Matthias Felleisen
College of Computer Science
Northeastern University
Boston, Massachusetts 02115
matthias@ccs.neu.edu
http://www.ccs.neu.edu/home/matthias/

Education

Aug. 1987	Ph.D., Indiana University, Bloomington, Indiana
Dec. 1983	Dipl. Wirtschafts-Ingenieur, Universität Karlsruhe, Germany
Aug. 1981	M.S., The University of Arizona, Tucson, Arizona

Professional Experience

2001–pr.	Trustee[1] Professor, Northeastern University, Boston, Massachusetts
1993–2001	Professor, Rice University, Houston, Texas
1992–1993	Associate Professor, Rice University, Houston, Texas
1987–1992	Assistant Professor, Rice University, Houston, Texas
1984–1986	Research/Teaching Assistant, Indiana University, Bloomington
summer '85	Research Associate, MCC, Austin, Texas
1982–1983	Programmer, IDS, Karlsruhe, Germany
1980	Associate Instructor, Universität Karlsruhe, Germany

Extended Visits

June 1996	Ecole Normale Superieure, Paris, France
1993–1994	Carnegie Mellon University, Pittsburgh, PA
May 1991	Ecole Normale Superieure, Paris, France

Research Interests

I am interested in all aspects of programming, programming languages, and programming environments. Over the past few years, I have also begun to investigate the role of programming language technology in software engineering.

[1] An endowed chair donated by the Board of Trustees of Northeastern University.

Educational Interests

I am interested in the role of programming as a (software) design activity and in how to teach programming at all levels of education: K-12, college, and graduate/professional school.

Awards/Fellowships

2012	SIGPLAN Most Influential ICFP Paper Award
2012	SIGPLAN Achievement Award
2011	SIGCSE Outstanding Educator Award
2009	ACM Karl V. Karlstrom Outstanding Educator Award
2007	ACM Fellow
1998	George R. Brown Award for Innovative Teaching
1986–1987	IBM Graduate Research Fellowship
1982–1983	Konrad-Adenauer Stipendium
1980–1981	Fulbright Fellowship
1979	Preis des Landes Baden-Württemberg for Best Vordiplom of the Year

Active Grants

2011–2014	M. Felleisen, Semantics Engineering for Scripting Languages. With Krishnamurthi (Brown) and Findler (NWU).
2010–2014	O. Shivers, M. Felleisen, M. Wand. Gnosys. Darpa.
2009–2013	M. Felleisen. A Hot-House for Programming Languages. NSF.
2009–2012	M. Felleisen, R. Pucella. Software Contracts in a Higher-Order World. AFOSR.
2009–2013	M. Felleisen. Integrating Mechanized Logic into the SE Curriculum. NSF. (collaborative with R. Page).

Past Grants

2006–2009	M. Felleisen. Interface-Oriented Programming. NSF.
2006–2009	M. Felleisen. Using Market Forces to Improve the Design of Software. NSF. (collaborative with R. Findler).
2005–2009	M. Felleisen. Support for Language-Oriented Programming in PLT Scheme. NSF.
2004–2008	M. Felleisen. Well-founded Behavioral Software Contracts. NSF. (collaborative with R. Findler).
2002–2008	W. Clinger, M. Felleisen, M. Wand. Scheme on .NET. Microsoft Research. $950,000
2003–2006	M. Felleisen. Robust Web Services. NSF. (collaborative with S. Krishnamurthi).

2001–2005 M. Felleisen. Computing Education for Every Student in Secondary Schools. NSF. Supplemental grants: Cord.org, $30,000; Exxon, $50,000.

2000–2005 Vardi, M., M. Felleisen. Integrating Logic in the Computer Science Curriculum. NSF.

2000–2002 Cartwright, R. and M. Felleisen. NextGen: A Programming Environment for Generic Java. Texas Advanced Technology Program.

1997–2001 M. Felleisen, Modular Program Analyses for Higher-Order Programming Languages. NSF.

1997–2001 M. Felleisen, with C. Cartwright. Re-inventing Computing Education in High Schools. US Department of Education.

1997–2001 M. Felleisen. Re-inventing Computing Education in High Schools. Exxon.

1997–2001 Cartwright, R. and M. Felleisen (joint). Exploring a "Safe" Approach to Software Engineering. NSF CISE Educational Innovation Grant.

1997–2000 Cartwright, R., with M. Felleisen. A Smart Programming Environment for Java. NSF.

1998–2000 Cartwright, R. and M. Felleisen (joint). A Static Debugger for Java. Texas Advanced Technology Program.

1996–1998 Cartwright, R. and M. Felleisen (joint). Smart Programming Environments. NSF.

1996–1998 Vardi, M., with M. Felleisen. Diagrammatic Reasoning in Hardware Verification. CISE Postdoctoral Program.

1996–1997 M. Felleisen. From Scheme to Object-Oriented Programming. NSF. Planning supplement to CISE/EI.

1994–1997 Cartwright, R., with M. Felleisen. Can We Unify the Programming Curriculum? NSF Education Infrastructure.

