

Chuck Connell

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

Twenty-five years of experience with all aspects of software development, across many platforms and programming languages. I am a skilled programmer, expert designer and capable team leader. I have taught software engineering and data structures at Boston University, written more than 50 technical articles on a wide range of topics and spoken at many industry conferences. I hold a B.A. in linguistic theory and an M.A. in computer science and have completed 12 post-master courses.

Seeking a position as a senior architect, technical team lead or development manager.

Skills

- Programming: Java, C, C#, OO design, refactoring, SQL, Intel assembly, PL/1, APL
- Leadership: design/architecture, agile methods, planning and scheduling, code reviews, quality control, project management, troubled project turnaround
- OS: Windows XP/7, Windows Server 03/08, Ubuntu Linux 8 server and workstation, VMware Server, VMware ESXi, embedded micro-kernels
- Computer science: software engineering methods, algorithms and complexity analysis, database operations and optimization, cryptography
- IBM/Lotus: Notes (6-8), Domino (6-8), Connections 3, Sametime 8, LotusLive, Traveler 8, LotusScript, C-language API, @-functions, widgets, Domino Designer, XPages, server builds and clustering, replication topology, security audits, email routing, application development
- Internet Standards: XML/DTD/WSDL/SOAP, HTML/CSS, SMTP/MIME/S-MIME
- Tools: Eclipse, NetBeans, Visual Studio, .Net, soapUI, InstallShield

Professional Experience

CHC-3 Consulting Inc, Owner, Oct 1995 to present. This is a consulting practice focused primarily on custom software development for corporate clients.

- Projects have included enterprise-grade legal case management software; low-level system call C programming; security audits for multinational corporations covering user accounts, email routing and database servers; Java coding to merge disparate security directories; extract/transform/load (ETL) between many data formats, including binary and XML; virtual server builds and cluster tuning; generating database output in XML to match a WSDL spec using soapUI; and query optimization comparing indexing schemes.
- Technologies have included Java, C, Eclipse, XML/WSDL/SOAP, S-MIME, IBM/Lotus products, Windows XP/7/Server, Ubuntu desktop and server, InstallShield, soapUI and VMware.
- Clients have included IBM, Bausch & Lomb, Alcoa, Mead Johnson, Hewlett-Packard and the U.S. federal court system.

Boston University, computer science instructor, various semesters 1984 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, Sept 1990 to Oct 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 certificate in computer science, 2009. Courses on computability, algorithms, compilers and software engineering.

Boston University, eight post-master courses, 1988-1990. Topics included cryptography, programming language syntax/semantics, networking/data communication and advanced complexity seminar.

Boston University, M.A. computer science, 1984. 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. linguistic theory, 1979. Senior thesis on formal semantics for natural language.

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

FBI Sentinel Is in Trouble, January 2012. The FBI is trying, once again, to modernize its case file system. The previous effort, Virtual Case File, failed in 2005 at a cost of \$170 million. The latest project, Sentinel, is costing \$450 million and has already slipped more than two years. Sentinel is in trouble.

Why Bad Things Happen to Big Software Projects, January 2012. Examines the special nature of very large, important software projects. Points out that many of these projects are designed for failure from the start, and offers new approaches so the projects are designed for success.

Beautiful Software (book), Amazon, April 2011. 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.

S-MIME Revisited, April 2011. This article updates a popular piece I wrote ten years ago about Lotus Notes and S-MIME. It gives detailed instructions for setting up the Domino Certificate Authority and Verisign digital IDs for secure email between a Notes organization and outside Internet email accounts.

Why Software Really Fails, And What to Do About It. Dr. Dobb's Journal, March 2010. 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.

Hey Programmers! We Got No Theory! Dr. Dobb's Journal, March 2010. There has been considerable interest lately in advancing software engineering from a fad-driven discipline to something founded more firmly on an underlying theory. This article serves as an introduction to the problem and points to the larger initiative at SEMAT.org.

The Missing Theory of Refactoring. Dr. Dobb's Journal, October 2009. Refactoring is a powerful method for improving the design of software, but there have always been some open questions about this helpful method: When should software be refactored? Which refactoring transformation is appropriate in a given situation? Why does refactoring improve design? This article answers these questions by supplying an overall theoretical framework for refactoring. As a good theory should, the framework suggests new refactorings for areas of software design not yet addressed.

Is Software Patentable? IpWatchdog.com, June 2009. I weigh in on the debate about whether software is inherently patentable and argue that *of course* it is.

Software Engineering ≠ Computer Science. Dr. Dobb's Journal, June 2009. Software engineering is different, in a frustrating way, from other disciplines of computer science such as computability and complexity. It is harder to define concepts in software engineering and harder to prove results. This essay proposes an explanation for these problems and shows what kinds of progress *can* be made.

Installing Domino 8.5 on Ubuntu 8.04. April 2009. The title is self-explanatory. Given the low prices of Dell server hardware lately, you can build a complete Domino server for about \$400.

A Software Schedule Ain't Nothin' But a Piece of Paper. Developer.com, March 2008. A somewhat humorous look at software schedules, real and imaginary, with practical tips to avoid software project crises.

Hosting Multiple Domino Servers at One IP Address. April 2006. Detailed instructions for making more than one Domino server visible externally at a single IP address by using TCP port mapping.

Notes to XML and Back Again. July 2005. A detailed code sample about how to export Notes data to XML and how to read XML data into Notes.

HIPAA Computer Security and Domino/Notes, Revisited. January 2005. With the deadline fast approaching for the federal HIPAA healthcare law, I revisited some earlier articles I had written on this topic.

Domino Server Cluster Tuning. October 2004. A technical article about the tuning parameters available to adjust server cluster failover behavior.

A Quagmire in the Tar Pit. Developer.com, May 2002. This article looks at the dilemma faced by software managers given the current proliferation of software development methods, including CMM, ISO-9000, Extreme Programming and Rapid Development. Each requires a large investment of time and effort to learn and implement, so which one should an organization commit to? Is one best for all uses? Are any worthwhile?

It's Not About Lines of Code. Developer.com and Slashdot.org, March 2002. 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.

Are There Limits to Software Estimation? Developer.com and Slashdot.org, January 2002. This article responds to a paper by J.P. Lewis and explores questions about our underlying ability to estimate the development effort for software projects. The article explains results from complexity theory and information theory and shows how these apply (and don't apply) to software estimation.

Why Software Engineering Is Not B.S. Dr Dobb's Journal, April 2001. 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.

Open Source Projects Manage Themselves? Dream On. Lotus Developer Network, September 2000. Challenges one of the central tenets of Eric Raymond's essay "The Cathedral and the Bazaar", which is that open source projects are self-managing. This article was linked by Slashdot, LinuxToday, NewsForge and many other sites.

Approximation Algorithms for NP-Hard Problems. May 1984 (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)

SEMAT: Cleaning Up the Confusion, Jargon and Fads of Software Engineering — An overview of the SEMAT initiative to re-found the field of software engineering on more firm definitions and theory. I presented this talk at the November 2011 meeting of Boston SPIN.

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.

Capability Maturity Model (CMM): An Overview — A summary of this important software development framework. This is a balanced presentation, including a discussion of CMM's weaknesses.