Jeremy Condit

jcondit@cs.berkeley.edu
565 Soda Hall, Berkeley, CA 94720
(510) 459-6269

Education

University of California, Berkeley (2001 - Present)
Advisor: Prof. George Necula, GPA: 4.0

- Ph.D. in Computer Science, expected Summer 2007
  Thesis: *Extensible Dependent Types for Low-Level Programming*
- M.S. in Computer Science, June 2004
  Thesis: *Data Slicing: Separating the Heap into Independent Regions*

Harvard University (1996 - 2000)
Advisor: Prof. Margo Seltzer, GPA: 3.96

- A.B. summa cum laude in Computer Science, June 2000
  Thesis: *Performance of Memory-Based and Asynchronous File Systems Under FreeBSD and Solaris*

Research Interests

Deputy: Extensible Dependent Types for Low-Level Programming
Dependent types are types that can depend upon the run-time value of a program expression. These types are powerful and expressive but are difficult to reason about at compile time. My research focuses on ways to make dependent types practical for use in low-level languages—and in particular, the C programming language—by using run-time checks as well as a new approach to analyzing dependencies in the presence of mutation. Using these dependent types, C programmers can annotate existing features of their code, including bounded pointers and tagged unions, allowing the compiler to enforce stronger safety properties. Our implementation, Deputy, has been used on a number of real-world C programs including Linux device drivers and the Linux kernel itself. Deputy is also a key component of the SafeDrive recovery system for Linux device drivers.

Capriccio: Scalable Threads for Internet Services
The Capriccio project argues that threads, rather than events, are the ideal programming model for high performance Internet servers. Previous research suggested that event-driven code was more efficient; however, we observed an important duality between thread-based and event-based systems that implies that threads can achieve performance equal to events (and vice versa). The Capriccio thread package is a proof of concept designed to demonstrate that threads can scale as well as events. Compiler and language support is critical in ensuring that stacks are allocated in a scalable manner.

Data Slicing: Separating the Heap into Independent Regions
Data slicing is a program transformation technique that allows the compiler to automatically separate data structures in a type-directed manner, mirroring pointer structure while separating non-pointer data. This technique was used in the CCured project, which added metadata to C programs in order to support run-time memory safety checks. Data slicing allowed CCured to track program metadata efficiently without altering the layout of data structures in memory.

Refereed Publications

Jeremy Condit, Matthew Harren, Zachary Anderson, David Gay, and George C. Necula. *Dependent Types for Low-Level Programming.* European Symposium on Programming (ESOP) 2007. (To appear.)

Feng Zhou, Jeremy Condit, Zachary Anderson, Ilya Bagrak, Rob Ennals, Matthew Harren, George C. Necula, and Eric Brewer. *SafeDrive: Safe and Recoverable Extensions Using Language-Based Techniques.* Operating System Design and Implementation (OSDI) 2006.

Eric Brewer, Jeremy Condit, Bill McCloskey, and Feng Zhou. *Thirty Years is Long Enough: Getting Beyond C.* Hot Topics in Operating Systems (HotOS) 2005.

George C. Necula, Jeremy Condit, Matthew Harren, Scott McPeak, and Westley Weimer. *CCured: Type-Safe Retrofitting of Legacy Software.* Transactions on Programming Languages and Systems (TOPLAS) 27:3 (May 2005).

Jeremy Condit and George C. Necula. *Data Slicing: Separating the Heap into Independent Regions.* Compiler Construction (CC) 2005. (European Association for Programming Languages and Systems (EAPLS) Best Paper Award for ETAPS 2005.)

Rob von Behren, Jeremy Condit, Feng Zhou, George C. Necula, and Eric Brewer. *Capriccio: Scalable Threads for Internet Services.* Symposium on Operating System Principles (SOSP) 2003.

Jeremy Condit, Matthew Harren, Scott McPeak, George C. Necula, and Westley Weimer. *CCured in the Real World.* Programming Language Design and Implementation (PLDI) 2003.

Rob von Behren, Jeremy Condit, and Eric Brewer. *Why Events are a Bad Idea (for high-concurrency servers).* Hot Topics in Operating Systems (HotOS) 2003.

Donald R. Latner, Ying Xiang, Jackie I. Lewis, Jeremy Condit, and Richard Condit. *The Vaccinia Virus Bifunctional Gene J3 (Nucleoside-2'-O-)-methyltransferase and Poly(A) Polymerase Stimulatory Factor Is Implicated as a Positive Transcription Elongation Factor by Two Genetic Approaches.* Virology. 269(2): 345-355, 2000.

Technical Reports

Jeremy Condit, Matthew Harren, Zachary Anderson, David Gay, and George C. Necula. *Dependent Types for Low-Level Programming.* UC Berkeley Technical Report EECS-2006-129 (October 2006).

Jeremy Condit, James R. Larus, Sriram K. Rajamani, and Jakob Rehof. *Region-Based Model Abstraction.* Microsoft Research Technical Report MSR-TR-2003-47 (August 2003).

Honors and Awards

EAPLS Best Paper Award for ETAPS (2005).
NSF Graduate Research Fellowship (2001).
Phi Beta Kappa, Harvard College (1999).
Detur Book Prize, Harvard College (1997).

Work Experience

Microsoft: Research Intern (Summer 2003 & 2004)
Worked with Galen Hunt and Jim Larus on the Singularity project. Designed and implemented a lightweight threading system based on ideas from the Capriccio project. Also worked with Jakob Rehof and Jim Larus on region-based model abstraction, which uses region inference to assist in generating models for use with Microsoft's software model checker.

Tellme Networks: Software Engineer (2000 - 2001)
Designed and implemented an interpreter for Voice XML, which is used to program a scalable and highly concurrent telephony platform.

Microsoft: Software Engineer Intern (Summer 1998 & 1999)
Added an extensibility feature to Microsoft Word for the Macintosh, and created a plug-in architecture for use with Windows 2000 Kernel Streaming technology.

References

George Necula
783 Soda Hall
Berkeley, CA 94720
necula@cs.berkeley.edu
(510) 643-1481

James Larus
Microsoft Research
One Microsoft Way
Redmond, WA 98052
larus@microsoft.com
(425) 706-2981

Eric Brewer
623 Soda Hall
Berkeley, CA 94720
brewer@cs.berkeley.edu
(510) 642-8143

Rastislav Bodik
773 Soda Hall
Berkeley, CA 94720
bodik@cs.berkeley.edu
(510) 642-2488