1993–1997 Cox, A., J. Dennis, M. Felleisen, and W. Zwaenepoel. CISE Research Instrumentation. NSF.

1992–1995 Cartwright, R. and M. Felleisen (joint). Fully Abstract Semantics for Practical Programming Languages. NSF.

1992–1995 Felleisen, M. Soft Typing. Army.

1991–1994 Cartwright, R., with M. Felleisen. Practical Softtyping. Texas Advanced Technology Program.

1990–1992 Felleisen, M. On the Expressive Power of Programming Languages. NSF.

1988–1990 Felleisen, M. and R.S. Cartwright. The Semantic Foundation of Program Optimization. NSF.

1987–1990 K. Kennedy, R.S. Cartwright, K. Cooper, and M.Felleisen. An Automated System for Deriving Efficient Parallel Programs. DARPA.

Professional Activities

2009-pr. Editor in Chief: *Journal of Functional Programming*

2004-2008 Editor: *Journal of Functional Programming*

2001-2008 Editor: *Journal of Functional Programming*
in charge of the Educational Pearls section;

"Welcome" (2003, issue 4);

"The Structure and Interpretation of the Computer Science Curriculum", with Findler, Flatt, and Krishnamurthi, Vol 14(4), 2004, 365–378

1996-2001. Editorial Board: *Journal of Functional Programming*

2012 Program Committee Chairman, *The 2013 European Symposium on Programming*.

2006 Program Committee Chairman, *The 2007 ACM Symposium on the Principles of Programming Languages*.

2006 Program Committee, *The 2006 CUFP Workshop* (at *ICFP 2006*).

2005 Program Committee, *Object-Oriented Programming, Systems, Languages, and Applications 2005*.

2004 Program Committee, *International Conference on Functional Programming*.

2004 Panelist. National Academy of Science, CSTB: *Building Dependable Systems*.

2003 Host, *Scheme and Functional Programming*.

2003 Program Committee, *Continuation Workshop '04*

2003 Selection Committee, *HOSC—Issue on the Scheme workshops.*

2000 Organizer, *Scheme and Functional Programming*.

2000 Program Committee, *Practical Applications of Declarative Programming Languages,* **2000**.

1999 Program Committee, *European Symposium on Programming* **2000**.

1999 Co-Organizer, *Functional and Declarative Languages in Education,* **1999**.

1997–2000 Steering Committee. *International Conference on Functional Programming*

1997–1999 Professional Activities Committee, SIGPLAN

1998 General Chair, *International Conference on Functional Programming*.

1997 Program Committee, *Conference on Declarative Programming Languages in Education*(at *Programming Languages, Implementations, and Logics*).

1996 Organizer, *Workshop on Functional Languages in the Introductory Curriculum*

1996 Co-Organizer, *Scheme Implementors' Workshop*

1995 Program Committee, *International Conference on Functional Programming*.

1994	Program Committee, *Conference on Lisp and Functional Programming* (LFP '94).
1993	Program Committee, *Workshop on State in Programming Languages*.
1993	Selection Committee, *LASC—Special Issue on Continuations.*
1993	Program Committee, *Conference on Functional Programming Languages and Architecture.*
1991	Program Committee, *Continuation Workshop '92*
1989	Program Committee, *17th ACM Symposium on the Principles of Programming Languages*.

Outreach & Software Projects

1995–pr.	Program by Design (formerly known as TeachScheme!)
1994–pr.	DrScheme. Collaborators: Flatt (Utah), Findler (Chicago), Krishnamurthi (Brown), Eli Barzilay (NEU), plus many other volunteers.
1985–1989	Prolog-in-Scheme and Schelog: macro-embeddings of Prolog into Scheme. Main collaborator: D. Sitaram.
1985–1989	extend-syntax: a pattern-oriented, hygienic macro system for Scheme88. Main collaborator: J. Greiner.

Books

2013	*Realm of Racket: For Freshmen, by Freshmen*. NoStarch Press. (With Bryce, DeMaio, Florence, Lin, Lindemann, Nussbaum, Peterson, Plessner, Van Horn, Barksi)
2010	*Semantics Engineering with PLT Redex*. MIT Press. Korean translation: 2012. (With R. Findler, M. Flatt).
2001	*How to Design Programs*. MIT Press. Korean translation: 2012. Spanish translation: 2008. Chinese translation: 2003. Polish translation: 2002. (With R. Findler, M. Flatt, S. Krishnamurthi)
1998	*A Little Java, A Few Patterns*, MIT Press. — Japanese translation appeared in 1998. (With D. P. Friedman)
1998	*The Little MLer*. MIT Press. (With D.P. Friedman)
1996	*The Seasoned Schemer*. MIT Press. (With D.P. Friedman)
1996	*The Little Schemer* (Fourth Edition). MIT Press. — Third Edition: 1989, MacMillan (SRA division). Trade Edition: 1987, MIT Press. Second Edition: 1985, Science Research Associates. Japanese translation: 1990, McGraw Hill International. French translation: 1991, Mason Publishers. (With D.P. Friedman)

Refereed Publications: Journals and Book Chapters

2011 Dimoulas, C., M. Felleisen. On Contract Satisfaction in a Higher-Order World. *ACM Transactions on Programming Languages and Systems*, 33(5), 16:1–16:29.

