

# Stephanie Weirich

*Curriculum Vitae*

September 5, 2007

---

## CONTACT INFO

Computer and Information Science Department  
Levine Hall, University of Pennsylvania, Philadelphia PA 19104  
Email: [sweirich@cis.upenn.edu](mailto:sweirich@cis.upenn.edu)  
Tel: (215) 573-2821  
Fax: (215) 898-0587  
Homepage: <http://www.cis.upenn.edu/~sweirich>  
Citizenship: USA

## RESEARCH INTERESTS

Programming languages, Type theory, Run-time type analysis, Functional programming, Type inference, Automated proof assistance, Dependent type systems, Metaprogramming, Language-based security.

## EDUCATION

August 2002, Ph.D. Computer Science, Cornell University  
Advisor: Greg Morrisett  
Dissertation title: Programming with Types  
Topic: Runtime type analysis in typed programming languages  
May 2000, M.S. Computer Science, Cornell University  
May 1996, B.A. Computer Science, *magna cum laude*, Rice University

## POSITIONS HELD

July 2002-present, University of Pennsylvania, Philadelphia, Pennsylvania  
Assistant Professor  
August 1996-July 2002, Cornell University, Ithaca, New York  
Instructor, Research Assistant and Teaching Assistant  
June 1999-July 1999, Lucent Technologies, Bell Labs, Murray Hill, New Jersey,  
Summer Intern

## AWARDS

Institute for Defense Analyses Computer Science Study Panel, 2007.  
National Science Foundation CAREER Award, 2003.  
Intel Graduate Student Fellowship, 2000–2001.  
National Science Foundation Graduate Research Fellowship, 1996–1999.  
CRA-W Distributed Mentorship Project Award, 1996.  
Microsoft Technical Scholar, 1995–1996.

## MEMBERSHIP

IFIP Working Group 2.8 (Functional Programming)  
Association for Computing Machinery, SIGPLAN

## GRANTS

- NSF “Collaborative Research: CT-T: Manifest Security” CCF-0716469 University of Pennsylvania. PI: Benjamin Pierce, Co-PIs: Stephanie Weirich, Steve Zdancewic. Carnegie Mellon University. PI: Frank Pfenning, Co-PIs: Robert Harper, Karl Crary. 2007-2011. \$1,000,000.
- NSF “A Practical Dependently-Typed Functional Programming Language” CCF-0702545, PI: Stephanie Weirich. 2007-2009. \$200,000.
- DARPA Computer Science Study Panel RA06-46, Phase I. PI: Stephanie Weirich. 2007-2008. \$99,411.
- NSF “CRI: Machine Assistance for Programming Languages Research” CNS-0551589, PI: Stephanie Weirich, Co-PIs: Benjamin Pierce, Steve Zdancewic. 2006-2008. \$200,000.
- NSF: “CAREER: Type-Directed Programming in Object-Oriented Languages” CCF-0347289, PI: Stephanie Weirich, 2003-2008. \$400,000.

## PROFESSIONAL SERVICE

Program committee member:

International Conf. on Functional Programming (ICFP) 2007.

International Conf. on Aspect-Oriented Software Development (AOSD) 2007.

Haskell Workshop 2007.

ACM Workshop on Types in Language Design and Implementation (TLDI) 2007.

ACM Conf. on Principles of Programming Languages (POPL) 2006.

European Symposium on Programming (ESOP) 2006.

ACM Workshop on Mechanizing Metatheory (Program Chair) 2006.

ML Workshop 2006.

MetaOCaml Workshop 2006.

Foundations of Object-Oriented Languages Workshop (FOOL) 2005.

ACM Conf. on Programming Language Design and Implementation (PLDI) 2004.

Foundations of Global Ubiquitous Computing Workshop (FGUC) 2004.

MetaOCaml Workshop 2004.

International Conf. on Functional Programming (ICFP) 2002.

IFIP TC2 Working Conf. on Generic Programming 2002.

Haskell Workshop 2001.

NSF panel: December 2004.

Journal reviewing: ACM Transactions on Programming Languages and Systems, Acta Informatica, Journal of Functional Programming, Science of Computer Programming, Higher-Order and Symbolic Computation, Journal of Automated Reasoning.

Conference and workshop reviewing: ICFP, POPL, PLDI, LICS, ECOOP, PEPM, FOOL, LCTES, ICALP, TYPES, Haskell.

Member: Haskell’ language standard committee, 2005-present.

TYPES forum moderator: 2003-present.

Workshop organizer: (with B. Pierce and S. Zdancewic) ACM Workshop on Mechanizing Metatheory, 2006, 2007.

