Co-taught	%Eval Retd
Winter	CMPS 290G	Topics in Software Engineering	8	no	75%
Winter	CMPS 280G	Seminar in Software Engineering	3	no	66%
Spring	CMPS 115	Software Engineering Methodology	33	no	73%
Spring	CMPS 198	Individual Study (undergraduate)	1	no	-
Spring	CMPS 297A	Individual Study (graduate)	1	no	-
Summer	CMPS 198	Individual Study (undergraduate)	1	no	-

Student Advising and Supervision

Adviser to Continuing Graduate Students

- Aaron Tomb
- Kenneth Knowles
- Jaeheon Yi
- Caitlin Sadowski
- Fan (Jason) Yang

Doctoral Dissertation Reading Committee Member

- Nathan Whitehead. *Combining Reason and Authority for Code Authentication and Verification*. UCSC, 3/2008.
- Andrew Santosa. *A Framework for Program Reasoning Based on Constraint Traces*. **National University of Singapore**, 3/2008.
- Jennifer Bevan. *Software Instabilities: Identification, Analysis, and Visualization*. UCSC, 10/2006.
- Sung Kim. *Adaptive Bug Prediction by Analyzing Project History*. UCSC, 8/2006.
- Philippe Meunier. *Modular Set-Based Analysis from Contracts*. **North-eastern University**, 5/2006.

Doctoral Qualifying Exam Committee Member

- Avik Chadhuri. *Foundations of Access Control for Secure Storage*. 1/2007.
- Aaron Tomb. *Hybrid Verification of Object-Oriented Programs*. 5/2006.
- Nathan Whitehead. *Combining Reason and Authority for Code Authentication and Verification*. 10/2005.
- Kai Pan. *An Investigation of Program Slice Encoding and Its Applications*. 11/2004.
- Sung Kim. *Semantic API Framework*. 11/2004.
- Jennifer Bevan. *Software Instabilities: Identification, Analysis, and Visualization*. 10/2003.

Masters of Science Thesis Reading Committee Member

- Daniel Libicki. *The GLIB Programming Language*. 3/2006.
- Spencer Tu. *EOS: A System for Evaluateable Objects in Scheme*. 3/2005.
- Michael K. Baker. *Object-Oriented Change Analysis*. 6/2004.

University Service

At UCSC

- 2007– Assistant Graduate Director for the Computer Science Department.

- 2004– Member of the Graduate Committee for the Computer Science Department.
- 2003–04 Member of the Faculty Recruitment Committee for the Computer Science Department.

At Rice University

- 1994–95 Vice President, Rice Graduate Student Association
- 1993–94 Member of the Honor Council
- 1992–93 Computer Science Representative to the Rice Graduate Student Association

Personal Statement

Cormac Flanagan
Associate Professor
Alfred P. Sloan Research Fellow
Computer Science Department
University of California, Santa Cruz

June 19, 2008

Research

Over the past two years, I continued to focus software validation and verification – a field with compelling practical motivation, since software defects cost an estimated \$60 billion annually, but also with deep technical challenges. My work ranges from the pragmatic development of particular tools to foundational research. It involves the study, development, and synthesis of a variety of software validation techniques, including type systems, static and dynamic analyses, theorem proving, and model checking. As indicated in my bibliography, this work has led to publications in high-quality conferences and journals.

Last fall, I was quite pleased to receive a Research Fellowship from the Alfred P. Sloan Foundation – the first such award received by a UCSC faculty member since 1999. My research has also been supported by Microsoft under their “Phoenix - Excellence in Programming” awards; I am grateful too for a matching grant from the University.

A significant part of my research is on concurrent systems, which are notoriously prone to errors that are difficult to detect or eliminate. These problems are exacerbated by modern, multi-core processors. Over the last several years, I have focused on the fundamental correctness property of *atomicity* (or serializability), and have developed analyses to automatically detect and eliminate atomicity violations.

This work appears to have been well-received. I presented a tutorial on atomicity at the 2005 Conference on Programming Language Design and Implementation, and have since accepted invitations to present this work at other venues, including at the Oregon Summer School on Concurrency and as part of the IBM Distinguished Seminar Series. My work has inspired a number of follow-on projects at other institutions, both academic and industrial. In particular, Microsoft has begun developing a tool to detect atomicity violations (based on my research), and I have starting consulting on this project. I collaborated with colleagues at other institutions to

incorporate the notion of atomicity into the widely-adopted Java Modeling Language.

On a somewhat different topic, I have recently begun work on an unusual approach to the long-standing problem of software verification. The fundamental undecidability of software verification causes multiple problems – unsoundness, incompleteness, limited specification languages, or finite state restrictions. In an attempt to circumvent these problems, I proposed an alternative approach of *hybrid verification*, which attempts to verify specifications statically, but can also enforce them at run-time if necessary. Although unconventional, hybrid verification provides several interesting advantages, and has become a central focus of my research.