2010 Culpepper, R., M. Felleisen. Debugging Macros. *Science of Computer Programming* **75**(7), 496–515.

2009 Felleisen, M., S. Krishnamurthi. Viewpoint: Why computer science doesn't matter. *Communications of the ACM* **52**(7), 37–40.

2007 Krishnamurthi S., P. Hopkins, J. McCarthy, P. Graunke, G. Pettyjohn , M. Felleisen. Implementation and Use of the PLT Scheme Web Server. *Journal of Higher-Order and Symbolic Computing* **20**(4), 431–460.

2006 Krishnamurthi S., R. B. Findler, P. Graunke, and Matthias Felleisen. Modeling Web Interactions and Errors. In Dina Goldin, Scott Smolka, Peter Wegner (eds.), *Interactive Computation: The New Paradigm*, Springer Verlag, 2006, 255–276.

2004 Matthews, J., R. Findler, P. Graunke, S. Krishnamurthi, M. Felleisen. Automatically Restructuring Programs for the Web. *Journal of Automated Software Engineering.* Special issue on *ASE* 2002. Vol. 11(4), October 2004, 337–364.

2004 Clements, J. and M. Felleisen. A tail-recursive machine semantics for stack inspection. *ACM Transactions on Programming Languages and Applications*. November 2004, 1029–1052.

2004 Felleisen, M., R. Findler, M. Flatt, S. Krishnamurthi. The TeachScheme! Project: Computing and Programming for Every Student *Journal of Computer Science Education* Special issue on K12 education, Vol. 14(1), 2004, 55–77.

2002 Findler, R., M. Flatt, S. Krishnamurthi, P. Steckler, and M. Felleisen. The DrScheme Programming Environment. *Journal of Functional Programming* **12**(2), 159–182.

1999 Flanagan, C., M. Felleisen. Compositional set-based analysis. *ACM Transactions on Programming Languages and Applications* **21** (2), 1999, 369–415.

1999 Flanagan, C. and M. Felleisen. The semantics of futures and an application. *Journal of Functional Programming* **9** (1), 1999, 1–31.

1998 Flatt, M., S. Krishnamurthi, and M. Felleisen. A programmer's reduction semantics for classes and mixins. J. Alves-Foss (Ed.), *Formal Methods for Java*. Lecture Notes in Computer Science **1523**. Springer Verlag, Berlin, 1998, 241–270.

1997 Ariola, Z. and M. Felleisen, The Call-By-Need Lambda Calculus. *Journal of Functional Programming* **7** (3), 1997, 265–301.

1996 Felleisen, M. and S. Weeks. On the orthogonality of assignments and procedures in algol. In *Algol-like Languages*, P. O'Hearn and R.D. Tennent, Eds. Birkhä user, 1996, 101–123.

1996 Cartwright, R. and M. Felleisen. Program Verification through Soft Typing. *ACM Computing Surveys*, June 1996, 349–351.

1994 Wright, A. and M. Felleisen. A syntactic approach to type soundness. Department of Computer Science, Rice University, Technical Report No 160. *Information and Computation* **115** (1), 38–94.

1994 Cartwright, R.S., Curien, P.-L., and M. Felleisen. Fully abstract models of observably sequential languages. *Information and Computation* **111** (2), 1994, 297-401.

1993 Sabry, A. and M. Felleisen. Reasoning with programs in continuation-passing style. *Lisp and Symbolic Computation* **6** (3/4), 289–360.

1992 Felleisen, M. and R. Hieb. The revised report on the syntactic theories for control and state. *Theoretical Computer Science* **102**, 235–272.

1991 Felleisen, M. On the expressive power of programming languages. *Science of Computer Programming* **17** (Special issue on ESOP papers), 35–75.

1990 Sitaram, D. and M. Felleisen. Control delimiters and their hierarchies. *Lisp and Symbolic Computation* **3** (1), 67–100.

1989 Felleisen, M. and D.P. Friedman. A syntactic theory of sequential state. *Theoretical Computer Science* **69** (3), 243–287.

1987 Felleisen, M. Reflections on Landin's J-operator: A partly historical note. *Journal of Computer Languages* (Pergamon Press) **12** (3/4), 197–207.

1987 Felleisen, M., D.P. Friedman, E. Kohlbecker, and B. Duba. A syntactic theory of sequential control. *Theoretical Computer Science* **52** (3), 205–237.

1986 Felleisen, M. and D.P. Friedman. Control operators, the SECD-machine, and the λ-calculus. In *Formal Description of Programming Concepts III*, edited by M. Wirsing. Elsevier Science Publishers B.V. (North-Holland), Amsterdam, 1986, 193–217.