Tutorial organizer: (with B. Pierce and S. Zdancewic) Using Proof Assistants for Programming Language Research or, How to write your next POPL paper in Coq, Jan 2008.

ICFP Programming Contest Co-organizer: 2000, 2004.

Panelist: CRA-W/CDC Programming Languages Summer School, UT Austin, May 2007.

Panelist: Women in Science and Engineering Conference, Princeton, February 2006.

## STUDENTS

Dissertation supervision:

Geoffrey Washburn, anticipated graduation date: December 2007.

Dimitrios Vytiniotis, anticipated graduation date: 2008.

Brian Aydemir, anticipated graduation date: 2009.

Thesis committee member:

Dan Dantas, Princeton University, August 2007 (anticipated).

Stephen Tse, August 2007.

Wahnghong Nam, December 2006.

Joeseph Vanderwaart, Carnegie Mellon University, August 2006.

Vladimir Gapayev, January 2006.

Independent study:

Graduate: Andrew Hilton (co-advised), Karl Mazurak, Jeff Vaughan, Fall 2004. Liang Huang, Spring 2004.

Undergraduate: David Gorski, Fall 2006. Parshant Mittal, Atish Davda, Fall 2005. Neal Parikh, Summer 2004.

## TEACHING EXPERIENCE

University of Pennsylvania:

Spring 2007, CSE 120, Programming Languages and Techniques I.

Fall 2006, CIS 700, Mechanizing Programming Language Metatheory.

Spring 2006, CIS 700, Parametric and Ad Hoc Polymorphism.

Fall 2006, CIS 670, Advanced Topics in Programming Languages: Advanced Type Systems.

Fall 2005, CIS 500, Software Foundations.

Spring 2004, CSE 340, Principles of Programming Languages.

Fall 2004, CIS 500, Software Foundations.

Spring 2003, CSE 340, Principles of Programming Languages.

Fall 2002, CIS 670, Advanced Topics in Programming Languages: Parametric and Ad Hoc Polymorphism.

Cornell University:

CS 212, Java Practicum.

CS 213, C++ Programming.

CS 214, A Taste of UNIX and C.

## UNIVERSITY SERVICE

Academic Performance Committee, 2004-present.

Senior Design Project Judge, 2006.

Graduate Admissions Committee, 2003-2004.

Curriculum Committee, 2002-2003.

## JOURNAL PUBLICATIONS

1. Daniel S. Dantas, David Walker, Geoffrey Washburn, and Stephanie Weirich. AspectML: A polymorphic aspect-oriented functional programming language. To appear in *Transactions*

on *Programming Languages and Systems*. 59 pages

2. Geoffrey Washburn and Stephanie Weirich. Boxes go bananas: Encoding higher-order abstract syntax with parametric polymorphism. *Journal of Functional Programming*, 17(6), November 2007. To appear. 56 pages
3. Simon L. Peyton Jones, Dimitrios Vytiniotis, Stephanie Weirich, and Mark Shields. Practical type inference for arbitrary-rank types. *Journal of Functional Programming*, 17(1):1–82, January 2007
4. Stephanie Weirich. Type-safe run-time polytypic programming. *Journal of Functional Programming*, 16(10):681–710, November 2006
5. Stephanie Weirich. Type-safe cast. *Journal of Functional Programming*, 14(6):681–695, November 2004
6. Karl Crary, Stephanie Weirich, and Greg Morrisett. Intensional polymorphism in type erasure semantics. *Journal of Functional Programming*, 12(6):567–600, November 2002

## REFEREED BOOK CHAPTERS

1. Michael Hicks, Stephanie Weirich, and Karl Crary. Safe and flexible dynamic linking of native code. In R. Harper, editor, *Types in Compilation: Third International Workshop, TIC 2000; Montreal, Canada, September 21, 2000; Revised Selected Papers*, volume 2071 of *Lecture Notes in Computer Science*, pages 147–176. Springer, 2001

