JEREMY H. BROWN

jhbrown@repinvariant.com
(617) 233-6109

Fields of Interest

Autonomous robotics; software architecture; computer systems architecture; parallel and distributed systems; programming languages.

Education

Massachusetts Institute of Technology Cambridge, MA
June, 2002

PhD, Electrical Engineering and Computer Science, with minor in control theory. Dissertation: "Sparsely Faceted Arrays: A Mechanism Supporting Parallel Allocation, Communication, and Garbage Collection;" see brief abstract, attached.

Massachusetts Institute of Technology Cambridge, MA
June, 1995

MEng and BS in EECS. Dissertation: "Feedback-Directed Specialization of C."

Industry Experience

Rep Invariant Systems, Inc Cambridge, MA
September 2007 - Present

Co-founder and President / Engineer. Rep Invariant (RI) is a startup developing software for the robotics industry. First product is a JAUS SDK, part of a larger Autonomous Software Integration Toolkit (ASIT) still under development. Primarily in C++, supported by various scripting languages. RI also provides robotics consulting services.

Personal responsibilities include business planning and development, finance, marketing, and engineering.

ITA Software Cambridge, MA
January 2006-September 2007

Senior Software Engineer and Architect on project to replace Air Canada's existing mainframe reservations, ticketing, and departure control (airport) systems with completely rewritten software running on modern, commodity hardware.

Worked in infrastructure and architecture groups responsible for non-functional requirements, and for supporting feature-driven groups. Coded primarily in Common Lisp, on Linux, with substantial use of Oracle/SQL. Extensive work on object-relational mapping layer, and troubleshooting robustness, concurrency, and performance issues discovered by ongoing system tests.

One of three architects on the Reservations Architecture Board (RAB). Led extensive team design work on issues including transactional correctness, lease-based resource allocation, and data lifecycle management. Documented decisions and rationales for each issue, with UML diagrams where appropriate.

Bluefin Robotics Corp. Cambridge, MA
November 2002-July 2005

Senior Software Engineer, and Lead Developer for Sealion project. Worked on autonomous underwater vehicles (AUVs.) Worked with team on complete redesign and reimplementation of all vehicle code, including migration from QNX to Debian Linux (with "realtime" patches.) Continuous development of framework, control, AI behaviors, device-integration, and operator GUI tools. Lead developer for Sealion AUVs, the first vehicles to deploy the new codebase. Extensive field experience with Sealion and other vehicles. Extensive system integration and debugging experience. Coding primarily in C++, with Qt for GUIs; scripting primarily in Python.

Zoom Telephonics Boston, MA
August 2001

Debugged Linux device driver and ported it from 2.2 to 2.4 kernel architecture.

Newton Research Labs Seattle, WA
August 1995 – August 1997

Frequent consulting. Designed and implemented hardware and software interfacing portable real-time vision systems to small, autonomous, high-speed land and air vehicles; provided systems troubleshooting during robotics competition.

Zoom Telephonics Boston, MA
August 1999

Java contracting: provided support for recently-acquired product during transition.

Xtigent Boston, MA
March 1999 – May 1999

Java contracting: authored distributed, replicated, database-backed, prioritized trouble-ticket tracking system using RMI.

Virtual Emporium Cambridge, MA
November 1997 – December 1997

Java contracting: authored generic framework for parsing stylized web-pages and extracting specific pieces of data; also authored tool for rapidly constructing parsing templates using sample web pages.

MIT LCS, Computer Resource Services Cambridge, MA
January 1991 – October 1992

Systems support programming. Authored remote-access console-server; concluded development of a secure, unified user-information and mail-forwarding service.

Research Experience

MIT AI Lab, Aries Group Cambridge, MA
August 1996 – August 2002

Studied issues in parallel computer system design and programmability. Designed Sparsely Faceted Arrays (SFAs), a parallel data structure; a parallel garbage collection strategy for managing SFAs; and a set of simple hardware mechanisms supporting SFAs and garbage collection. Implemented all in simulation using C, GTK, and Scheme, on UNIX and Linux workstations. Additionally, developed idempotent messaging protocol for fault-tolerant networks; and wrote a functional simulator for an advanced Cellular Automata Machine (CAM) (see Norman Margolus, "An embedded DRAM architecture for large-scale spatial-lattice computations," ISCA 2000.)
Advisor: Tom Knight.

MIT AI Lab, Reinventing Computing Group Cambridge, MA
September 1994 – July 1996

Implemented several profile-based compile-time optimizations for C programs, using the SUIF compiler framework. Collaborated in the design and layout of a Dynamically Programmable Gate Array (DPGA) (US pat. #5742180). Managed development of HyperCode, a gcc-based system to generate hyperlinked source code as a side-effect of compilation.
Advisor: Tom Knight.

MIT AI Lab, Transit Project Cambridge, MA
October 1993 – September 1994

Collaborated in the design of a VLSI standard cell library. Authored an active-message software layer

for Transit parallel computer.
Advisor: Tom Knight.

MIT Media Lab, Project ALIVE Cambridge, MA
Summer 1993

Wrote AI behaviors and motion-control for synthetic creatures in a non-intrusive virtual-reality setting. Collaborated in interfacing vision-system to behavior code.
Advisor: Pattie Maes.

MIT AI Lab, Math and Computation Cambridge, MA
October 1992 – January 1993

Implemented simple virtual reality demonstration using VPL Eyephone goggles, Polhemus sensors, and HP Starbase graphics under X windows.
Advisor: Hal Abelson.