1986 Felleisen, M. and D.P. Friedman. A closer look at export and import statements. *Journal of Computer Languages* (Pergamon Press) **11** (1), 28–37.

Conference Publications

2012 V. St-Amour, S. Tobin-Hochstadt, and M. Felleisen. Optimization Coaching. In *Proceedings 2012 OOPSLA*, 163–178.

2012 A. Takikawa, S. Strickland, C. Dimoulas, S. Tobin-Hochstadt, and M. Felleisen. Gradual Typing for First-Class Classes. In *Proceedings 2012 OOPSLA*, 793–810.

2012 Chang, S. and M. Felleisen. The Call-by-need Lambda Calculus, Revisited. In *Proc. 2012 European Symposium on Programming*, 128–147.

2012 Dimoulas, C., Tobin-Hochstadt, S., and M. Felleisen. Complete Monitors for Behavioral Contracts. In *Proc. 2012 European Symposium on Programming*, 214–233.

2012 Klein, C., J. Clements, C. Dimoulas, C. Eastlund, M. Felleisen, M. Flatt, J. McCarthy, J. Rafkind, S. Tobin-Hochstadt, R. B. Findler. Run Your Research. In *Proc. 39th ACM Symposium on Principles of Programming Languages*, 2012, 285–296.

2012 St-Amour, V., S. Tobin-Hochstadt, M. Felleisen. Typing the Numeric Tower. In *Proc. 2012 Conf. Principles and Practice of Declarative Programming*, 289–304.

2011 Chang S., E. Barzilay, J. Clements, and M. Felleisen. From Stack Traces to Lazy Rewriting Sequences. In *Proc. 23rd Symposium on Implementation and Application of Functional Languages*, 100-0115.

2011 Tobin-Hochstadt, S., V. St-Amour, R. Culpepper, M. Flatt, M. Felleisen. Languages as Libraries. In *Proc. Symposium on Programming Languages: Design and Implementation*, 132–141.

2011 Dimoulas, C., R. Findler, C. Flanagan, M. Felleisen. Correct Blame for Contracts: No More Scapegoating. In *Proc. 38th ACM Symposium on Principles of Programming Languages*, 2011, 215–226.

2010 Strickland, T.S., M. Felleisen. Contracts for first-class modules. In *Proc. Sixth Symposium on Dynamic Languages 2010*.

2010 Eastlund C., M. Felleisen Hygienic macros for ACL2. In *Proc. 2010 Symposium on Trends in Functional Programming*, 84–101.

2010 Chang, S., D. van Horn, M. Felleisen Evaluating Call-By-Need on the Control Stack. In *Proc. 2010 Symposium on Trends in Functional Programming*, 1–15.

2010 Culpepper, R. and M. Felleisen. Fortifying macros. In *Proc. 2010 ACM International Conference on Functional Programming*, 2010, 235–246.

2010 Strickland, T.S., M. Felleisen. Implementing General Contract Boundaries. In *Proc. 21st Symposium on Implementation and Application of Functional Languages*.

2009 Strickland, T.S., M. Felleisen. Contracts for First-Class Modules. In *Proc. Fifth Symposium on Dynamic Languages 2009*, 27–38.

2009 Eastlund C., M. Felleisen Making induction manifest in modular ACL2. In *Proc. 2009 Conf. Principles and Practice of Declarative Programming*, 105–116.

2009 Dimoulas C., R. Pucella, M. Felleisen. Future contracts. In *Proc. 2009 Conf. Principles and Practice of Declarative Programming*, 195–206.

2009 Felleisen, M.,R.B. Findler, M. Flatt, S. Krishnamurthi. A functional I/O system or, fun for freshman kids. In *Proc. 2007 ACM International Conference on Functional Programming*, 2009, 47–58

2009 Strickland, S., Tobin-Hochstadt, S., and M. Felleisen. Practical Variable-Arity Polymorphism. In *Proc. 2009 European Symposium on Programming*, 32–46.

2009 Eastlund, C. and M. Felleisen. Toward a Practical Module System for ACL2. In *Proc. 9th ACM Symposium on Practical Aspects of Declarative Languages*, 2009, 46–60.

2008 Tobin-Hochstadt, S. and M. Felleisen. The Design and Implementation

of Typed Scheme. In *Proc. 35th ACM Symposium on Principles of Programming Languages*, 2008, 395–407.

2007 Flatt, M., R.B. Findler, G. Yu, M. Felleisen. Adding delimited and composable control to a production programming environment. In *Proc. 2007 ACM International Conference on Functional Programming*, 2007, 165–177.

2007 Culpepper. R and M. Felleisen. Debugging Macros, In *Proc. Generative Programming and Component Engineering*, 2007, 135–144.

2006 Flatt, M., R.B. Findler, M. Felleisen. Scheme with Classes, Mixins, and Traits. In *Proc. Asian Symposium on Programming Languages and Systems (ASPLA 2006)*, 270–289.

