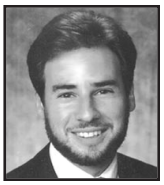

Steve McConnell

After the Gold Rush

EDITOR-IN-CHIEF: Steve McConnell • Construx Software • software@construx.com

In January 1848, James Marshall discovered gold in California's American River, near a mill he was building for John Sutter. At first, he and Sutter dismissed the pea-sized nuggets as a nuisance; they believed the attention that gold would bring would spoil Sutter's plans to build an agricultural empire. But within months, word spread, and by 1849 thousands of men and a handful of women from the United States and around the world ("49ers") headed to California to make their fortunes in what became known as the California Gold Rush. The unprecedented migration west that followed would create something new to America: an economy driven by high-risk entrepreneurialism, fueled by dreams of striking it rich. Precious few 49ers actually realized that dream during the gold rush days, but the dream lives on in many modern software companies and individual software developers—with about the same chances of striking it rich or going bust as miners in California had in 1849.

The California Gold Rush was unique because the gold was found in riverbeds instead of embedded in rock. This meant that, at first, anyone with a tin pan and an entrepreneurial spirit had a chance to make a fortune. But by mid-1849, most of the easy gold had been found; a typical miner spent 10 hours a day in ice cold water, digging, sifting, and washing. As time passed, this backbreaking work yielded less and less gold. Occasional lucky strikes continued well into the 1850s, providing just enough good news to encourage thousands to continue digging.

After the early days, miners had to use more advanced techniques to extract gold. By the early 1850s, a single miner could no longer work his claim alone. He needed help and technology. At first, miners banded together informally to build dams, reroute rivers, and expose the gold. But soon even more capital-intensive techniques were required, and the informal groups of miners were replaced by corporations. By the mid 1850s, most of the miners who remained were corporate employees rather than individual entrepreneurs.

Software Gold Rushes

The advent of a major new technology often means the beginning of what I think of as a "software gold rush." I've personally seen gold rushes with the advent of the IBM PC and Microsoft DOS operating system, the migration from DOS to Windows, and the growth of Internet computing.

Gold rush software development is characterized by high-risk, high-reward development practices. Because few companies have established competitive presences in the marketplace, much of the technological gold seems to be just lying on the ground, waiting for anyone with the right mix of innovation and initiative to pick it up. Software 49ers rush into the new technology, hoping to strike it rich. The stereotypical technology goldrushers are two guys working in a garage who ultimately make a fortune—dynamic duos such as Bill Gates and Paul Allen (Microsoft), Steve Jobs and Steve Wozniak (Apple Computer), Bob Frankston and Dan Bricklin (VisiCalc), and so on.

Gold rush software development is a high-risk activity. The practices employed during a software gold rush are usually associated with hacking rather than engineering: small team sizes, informal processes, long hours, little documentation, barebones quality assurance practices—practices I refer to collectively as "commitment-based development" (*Rapid Development*, Microsoft Press, 1996). Using these practices puts all but the smallest software projects at high risk of failure.

The odds of striking it rich during a software gold rush are about as good as they were during the California gold rush—for every success story, there are hundreds or even thousands of projects that go bust. But the failures aren't nearly as interesting as the successes, and so we don't hear very much about them. As with the California gold rush, projects run with commitment-based development are successful just often enough, and are so enormously lucrative when they do succeed that they convince soft-

LEADERSHIP CHANGES

Four members of our Editorial Board—Tom DeMarco, Ophir Frieder, Sadahiro Isoda, and Tsuyoshi Nakajima—and contributing editor Shari Lawrence Pfleeger are retiring this year. We also have three retiring members of the Industrial Advisory Board, Stephen Andriole, Ken Dymond, and Takaya Ishida. All our editors are vital to our goal of maintaining high technical quality in useful, practical information. I thank each of these volunteers for their contributions and support throughout their tenures.

I am happy to announce that Lawrence D. Graham, Jeffrey M. Voas, and Karl E. Wiegers have joined our Editorial Board. Please join me in welcoming them to *IEEE Software*.

—Steve McConnell

Lawrence D. Graham is an associate at the law firm Christensen, O'Connor, Johnson & Kindness. He practices patent, copyright, trademark, and other intellectual-property law. He has also served as an adjunct professor at the Pepperdine University Graduate School of Business, has lectured at the Seattle University and University of Washington law schools, and is presently an adjunct professor at the Seattle University School of Law.