## REFEREED CONFERENCE PUBLICATIONS

1. Dimitrios Vytiniotis and Stephanie Weirich. Free theorems and runtime type representations. In *Mathematical Foundations of Programming Semantics (MFPS XXIII)*, pages 357–373, New Orleans, LA, USA, April 2007
2. Simon L. Peyton Jones, Dimitrios Vytiniotis, Stephanie Weirich, and Geoffrey Washburn. Simple unification-based type inference for GADTs. In *International Conference on Functional Programming (ICFP)*, pages 50–61, Portland, OR, USA, September 2006
3. Dimitrios Vytiniotis, Stephanie Weirich, and Simon L. Peyton Jones. Boxy type inference for higher-rank types and impredicativity. In *International Conference on Functional Programming (ICFP)*, pages 251–262, Portland, OR, USA, September 2006
4. Daniel S. Dantas, David Walker, Geoffrey Washburn, and Stephanie Weirich. PolyAML: A polymorphic aspect-oriented functional programming language. In *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, pages 306–319, Tallinn, Estonia, September 2005
5. Brian E. Aydemir, Aaron Bohannon, Matthew Fairbairn, J. Nathan Foster, Benjamin C. Pierce, Peter Sewell, Dimitrios Vytiniotis, Geoffrey Washburn, Stephanie Weirich, and Steve Zdancewic. Mechanized metatheory for the masses: The POPLmark challenge. In *The 18th International Conference on Theorem Proving in Higher Order Logics (TPHOLs)*, pages 50–65, Oxford, UK, August 2005

6. Geoffrey Washburn and Stephanie Weirich. Generalizing parametricity using information flow. In *IEEE Symposium on Logic in Computer Science (LICS)*, pages 62–71, Chicago, IL, USA, June 2005
7. Geoffrey Washburn and Stephanie Weirich. Boxes go bananas: Encoding higher-order abstract syntax with parametric polymorphism. In *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, pages 249–262, Uppsala, Sweden, August 2003
8. Stephanie Weirich. Higher-order intensional type analysis. In Daniel Le Métayer, editor, *11th European Symposium on Programming (ESOP)*, pages 98–114, Grenoble, France, April 2002
9. Stephanie Weirich. Encoding intensional type analysis. In D. Sands, editor, *10th European Symposium on Programming (ESOP)*, pages 92–106, Genova, Italy, April 2001
10. Stephanie Weirich. Type-safe cast: Functional pearl. In *Proceedings of the Fifth ACM SIGPLAN International Conference on Functional Programming (ICFP)*, pages 58–67, Montreal, Canada, September 2000
11. Karl Crary and Stephanie Weirich. Resource bound certification. In *Twenty-Seventh ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, pages 184–198, Boston, MA, USA, January 2000
12. Karl Crary and Stephanie Weirich. Flexible type analysis. In *Proceedings of the Fourth ACM SIGPLAN International Conference on Functional Programming (ICFP)*, pages 233–248, Paris, France, September 1999
13. Karl Crary, Stephanie Weirich, and Greg Morrisett. Intensional polymorphism in type erasure semantics. In *Proceedings of the Third ACM SIGPLAN International Conference on Functional Programming (ICFP)*, pages 301–313, Baltimore, MD, USA, September 1998
14. Cormac Flanagan, Matthew Flatt, Shriram Krishnamurthi, Stephanie Weirich, and Matthias Felleisen. Catching bugs in the web of program invariants. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)*, pages 23–32, 1996

## REFEREED WORKSHOP PUBLICATIONS

1. Dimitrios Vytiniotis and Stephanie Weirich. Dependent types: Easy as PIE. In Marco T. Morazán and Henrik Nilsson, editors, *Draft Proceedings of the 8th Symposium on Trends in Functional Programming*, pages XVII–1—XVII–15. Dept. of Math and Computer Science, Seton Hall University, April 2007. TR-SHU-CS-2007-04-1
2. Stephanie Weirich. RepLib: A library for derivable type classes. In *Haskell Workshop*, pages 1–12, Portland, OR, USA, September 2006
3. Geoffrey Washburn and Stephanie Weirich. Good advice for type-directed programming: Aspect-oriented programming and extensible generic functions. In *Workshop on Generic Programming (WGP)*, pages 33–44, Portland, OR, USA, September 2006
4. Brian Aydemir, Aaron Bohannon, and Stephanie Weirich. Nominal reasoning techniques in Coq. In *International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice (LFMTP)*, pages 60–69, Seattle, WA, USA, August 2006

5. Benjamin C. Pierce, Peter Sewell, Stephanie Weirich, and Steve Zdancewic. It is time to mechanize programming language metatheory. In *Verified Software: Theories, Tools, Experiments (VS:TTE)*, Zürich, Switzerland, October 2005. 5 pages
6. Dimitrios Vytiniotis, Geoffrey Washburn, and Stephanie Weirich. An open and shut typecase. In *ACM SIGPLAN Workshop on Types in Language Design and Implementation (TLDI)*, pages 13–24, Long Beach, CA, USA, January 2005
7. Stephanie Weirich and Liang Huang. A design for type-directed Java. In Viviana Bono, editor, *Workshop on Object-Oriented Developments (WOOD)*, ENTCS, pages 117–136, 2004
8. Greg Morrisett, Karl Crary, Neal Glew, Dan Grossman, Richard Samuels, Frederick Smith, David Walker, Stephanie Weirich, and Steve Zdancewic. TALx86: A realistic typed assembly language. In *Second ACM SIGPLAN Workshop on Compiler Support for System Software*, pages 25–35, Atlanta, GA, USA, May 1999. Published as INRIA research report number 0228, March 1999