2006 Tobin-Hochstadt, S., M. Felleisen. Interlanguage Migration: From Scripts to Programs. *Proc. Dynamic Languages Symposium (OOPSLA 2007 Track)*, 964–974.

2006 Meunier, P., R. Findler, and M. Felleisen Modular Set-Based Analysis from Contracts. In *Proc. 33rd ACM Symposium on Principles of Programming Languages*, 2006, 218–231.

2005 Pettyjohn, G., J. Clements, J. Marshall, S. Krishnamurthi, M. Felleisen Continuations from Generalized Stack Inspection. In *Proc. 2005 ACM International Conference on Functional Programming*, 216–227.

2005 Cobbe, R. and M. Felleisen Environmental acquisition revisited. In *Proc. 32nd ACM Symposium on Principles of Programming Languages*, 2005, 14–25.

2004 Culpepper, R., M. Felleisen Taming macros. In *Proc. 2004 GPCE Symposium*, 225–243.

2004 Antoniu, T., P. Steckler, S. Krishnamurthi, Erich Neuwirth, M. Felleisen Validating the unit correctness of spreadsheet programs. In *Proc. 2004 International Conference of Software Engineering*, 439–448.

2004 Matthews, J., R. Findler, M. Flatt, M. Felleisen. A Visual Environment for Developing Context-Sensitive Term Rewriting Systems. In *Proc. 2004 International Conference on Rewriting Techniques and Applications*, 301–311.

2004 Findler, R., M. Flatt, M. Felleisen. Semantic casts. In *Proc. 2004 European Conference of Object-Oriented Programming*, 364–389.

2003 Graunke, P., R. Findler, S. Krishnamurthi, M. Felleisen. Modeling Web Interactions. In *Proc. 2003 European Symposium on Programming*, 238–252.

2003 Clements, J., M. Felleisen. A tail-recursive semantics for stack-inspection. In *Proc. 2003 European Symposium on Programming*, 22–37.

2002 Felleisen, M.. Developing Interactive Web Programs In *Proc. 2002 Oxford Summer School on Advanced Functional Programming*, Springer Lecture Notes, 100–128.

2002 Findler, R., M. Felleisen. Contracts for higher-order functions. In *Proc. 2002 International Conference on Functional Programming*.

2002 Logan, M., M. Felleisen, D. Blank-Edelman. Environmental acquisition in network management. In *Usenix LISA*, 175–184.

2001 Findler, R., M. Latendresse, M. Felleisen. Behavioral contracts and behavioral subtyping. In *Proc. 2001 SIGSOFT Conference on the Foundations of Software Engineering.* 229–236.

2001 Findler, R., M. Felleisen. Contract soundness for object-oriented languages. In *Proceedings 2001 OOPSLA*, 1–15.

2001 Graunke, P., R. Findler, S. Krishnamurthi, M. Felleisen. Automatically restructuring programs for the web. In *Automated Software Engineering*, 211–222.

2001 Graunke, P., S. Krishnamurthi, S. Van Der Hoeven, M. Felleisen. Programming the web with high-level programming languages. In *Proc. 2001 European Symposium on Programming*, 122–136.

2001 Clements, J., M. Flatt, and M. Felleisen. Modeling an Algebraic Stepper. In *Proc. 2001 European Symposium on Programming*, 320.-334.

2000 Krishnamurthi S., M. Felleisen, and B. Duba. From macros to reusable generative programming. In *Proc. First International Symposium on Generative and Component-Based Software Engineering*. Springer Lecture Notes.

1999 Flatt, M., R. Findler, S. Krishnamurthi, and M. Felleisen. Programming languages as operating systems (or, revenge of the son of the Lisp machine). In *Proc. 1999 International Conference on Functional Programming*, 138–146.

1999 Krishnamurthi S., Y.D. Erlich, and M. Felleisen. Expressing structural properties as language constructs. In *Proc. 1999 European Symposium on Programming*, 258–272.

1998 Krishnamurthi S., and M. Felleisen. Toward a formal theory of extensible software. In *Proc. 1998 Sigsoft Conference on the Foundations of Software Engineering.* 1998, 88–97.

1998 Krishnamurthi S, M. Felleisen, and D.P. Friedman. Synthesizing object-oriented and functional design to promote re-use. In *Proc. 1998 European Conference on Object-Oriented Programming.* Springer Lecture Notes in Computer Science. Berlin, 1998, 91–113.

1998 Flatt, M., and M. Felleisen. Units: Cool modules for HOT languages. In *Proc. Sigplan 1998 Conference on Programming Language Design and Implementation*, 236–248.

1998 Flatt, M., S. Krishnamurthi, and M. Felleisen. Classes and mixins. In *Proc. 25th ACM Symposium on Principles of Programming Languages*, 1998, 171–183.

1997 Findler, R. C. Flanagan, M. Flatt, S. Krishnamurthi, and M. Felleisen. DrScheme: a pedagogic programming environment for Scheme. In *Proc. 1997 Symposium on Programming Languages: Implementations and Logics*, 369-386.