Graham received a BS in electrical engineering from the US Air Force Academy, an MBA from Pepperdine University, and a JD from the University of Washington. His book, *Legal Battles that Shaped the Computer Industry*, will be published by Greenwood Press in 1999. He can be reached at graham@cojk.com.

Jeffrey M. Voas is the cofounder of and chief scientist at Reliable Software Technologies Corp. and acting director of Software Assurance Research, Software Testing Assurance Corp. He is also an adjunct professor of computer science at West Virginia University, on the board of governors of the Center for National Software Studies, and chairman of the IEEE Computer Society Task Force in Software Assurance. He has written over 100 articles and two books, *Software Assessment: Reliability, Safety, Testability* (with Michael Friedman) and *Software Fault Injection: Inoculating Programs against Errors* (with Gary McGraw).

Voas received an MS and PhD in computer science from the College of William and Mary. He can be reached at jmvoas@rstcorp.com.

Karl E. Wiegers is a principle consultant at Process Impact, a Rochester-based company that provides software process consulting and education services. For 18 years, he worked for the Eastman Kodak Company in software process, software quality, and project management. Karl has led process improvement activities in small application development groups, Kodak's Internet software architecture group, and a division of 500 software engineers developing embedded and host-based digital-imaging software products. He has written over 70 articles and his book, *Creating a Software Engineering Culture* (Dorset House, 1996), won the Jolt Productivity Award from *Software Development* magazine.

Wiegers received a BS from Boise State College and an MS and PhD, both from the University of Illinois at Urbana-Champaign. He is a member of the IEEE, IEEE Computer Society, ACM, and ASQ. He can be reached at kwiegers@acm.org.

ware developers that such high-risk practices can work, thus keeping the entrepreneurial dream alive.

AFTER THE GOLD RUSH

Post–gold rush software development is characterized by more methodical, lower-risk, capital-intensive development practices. Projects use relatively large teams, rely on more formal processes, adhere to more standards (compatibility with legacy code, industry-wide protocols, and so on), and work with much larger code bases. The emphasis is less on rushing software to market quickly and more on reliability, interoperability, usability, and other "esoteric" product characteristics that hardly matter during a gold rush but matter a lot when a technology matures.

Companies most successful during one gold rush are likely to fail during the next. The archetypal post–gold rushers are the companies that became established during an earlier gold rush. These companies repeat Marshall and Sutter's mistake of seeing new-technology gold as a nuisance that will interfere with their well-laid plans for extracting maximum value from the claims they staked during the last gold rush. Examples of companies that were slow to pick up new-technology gold nuggets include IBM during the early days of PC-DOS; Lotus during the early days of Windows; and Microsoft during the early days of the Internet. We'll undoubtedly see this pattern repeated during the late days of the Internet by some of the companies that had the greatest successes in the early days—Netscape, Yahoo, Amazon.com—only time will tell which will successfully make the next great transition.

Gold rush–style development practices have even lower odds of working in a post–gold rush phase. In the early days of a new technology, there are few established players or products. The technological barriers to entry are low, and early products can be small and still succeed. The first version of MS Word for Windows consisted of just 249,000 lines of code. As with the California gold rush, fewer people and less capital are needed to stake a claim during the early days of a new technology. Two guys in a garage have a chance to compete against the major corporations when a successful product can be built with 249,000 lines of code. As the technology matures, however, the easy gold runs out, and successful companies have to compete on the basis of more capital-intensive projects. The current version of Word , for example, consists of more than 5 million lines of code. One of the most damaging mistakes that successful

gold rush companies make is to persist in using gold rush development approaches as the technology matures and their projects scale up. To compete successfully in the post–gold rush phase, successful projects need to do a lot more than simply multiply the number of guys and get a bigger garage.

Gold rush economics are sometimes more sensible than they appear. It may be as hard for an established company to compete in a gold rush phase as it is for two guys in a garage to compete post–gold rush. During a gold rush, having thousands of individual software developers take on entrepreneurial risk voluntarily—with one in a thousand striking it rich and the rest chalking their losses up to experience—is tremendously beneficial from a macroeconomic point of view. No one but the individual entrepreneurs pays for the failures, and everyone has a chance to benefit by buying and using the products that succeed. But how can an individual company harness this dynamic? What company could possibly afford to fund thousands of individual entrepreneurs during a gold rush phase just to find the one or two that successfully develop new gold rush technology? *No* company can—which is one reason that software company acquisitions during a gold rush phase are more sensible than they might at first appear. Some industry observers thought Microsoft was crazy to pay $130 million to acquire Vermeer Technology, original creators of FrontPage, when it had only about $10 million in annual revenue. But from the entrepreneurial gold rush point of view, paying $130 million for the one success in a thousand is a cheap alternative to funding thousands of entrepreneurial experiments internally, nearly all of which would ultimately be dead ends.