## THESES

1. Stephanie Weirich. *Programming With Types*. PhD thesis, Cornell University, August 2002

## TECHNICAL REPORTS

1. Karl Crary, Robert Harper, Frank Pfenning, Benjamin C. Pierce, Stephanie Weirich, and Stephan Zdancewic. Manifest security. Technical report, January 2007. White paper
2. Dimitrios Vytiniotis, Stephanie Weirich, and Simon L. Peyton Jones. Boxy type inference for higher-rank types and impredicativity, Technical Appendix. Technical Report MS-CIS-05-23, University of Pennsylvania, April 2006
3. Dimitrios Vytiniotis, Stephanie Weirich, and Simon L. Peyton Jones. Simple unification-based type inference for GADTs, Technical Appendix. Technical Report MS-CIS-05-22, University of Pennsylvania, April 2006
4. Simon L. Peyton Jones, Dimitrios Vytiniotis, Stephanie Weirich, and Mark Shields. Practical type inference for arbitrary-rank types (technical appendix). Technical Report MIS-CIS-05-14, University of Pennsylvania, July 2005
5. Geoffrey Washburn and Stephanie Weirich. Generalizing parametricity using information flow (extended version). Technical Report MS-CIS-05-04, Computer and Information Science, University of Pennsylvania, July 2005
6. Simon L. Peyton Jones, Geoffrey Washburn, and Stephanie Weirich. Wobbly types: Practical type inference for generalised algebraic datatypes. Technical Report MS-CIS-05-26, University of Pennsylvania, Computer and Information Science Department, Levine Hall, 3330 Walnut Street, Philadelphia, Pennsylvania, 19104-6389, July 2004
7. Dan S. Dantas, David Walker, Geoffrey Washburn, and Stephanie Weirich. Analyzing polymorphic advice. Technical Report TR-717-04, Princeton University Computer Science, December 2004

8. Liang Huang and Stephanie Weirich. A design for type-directed programming in Java (extended version). Technical Report MS-CIS-04-11, University of Pennsylvania, Computer and Information Science, October 2004
9. Dimtrios Vytiniotis, Geoffrey Washburn, and Stephanie Weirich. An open and shut typecase (extended version). Technical Report MS-CIS-04-26, University of Pennsylvania, Computer and Information Science, October 2004
10. Geoffrey Washburn and Stephanie Weirich. Boxes go bananas: Encoding higher-order abstract syntax with parametric polymorphism (extended version). Technical Report MS-CIS-03-26, University of Pennsylvania, Computer and Information Science, September 2003
11. Karl Crary, Stephanie Weirich, and Greg Morrisett. Intensional polymorphism in type erasure semantics (extended version). Technical Report TR98-1721, Cornell University, Computer Science, November 1998
12. Michael Hicks and Stephanie Weirich. A calculus for dynamic loading. Technical Report MS-CIS-00-07, University of Pennsylvania, April 2000

## TECHNICAL PRESENTATIONS

1. *Formal Reasoning About Programs and Programming Languages*. National Security Agency. Fort Meade, MD, USA. July 20, 2007.
2. *Engineering Aspects of Formal Metatheory*. Harvard University, Boston MA, USA. June 1, 2007.
3. *Getting started in PL design research*. CRA-W/CDC Programming Languages Summer School, UT Austin, May 2007.
4. *Career paths: How to get started in academia or industry*. CRA-W/CDC Programming Languages Summer School, UT Austin, May 2007.
5. *Dependently-Typed Languages*. Working session summary. IFIP WG 2.11, Portland, OR, October 2006.
6. *Simple Unification-Based Type Inference for GADTs*. International Conference on Functional Programming (ICFP). Portland, OR. September 2006.
7. *RepLib: A Library for Derivable Type Classes*. Haskell Workshop. Portland, OR. September 2006.
8. *Parametricity and GADTs*. IFIP Working Group 2.8 (Functional Programming). Boston, MA. July 2006.
9. *Practical Type Inference for Advanced Type Systems*. International Federation for Information Processing (IFIP) Working Group 2.11, Dagstuhl, Wadern, Germany. January 2006.
10. *Boxy Types: Inference for Higher-rank Types and Impredicativity*. International Federation for Information Processing (IFIP) Working Group 2.8, Kalvi Manor, Estonia. October 2005.

