Chuck Connell

128 Great Road, Bedford, MA 01730
781-275-0484, connell@chc-3.com

Broad experience with all aspects of software development across many platforms and programming languages. I am a skilled programmer, expert designer and accomplished team leader. Significant recent work with Hadoop – HDFS, Python MapReduce, Hive, Sqoop. I have taught data structures and software engineering at Boston University, published 50 technical articles and spoken at industry conferences. I hold a B.A. in linguistic theory and an M.A. in computer science and have completed 12 post-master's CS courses toward a Ph.D.

Seeking a leadership position as a senior architect or development manager.

Skills

- Programming: Java, Python, C, OO design, refactoring, SQL, C#, Fortran, Intel assembly, APL
- Computer science: algorithms and complexity, software engineering methods, database theory and optimization, cryptography and security, data communication and networking
- Leadership: design/architecture, project management, agile methods, CMM, planning and scheduling, code reviews, quality control, troubled project turnaround
- Platforms/tools: Hadoop (HDFS, MapReduce, Python coding, Hive, Sqoop), RDBMS, Eclipse, NetBeans, Visual Studio, ETL, .Net, soapUI, InstallShield
- OS: Windows XP/7/Server, Linux server/workstation, VMware Server/ESXi, embedded micro-kernels, cloud services
- IBM/Lotus: Notes, Domino, Connections, Sametime, LotusLive, Traveler, LotusScript, Domino Designer, Java and C APIs, server clusters, replication, security, email routing, applications

Professional Experience

Independent Consultant, 1995 to present. Practice focuses on custom software development for corporate clients including IBM, Bausch & Lomb, Alcoa, Mead Johnson, Hewlett-Packard and the U.S. federal court system. I have completed more than 75 projects, such as:

- Nuance Communication – Analyzed and solved a big data problem. Chose and prototyped Hadoop HDFS, MapReduce, Hive, Sqoop, Python job streaming, CentOS and Cloudera CDH. Wrote technical papers with results.

- Greater Boston Legal Service – Designed and coded enterprise-grade legal services application spanning multiple locations. Technologies included OO design, ETL of existing data, mirrored server clusters, NoSQL databases, public/private key security, binary to XML data conversion, and server consolidation with VMware and Ubuntu.

- Hewlett-Packard – Complex Java coding with database APIs. Technologies included Eclipse, OO design, NoSQL middleware and merging of several security directories.

- Bacardi Ltd – Security audit for multinational company, including user and organization policies, password expiration, root certificates, global server replication and two-factor authentication.
- Bausch & Lomb – Several projects including email routing troubleshooting and repair, migration of document management from Lotus Notes to EMC Documentum, and programmatic generation of PDF and Excel file formats.

Boston University, Computer Science Instructor, various semesters to present. I wrote and taught two courses:

- Software Engineering – requirements analysis, module design, agile methods, open source, programming style, estimating and tracking, quality assurance, release management
- Data Structures – lists, queues, rings, stacks, algorithm design

Lotus/IBM, Development Manager and Principal Engineer, 1990 to 1995. I led the design and development of the multiplatform C-language database API for Lotus Notes. I managed six engineers (responsible for technical leadership and performance reviews) and represented our product at industry events, giving many technical talks. Technologies included C, Visual Studio, Windows, Macintosh and UNIX.

Education

Tufts University, post-master’s certificate in computer science. Courses on computability, algorithms, compilers and software engineering.

Boston University, eight post-master’s courses. Topics included cryptography, programming language syntax/semantics, networking/data communication and an advanced complexity seminar.

Boston University, M.A. in computer science. Courses on operating systems, analysis of algorithms, database theory, artificial intelligence, software engineering and complexity. Thesis on approximation algorithms for NP-hard problems.

Hampshire College, B.A. in linguistic theory. Senior thesis on formal semantics for natural language.

Sample Publications *(full list and links at chc-3.com/pub/pubs.htm)*

Hadoop as a Regular Database. The Hive project, built on top of HDFS and MapReduce, is a data warehouse that scales large and is fast at adding new records, It also contains easy-to-use tools for querying, organizing and extracting existing data. This article provides a tutorial for Hive, showing how a few simple commands create vast power to store and extract huge amounts of data.

Hadoop from Scratch: Bare Metal to Python MapReduce Jobs. Tutorial explaining how to build a complete Hadoop ecosystem (HDFS, MapReduce, Hive, Pig, etc.) on a development/test server, including a template for streaming Python scripts as MapReduce jobs. I used CentOS and the Cloudera CDH distribution of Hadoop.

Beautiful Software (book). From the introduction ... *Software quality matters. Software runs our banking operations, air traffic control, stock markets, personal information privacy and many other facets of our lives. Good software helps all of these things run smoothly; bad software has the potential to hurt or even kill people. But what, exactly, distinguishes good software from bad software?* The book addresses this question and is written for anyone in the computer field or related areas – programmers, managers, investors, engineers, scientists.

Why Software Really Fails, And What to Do About It. Dr. Dobb's Journal. It is not news that software projects fail more often than other kinds of engineering. But why? In this essay, I maintain that we don't understand the true nature of software. Software is a machine, but we don't apply standard, well-known machine design principles to our software projects.

Is Software Patentable? IpWatchdog.com. I weigh in on the debate about whether software is inherently patentable and argue that *of course* it is.

It's Not About Lines of Code. Developer.com and Slashdot.org. What makes a programmer highly productive? Is it lines of code per day? Lines of good code? In this article I examine the concept of software productivity. I look at some of the standard definitions for productivity and show why they are wrong. I then propose a new definition that captures what programming really is about.

Why Software Engineering Is Not B.S. Dr Dobb's Journal. Examines the status of software engineering within the field of computer science. Argues that software engineering's low standing is not justified and explains why it should be taken more seriously.

Analysis of Crypto-System Software DES. Cryptologia. Security analysis of a software version of the encryption/decryption scheme Data Encryption Standard.

Approximation Algorithms for NP-Hard Problems (master's thesis). Begins with a review of complexity theory and then focuses on one particularly interesting NP-Hard problem and an approximation algorithm for it.

Sample Talks *(full list and links at chc-3.com/talk/talks.htm)*

Healing Sick Software Projects — A talk aimed at project managers and technical leads about how to turn around software projects that are in crisis.

Why Software Is (Almost) Always Late — A description of my top six reasons why software projects so often run long, and what to do about them.