An unexpected outcome of this research has been its influence on the JavaScript programming language. JavaScript is a key part of the so-called Web 2.0 movement, and is implemented on essentially all web browsers. The next version of JavaScript will mix both typed and untyped code, in a manner that requires techniques based on hybrid verification to guarantee type soundness. I recently joined the JavaScript Standardization Committee as an invited expert, and have begun work on formalizing and verifying key parts of the emerging language standard. It is generally quite difficult to transfer programming language ideas from research to practice, so I am quite pleased with these developments.

Research collaborations

Much of my research is joint work with a number of excellent colleagues. My contributions to collaborative projects during this review period are as follows.

- I often work with Stephen Freund (who spent his 2005-06 sabbatical year at UCSC) on concurrency analyses [3, 21, 24, 61]. Our contributions are generally balanced; I tend to contribute more on initial idea generation, technical formalism, and presenting results, while Stephen Freund tends to do more work on the implementation and evaluation side. We also developed techniques for dynamically inferring object models of legacy software [18].
- Stephen Freund and I also completed two journal papers [5, 6] that expanded on earlier joint research projects, with helpful feedback from our co-authors Shaz Qadeer and Sanjit Seshia.
- In the area of hybrid type checking, my research group (plus Stephen Freund) extended my initial work on hybrid type checking [19] in two

orthogonal directions: (1) to an object-oriented language [20]; and (2) to a functional language called Sage that has an unusually rich type system [17]. Our prototype Sage implementation is available at `sage.soe.ucsc.edu`. The work on these projects was generally balanced, with two of my students (Aaron Tomb and Kenneth Knowles) completing the underlying correctness proofs.

- I collaborated with a number of experts to incorporate atomicity into the widely-adopted Java Modeling Language [22]. My co-authors performed much of the work, while I provided detailed feedback based on my knowledge of atomicity. (With some of these colleagues, I am seeking funding under the NSF Computing Research Infrastructure program to continue work on this and related topics.)
- My student Aaron Tomb published his first peer-reviewed paper during this period [23], while I provided guidance and helped with presentation issues.

Teaching

Over the past two years, I have taught three graduate courses, three undergraduate courses, two graduate seminars, and directed fifteen independent studies. Teaching these classes was generally enjoyable and rewarding. One (small) difficulty I encountered is that my last three classes were all scheduled for 8am – a time slot that appears somewhat difficult for students and so does not facilitate the interactive style of teaching that I most enjoy. Nevertheless, I am generally quite satisfied with the results. More importantly, the students appear to have though highly of these classes. For example, my overall effectiveness as a teacher was rated as very good or excellent by 95% of the students in CMPS 203, by 92% of the students in CMPS 115, and all of the students in CMPS 290G.

In addition to these teaching responsibilities, I have been serving as advisor for three Ph.D. students. One of these students successfully advanced to candidacy in Spring 2006 and has already published three papers. The other two students are more junior, but I am generally quite satisfied with their progress to date.

I have participated in several thesis committees at UCSC, both at the M.Sc. and Ph.D. level, as well as on a number of candidacy committees. I was also the external member on a Ph.D. thesis committee at Northeastern University – a worthwhile role since this thesis built on some results from my own Ph.D. research.

I just wanted to thank you for being such a great programming teacher!!! You had to go through the wrath of the mistakes on my programming assignments and yet you still helped me out. You make programming such a fun class and I will always remember what you taught me, programming is all about that perfect team!!! - Albert De Guzman

First off I'd like to say I think you did an excellent job teaching this last Spring quarter. I am a politics major and, after taking your class, have decided to minor in computer science as I believe it will be an essential asset in the future. - Jeff McLaughlin

University service and other professional activities

Over the past two years, I have actively contributed to the wider research community through a variety of activities. In 2004-05, I chaired the steering committee for the ACM Workshop on Program Analysis for Software Tools and Engineering. I also accepted invitations to serve on eight program committees in 2005, most notably the ACM Conference on Programming Language Design and Implementation, which is one of the top two conferences in my field. I served on six additional program committees so far this year. These activities not only involve important work, but I believe they also bring UCSC's expertise in software engineering to the attention of the broader research community. More locally, I have continued to serve on the Graduate Committee of the Computer Science Department.

Over the past year, I have expanded my industrial collaborations in order to increase the impact and relevance of my research. Through my joint work with Microsoft, I hope to help improve the reliability of their Common Language Runtime. As a member of the JavaScript Standardization Committee (via a consulting arrangement with Mozilla Corporation), I hope to positively influence a widely-used programming language.