1997 Flanagan, C., M. Felleisen. Compositional set-based analysis. In *Proc. Sigplan 1997 Conference on Programming Language Design and Implementation*, 235–249.

1996 Flanagan, C., M. Flatt, S. Krishnamurthi, S. Weirich, and M. Felleisen. Static debugging or browsing the web of program invariants. In *Proc.*

Sigplan 1996 Conference on Programming Language Design and Implementation, 1996, 23–32.

1995 Morrisett, G., M. Felleisen, and R. Harper. Modeling memory management. In *Proc. Symposium on Functional Programming and Computer Architectures*, 1995, 66–77. Also appeared as: Department of Computer Science, Carnegie Mellon University, Technical Report, 1994.

1995 Flanagan, C. and M. Felleisen. The semantics of futures and its use in program optimization. In *Proc. 22nd ACM Symposium on Principles of Programming Languages*, 1995, 209–220.

1995 Ariola, Z., M. Felleisen, M. Odersky, and P. Wadler. The call-by-need λ-calculus. In *Proc. 22nd ACM Symposium on Principles of Programming Languages*, 1995, 233–246.

1994 Sabry, A. and M. Felleisen. Is continuation-passing useful for data flow analysis? In *Proc. SIGPLAN 1993 Conference on Programming Language Design and Implementation*, 1994, 1–12.

1994 Cartwright, R., and M. Felleisen. Extensible denotational language specifications. In *Theoretical Aspects of Computer Software*. Lecture Notes in Computer Science **789**. Springer Verlag, 1994, 244–272. (Invited Paper)

1993 Flanagan, C., A. Sabry, B.F. Duba, and M. Felleisen. The essence of compiling with continuations. In *Proc. SIGPLAN 1993 Conference on Programming Language Design and Implementation*, 1993, 237–247.

1993 Weeks, S. and M. Felleisen. On the orthogonality of procedures and assignment in Algol 60. In *Proc. 20th ACM Symposium on Principles of Programming Languages*, 1993, 57–70.

1992 Kanneganti,R., R. Cartwright, M. Felleisen. SPCF: Its Model, Calculus, and Computational Power. In *Proc. REX Workshop on Semantics and Concurrency*. Lecture Notes in Computer Science **666**. Springer Verlag, 1992, 318–347. (Invited Paper)

1992 Sabry, A. and M. Felleisen. Reasoning with programs in continuation-passing style. In *Proc. 1992 ACM Symposium on Lisp and Functional Programming*, 1992, 288–298.

1992 Cartwright, R. and M. Felleisen. Observable sequentiality and full abstraction. In *Proc. 19th ACM Symposium on Principles of Programming Languages*, 1992, 328–342.

1991 Crank, E. and M. Felleisen. Parameter-passing techniques and the λ-calculus. In *Proc. 18th ACM Symposium on Principles of Programming Languages*, 1991, 233–245.

1991 Sitaram, D. and M. Felleisen. Modeling continuations without continuations. In *Proc. 18th ACM Symposium on Principles of Programming Languages*, 1991, 185–196.

1990 Felleisen, M. On the expressive power of programming languages. In *Proc. 1990 European Symposium on Programming*, Neil Jones, Ed. Lecture Notes in Computer Science **432**. Springer Verlag, 1990, 134–151.

1990 Sitaram, D. and M. Felleisen. Reasoning with continuations II: Full abstraction for models of control. In *Proc. 1990 Conference on Lisp and*

Functional Programming, 161–175.

1989 Cartwright, R.S. and M. Felleisen. The semantics of program dependence. In *Proc. 1989 ACM Conference on the Design and Implementation of Programming Languages*, 13–27.

1988 Felleisen, M. λ-v-CS: An extended λ-calculus for Scheme. In *Proc. 1988 Conference on Lisp and Functional Programming*, 72–85.

1988 Felleisen, M., M. Wand, D.P. Friedman, and B. Duba. Abstract continuations: A mathematical semantics for handling full functional jumps. In *Proc. 1988 Conference on Lisp and Functional Programming*, 52–62.

1988 Felleisen, M. The theory and practice of first-class prompts. In *Proc. 15th Symposium on Principles of Programming Languages*, 180–190.

1987 Felleisen, M. and D.P. Friedman. A reduction semantics for imperative higher-order languages. In *Proc. Conference on Parallel Architectures and Languages Europe, Volume II: Parallel Languages*. Lecture Notes in Computer Science **259**. Springer-Verlag, 1987, 206–223.

1987 Felleisen, M. and D.P. Friedman. A calculus for assignments in higher-order languages. In *Proc. 14th Symposium on Principles of Programming Languages*, 314–325.

1986 Felleisen, M., D.P. Friedman, E. Kohlbecker, and B. Duba. Reasoning with continuations. In *Proc. First Symposium on Logic in Computer Science*, 131–141.

1986 Kohlbecker, E., D.P. Friedman, M. Felleisen, and B. Duba. Hygienic macro expansion. In *Proc. 1986 Conference on LISP and Functional Programming*, 151–161.