11. *A Core Language for Generalised Algebraic Datatypes*. International Federation for Information Processing (IFIP) Working Group 2.8, West Point, USA. November 2004.
12. *A Design for Type-directed Java*. Programming Languages Seminar, Yale University, New Haven, CT. October 1, 2004.
13. *2004 ICFP Programming Contest Results*. (Presented jointly with Benjamin Pierce and Steve Zdancewic) International Conference on Functional Programming, Snowbird, UT. September 20, 2004.
14. *A Core Language for Generalised Algebraic Datatypes*. Dagstuhl Seminar 04381: Dependently Typed Programming, Wadern, Germany. September 12, 2004.
15. *A Design for Type-Directed Java*. Microsoft Research Lab, Cambridge, UK. August 31, 2004.
16. *A Design for Type-Directed Java*. Workshop on Object-Oriented Developments (WOOD '04). London, UK, August 2004.
17. *Unifying Nominal and Structural Ad-hoc Polymorphism*. International Federation for Information Processing (IFIP) Working Group 2.8, Coffs Harbour, Australia. January 2003.
18. *Unifying Nominal and Structural Ad-hoc Polymorphism*. Computer Science Colloquium, City University of New York Graduate Center. New York, NY. October 30, 2003.
19. *Boxes Go Bananas: Parametric Higher-Order Abstract Syntax in System F*. Laboratory for Secure Systems Seminar, Stevens Institute of Technology. Hoboken, NJ. May 5, 2003.
20. *Run-time type analysis in Haskell with an Awful Lot of Newtypes*. International Federation for Information Processing (IFIP) Working Group 2.8, Crans-Montana, Switzerland. January 2003.
21. *Polytypic Programming and Intensional Type Analysis*. New Jersey Programming Languages Seminar. University of Pennsylvania, Philadelphia, PA. September 20, 2002.
22. *Programming with Types*. OHSU/Oregon Graduate Institute, Beaverton, OR. February 11, 2002.
23. *Programming with Types*. University of Oregon, Eugene, OR. February 15, 2002.
24. *Programming with Types*. University of Pennsylvania, Philadelphia, PA. February 19, 2002.
25. *Programming with Types*. University of Virginia, Charlottesville, VA. February 28, 2002.
26. *Programming with Types*. University of Maryland, College Park, MD. March 4, 2002.
27. *Programming with Types*. Northeastern University, Boston, MA. March 13, 2002.
28. *Programming with Types*. University of California, San Diego, CA. March 15, 2002.
29. *Programming with Types*. Purdue University, West Lafayette, IN. March 25, 2002.
30. *Programming with Types*. University of Michigan, Ann Arbor, MI. March 27, 2002.
31. *Programming with Types*. University of Texas, Austin, TX. April 2, 2002.

32. *Higher-order Intensional Type Analysis*. European Symposium on Programming (ESOP '02). Grenoble, France, April 2002.
33. *Programming with Types*. University of Colorado at Boulder, CO. April 16, 2002.
34. *Programming with Types*. Pennsylvania State University, State College, PA. April 19, 2002.
35. *Programming with Types*. Massachusetts Institute of Technology, Boston, MA. April 25, 2002.
36. *Programming with Types*. Rice University, Houston TX. April 29, 2002.
37. *Run-Time Type Analysis and Program Verification*. Research, Careers and Computer Science: A Maryland Symposium. University of Maryland, College Park, MD. November 2001.
38. *Polytypic Programming and Intensional Type Constructor Analysis*. International Federation for Information Processing (IFIP) Working Group 2.8, Åre, Sweden. April 2001.
39. *Encoding Intensional Type Analysis*. European Symposium on Programming (ESOP '01). Genova, Italy. April 2001.
40. *Resource Bound Certification*. Harvard University, Boston, MA. February 2001.
41. *Functional Pearl: Type-Safe Cast*. International Conference on Functional Programming. Montreal, Canada. September 2000.
42. *Resource Bound Certification*. IBM Research, Hawthorne, NY. June 2000.
43. *Resource Bound Certification*. ACM Symposium on Principles of Programming Languages (POPL '00). Boston, MA, USA. January 2000.
44. *Flexible Type Analysis*. International Conference on Functional Programming (ICFP '99). Paris, France, September 1999.
45. *Type Analysis and Typed Compilation*. Princeton University, Princeton, NJ. June 1999.
46. *Intensional Polymorphism in Type-Erasure Semantics*. International conference on Functional Programming (ICFP '98). Baltimore, MD, USA, September 1998.