Teaching Experience

MIT EECS Department Cambridge, MA
Spring 2002

Teaching assistant for 6.004, Computation Structures, an introductory computer architecture and systems course. Taught recitations, provided assistance in lab, graded projects, and administered and graded tests for approximately 50 students.

Spring 2000

Co-designer and instructor for 6.911, a reading and discussion seminar on unorthodox computer architectures. Selected papers, presented several topics, and guided discussion for approximately 50 people (about 20 registered students.)

Fall 1996

Teaching assistant for 6.035, Computer Language Engineering, a course on compiler design and implementation. Taught recitations, and graded for approximately 20 students.

Fall 1993

Undergraduate teaching assistant for 6.312, Acoustics; taught tutorials and graded for approximately 15 students.

Publications

Christopher Smith, Jeremy Brown, Matt Zucker, Nathaniel Fairfield, Reed Christenson, Gil Jones. “The Bluefin Software Architecture for Autonomous Underwater Vehicles,” Proceedings of the Association for Unmanned Vehicle Systems International (AUVSI) Symposium, poster session only, July 2003.

Jeremy Brown, J.P. Grossman, Thomas F. Knight, Jr. “A Lightweight Idempotent Messaging Protocol for Faulty Networks,” Proceedings of the Fourteenth ACM Symposium on Parallel Algorithms and Architectures (SPAA), pp. 249-257, August 2002.

Jeremy Brown. “Sparsely Faceted Arrays: A Mechanism Supporting Parallel Allocation, Communication, and Garbage Collection,” PhD Thesis AI-TR-2002-005, June 2002.

Jeremy Brown and Ian Eslick. “Implementing continuous, profile-based optimizations in SUIF,” the First SUIF Compiler Workshop, January 1996.

Edward Tau, Ian Eslick, Derrick Chen, Jeremy Brown, and Andre DeHon. “A First Generation DPGA Implementation,” Proceedings of the Third Canadian Workshop on Field-Programmable Devices (FPD’95), May, 1995.

Pattie Maes, Sandy Pentland, Bruce Blumberg, Trevor Darrell, Jeremy Brown, and Johnny Yoon. “ALIVE: Artificial Life Interactive Video Environment,” Intercommunication, Vol. 7, pp. 48-49, Winter 1994.

Jeremy Brown, Jake Harris, Lara Karbiner, Massimiliano Poletto, Andre DeHon, Ian Eslick, and Thomas F. Knight Jr. “HyperCode,” Second International World Wide Web Conference, October 1994.

Andre DeHon, Jeremy Brown, Ian Eslick, Jake Harris, Lara Karbiner, and Thomas F. Knight, Jr. “Global Cooperative Computing,” Second International World-Wide Web Conference 1994, October 1994.

Selected Manuscripts

Jeremy Brown and Thomas F. Knight, Jr. “A Minimal Trusted Computing Base for Dynamically Ensuring Secure Information Flow,” Project Aries TM-015, November, 2001.

Jeremy Brown, J.P. Grossman, Andrew Huang, and Tom Knight.“A Capability Representation with Embedded Address and Nearly-Exact Object Bounds,” Project Aries TM-005, April 2000.

Charles Coffing and Jeremy Brown. “A System for Transparent File Compression With Caching Under Linux,” 1997.

Patents

Jeffrey P. Grossman, Thomas F. Knight, Jr., Jeremy H. Brown, Andrew W. Huang. “Mechanism to Reduce the Cost of Forwarding Pointer Aliasing,” US Patent No. 7,107,430, 2006.

Jeremy H. Brown, Thomas F. Knight, Jr., Jeffrey P. Grossman, Andrew W. Huang. “Capability Addressing with Tight Object Bounds,” US Patent No. 6,826,672, 2004.

Andre DeHon, Thomas F. Knight, Jr., Edward Tau, Michael Bolotski, Ian Eslick, Derrick Chen, Jeremy Brown. “Dynamically programmable gate array with multiple contexts,” US Patent No. 5,742,180, 1995.

Additional Technical Projects

MIT Aerial Robotics Club *1996-1997*

Worked on robotic helicopter used in autonomous flying vehicle competition. Designed and implemented hardware and software interfacing several sensors to onboard PC104 x86 stack, including a Newton Labs vision system and a GEC-Marconi inertial guidance system. Configured and maintained QNX real-time OS installation.

MIT Rocket Team *1999-2001*

Numerous tasks testing custom rocket engine design intended for use in Cheap Access to Space (CATS) competition. Compiled and managed equipment and procedures checklists, significantly improving team efficiency during engine tests.

PhD Thesis Brief Abstract

In my dissertation, I introduce *Sparsely Faceted Arrays* (SFAs), a low-level mechanism for naming regions of memory, or facets, on different processors in a distributed, shared memory parallel processing system. A sparsely faceted array names a *virtual* globally-distributed array whose *actual* facets are lazily allocated. By providing simple semantics and making efficient use of memory, SFAs enable efficient implementation of a variety of non-uniformly distributed data structures and related algorithms.

Because facets are dynamically allocated, SFAs have complex internal structure. I present a parallel memory-management strategy, based on reference-counting, that is capable of garbage collecting sparsely faceted arrays. I justify the use of a reference-counting strategy by arguing that conventional tracing techniques such as mark/sweep and copying GC are inherently unscalable in parallel systems.