Miscellaneous

2009 Eastlund, C., and M. Felleisen. Automatically Verified GUI Programs. *Proc. Symp. ACL2 Theorem Prover and its Applications*, 2009.

2008 Page, R., Eastlund, C., and M. Felleisen. Functional programming and theorem proving for undergraduates: a progress report. *Proc. Works 2008, Functional and declarative programming in education*.

2007 Eastlund, C., Vaillancourt, D., and M. Felleisen. ACL2 for Freshmen: First Experiences. *Proc. Symp. ACL2 Theorem Prover and its Applications*, 2007.

2006 Vaillancourt, D., R. Page, M. Felleisen. Dracula: ACL2 in DrScheme. *Proc. Symp. ACL2 Theorem Prover and its Applications*, 2006.

2006 Culpepper, R. and M. Felleisen. A Macro Stepper for Sceme *Proc. Worksh. Scheme and Functional Programming*, Portland (OR), 2006.

2004 Flanagan, C., A. Sabry, B.F. Duba, and M. Felleisen. The essence of compiling with continuations. In *20 Years of the ACM SIGPLAN Conference on Programming Language Design and Implementation (1979 - 1999): A Selection*. K.S. McKinley, Editor, ACM SIGPLAN Notices,

Volume 39, Number 4, April 2004.

2004 Clements, Felleisen, Findler, Flatt, Krishnamurthi. Fostering little languages with macros. Dr Dobb's Journal, March 2004. Invited paper.

2004 Felleisen, Findler, Flatt, Krishnamurthi. Building little languages with macros. Dr Dobb's Journal, March 2004. Invited paper.

2003 Felleisen, M. On *Gordon Plotkin's Call-By-Name, Call-By-Value, and the Lambda Calculus*. In *Reminiscences on influential papers*, F. Reig and M. Franz (eds.). SIGPLAN Notices 2003.

2002 M. Felleisen, R. Findler, M. Flatt, and S. Krishnamurthi. The structure and interpretation of the computer science curriculum. In *Proc Workshop Functional and Declarative Programming in Education*, 2003.

2001 Clements, J., P. Graunke, S. Krishnamurthi, and M. Felleisen. Little languages and their programming environments. In *Proc. Monterey Workshop*, 2001.

2001 Findler, R.B., M. Latendresse, and M. Felleisen Object-oriented Programming Languages Need Well-founded Contracts Rice University, Technical Report 01-372.

1999 Felleisen, M. and R. Cartwright. Safety as a software metric. In *Proceedings of 12th Conference on Software Engineering Education and Training* (CSEE&T'99), (Invited Position Paper). New Orleans, March 1999.

1999 Krishmanurthi, S. and M. Felleisen The technology that computer science education overlooked. In *Proceedings of International Conference on Mathematics/Science, Education and Technology.* San Antonio, March 1998.

1998 Flanagan, C. and M. Felleisen. A new way of debugging Lisp programs. In *Lisp in the Mainstream: The 40th Annniversary Conference of Lisp Users*. Berkeley, November 1998.

1997 Felleisen, M. Findler, R. M. Flatt, and Shriram K. The DrScheme Project: An Overview. In *SIGPLAN Notices*, June 1998. (Invited Column)

1995 Cormac Flanagan, Matthias Felleisen. Set Based Analysis for Full Scheme and Its Use in Soft-Typing Rice University, Technical Report TR95-253.

1995 Cormac Flanagan, Matthias Felleisen. The Semantics of Futures. Rice University, Technical Report TR95-238.

1995 Cormac Flanagan, Matthias Felleisen. Well-Founded Touch Optimization of Futures. Rice University, Technical Report TR95-239.

1987 Felleisen, M. The calculi of λ_v-CS conversion: a syntactic theory of control and state in imperative higher-order programming languages. Computer Science Department, Indiana University, Technical Report No 226.

1987 Duba, B. F., M. Felleisen, and D.P. Friedman. Dynamic identifiers can be neat. Computer Science Department, Indiana University, Technical Report No 220.

1987 Felleisen, M., D.P. Friedman, B. Duba, and J. Merrill, Beyond continuations. Computer Science Department, Indiana University, Technical Report No 216.

1986 Felleisen, M. Recursion and Circularity: Extended Puzzle with Solution. Computer Science Department, Indiana University, Technical Report 201.

1985 Felleisen, M. Transliterating Prolog into Scheme. Computer Science Department, Indiana University, Technical Report 183.

Ph.D. Students

2012 Christos Dimoulas. *Foundations for Behavioral Higher-Order Contracts.*

2012 Carl Eastlund. *Modular Proof Development in ACL2.*

2012 T. Stephen Strickland. *Scaling Contracts to Realistic Languages.*

2010 Ryan Culpepper. *Refining Syntactic Sugar.*

2009 Sam Tobin-Hochstadt. *Typed Scheme: From Scripts to Programs.*

2008 Richard Cobbe. *Much Ado about Nothing: Putting Null in its Place.*

2006 Philippe Meunier. *Modular Set-Based Analysis from Contracts.*

2005 John Clements. *Portable and High-level Access to the Stack with Continuation Marks.*

2003 Paul Graunke. *Web Interactions.*

2001 Robert Findler. *Behavioral Software Contracts.*

2000 Shriram Krishnamurthi. *Language Technology Reuse.*

1999 Matthew Flatt. *Programming Languages for Reusable Software Components.*

1997 Cormac Flanagan. *Compositional Set-based Analysis for Static Debugging.*

1994 Amr Sabry. *The Formal Relationship between Direct and Continuation-passing Style Optimizing Compilers.*

1994 Andrew Wright. *Practical Soft Typing*. (Joint supervision with Robert Cartwright)

1994 Dorai Sitaram. *Models of Control and Their Implications for Programming Language Design*.

1992 Rebecca Parsons. *A Semantic Framework for Generalized Program Dependence*. (Joint supervision with Robert Cartwright)

M.S. Students

2010 Hari Prashanth K R. *Functional Data Structures in Typed Scheme*.

2011 Yue Zoe Zhang. *An Attempted Proof with Modular ACL2: Soundness of the Racket Bytecode Verifier*.

1990 Erik Crank. *Parameter-Passing Techniques and the λ-Calculus*.

1989 Laura Arbilla. *A Correspondence between Scheme and the λ_v-CS-Calculus*.

1987 John Gateley. *A Call-by-value Combinator Calculus*. (Indiana)

Reader and External Examiner

2003 Danny Dubé. *Demand-Driven Type Analysis for Dynamically-Typed Functional Language.* Université de Montréal.

2000 Arne Kutzner. *Ein nichtdeterministischer "call-by-need" Kalkül mit "erratic choice" Operatoren*. Universität Frankfurt.

1995 Ramarao Kanneganti. *A Universal Domain for Sequential Computation*. Rice University.

1994 Luc Moreau. *Sound Evaluation of Parallel Functional Programs with First-class Continuations*. Université de Liège.

1992 Xavier Leroy. *Typage polymorphe d'un langage algorithmique*. Université Paris 7.

1990 Mike Fagan. *Soft Typing: An Approach to Type Checking for Dynamically Typed Languages*. Rice University.

Invited Conference & Workshop Keynote Lectures

2011 Functional Programming is Easy, and Good for You. NYC, NY. Goldman Sachs TV, November 2011.

2011 Multilingual Component Programming in Racket. Portland, OR. October 2011.

2011 The TeachScheme! Project. At the 2011 SIGCSE Conference, Dallas, TX., March 2011.

2010 The TeachScheme! Project. At the 2010 International Conference on Functional Programming, Baltimore, MD. September 2010.

2009 From Soft Scheme to Typed Scheme: Experience from 20 Years of Script Evolution, and Some Ideas of What Works. At the *2009 Scripts to Programs*, Genoa, Italy, July 2009.

2005 How to Design Class Hierarchies. At the *2005 Functional and Declarative Programming in Education*, Talinn, Estonia, September 2005. (Delivered by M. Flatt)

2005 The First Year. At: *Proc. CCSNE*, Providence, RI, June 2005.

2004 Functional Classes, Functional Objects. At: *2004 European Conference of Object-Oriented Programming*, Oslo, Norway, June 2004.

2002 Next Generation Software Systems and Programming Languages. Northeast Programming Languages Workshop. Keynote talk. IBM Watson, May 2002.

2002 Next Generation Software Systems and Programming Languages. *Simposium Internacional de Sistemas Computacionales*, Monterrey Tech., Mexico City, Mexico. Keynote Address. April 2002.

2002 From POPL to the Classroom and Back. At *Symposium on the Principles of Programming Languages*, Portland, OR, Feb 2002.

2001 "Why are they still teaching Scheme when Haskell is so much better?" At IFIP Working Group 2.3, 2001, Dartmouth.

2000 Static Analysis from one Consumer's Perspective. At *Static Analysis Symposium* 2000, Santa Barbara, California.

1999 The Meaning of Contracts. At *Inaugural Workshop: Preuves, Programmes, et Systemes*, Paris, France.

1997 TeachScheme! – A New Approach in Introductory Computing. At *Frontiers in Engineering*, Pittsburgh, PA.

1994 Extensible Denotational Language Specifications. At *Theoretical Aspects of Computer Software*, Sendai, Japan.

1993 Expressing and Reasoning about State. At *Workshop on State in Programming Languages*, Copenhagen, Denmark.

1992 Full Abstraction and Observable Sequentiality: An Abstract Characterization of Observably Sequential Function Spaces and the Computational Power of SPCF. At *North American Jumelage*. Cornell University, Ithaca, NY.

1992 Sequential PCF: Models and Logics. At *REX Workshop on Semantics—Foundations and Applications*. Beekbergen, the Netherlands.

1992 Full Abstraction and Observable Sequentiality. At *Categorical Logic in Computer Science*. Aarhus, Denmark.