California or Bust, Software Engineering Style

Gold rush software projects might be inherently risky, but the use of haphazard software development practices has made them riskier than they need to be. Developers working on gold rush projects have been saddled for decades with the methodological equivalents of tin pans and shovels. Historically, software engineering has focused its attention almost exclusively on post–gold rush projects. This needs to change. Gold rush software projects are essential to the forward march of technology and vital to the economy. More important, these are the projects that have the power to capture the imaginations of leading software practitioners. Modern software engineering needs to rise to the challenge of discovering and refining practices that can make more of these projects successful. ❖

IEEE SOFTWARE

Editorial Board

Ted Biggerstaff (Microsoft), Maarten Boasson (Hollandse Signaalapparaten), Terry Bollinger (MITRE), Andy Bytheway (Univ. of the Western Cape), David Card (Software Productivity Consortium), Carl Chang (Univ. of Ill., Chicago), Larry Constantine (Constantine & Lockwood), Christof Ebert (Alcatel Telecom), Robert Glass (Computing Trends), Lawrence D. Graham (Christensen, O'Connor, Johnson, & Kindness), Natalia Juristo (Universidad Politécnica de Madrid), Barbara Kitchenham (Univ. of Keele), Tomoo Matsubara (Matsubara Consulting), Nancy Mead (Software Eng. Inst.), Stephen Mellor (Project Technology), Pradip Srimani (Colorado State Univ.), Wolfgang Strigel (Software Productivity Centre), Jeffrey M. Voas (Reliable Software Technologies Corporation), Karl E. Wiegers (Process Impact)

Industry Advisory Board

Robert Cochran (Catalyst Software), Annie Kuntzmann-Combelles (Objectif Technologie), Alan Davis (Omni-Vista), Enrique Draier (Netsystem SA), William Griffin (GTE Labs), Arthur Hersh (Hersh Group), Eric Horvitz (Microsoft), Dehua Ju (ASTI Shanghai), Donna Kasperson (Science Applications Int'l), Günter Koch (Austrian Research Centers), Wojtek Kozaczynski (Rational Software Corp.), Karen Mackey (Lockheed Martin), Masao Matsumoto (Univ. of Tsukuba), Susan Mickel (Rational Univ.), Deependra Moitra (Lucent Technologies, India), Melissa Murphy (Sandia), Kiyoh Nakamura (Fujitsu), Grant Rule (Guild of Independent Function Point Analysts), Chandra Shekaran (Microsoft), Martyn Thomas (Praxis), Sadakazu Watanabe (Fukui Univ.)

Contributing Editors

Ware Myers, Roger Pressman, Ellen Ullman, Mike Yacci

Magazine Operations Committee

Carl Chang (chair), William Everett (vice chair), James Aylor, Jean Bacon, Wushow Chou, George Cybenko, William Grosky, Steve McConnell, Daniel E. O'Leary, Ken Sakamura, Munindar P. Singh, James J. Thomas, Yervant Zorian

Publications Board

Benjamin Wah (chair), Jon Butler, Carl Chang, Alan Clements, Dante Del Corso, Richard Eckhouse, William Everett, Francis Lau, Dave Pessel, Sorel Reisman

Editor-in-Chief: **Steve McConnell**
10662 Los Vaqueros Circle
Los Alamitos, CA 90720-1314
software@construx.com

Editors-in-Chief Emeritus:
Carl Chang and Alan M. Davis

Managing Editor: **Dale C. Strok**
dstrok@computer.org
Staff Editor: **Anne C. Lear**
Assistant Editor: **Cheryl Baltes**
Magazine Assistant: **Robin Martin**
rmartin@computer.org

Art Director: **Jill Boyer**
Cover Illustration: **Dirk Hagner**
Technical Illustrator: **Alex Torres**
Production Artist: **Jill Boyer**

Publisher: **Matt Loeb**
Membership/Circulation
Marketing Manager: **Georgann Carter**
Advertising Manager: **Patricia Garvey**
Advertising Coordinator:
Marian Anderson

Editorial: Send 2 electronic versions (1 word-processed and 1 postscript) plus 2 hard copies of articles to Managing Editor, *IEEE Software*, 10662 Los Vaqueros Cir., PO Box 3014, Los Alamitos, CA 90720-1314; software@computer.org. Articles must be original and not exceed 5,400 words including figures and tables, which count for 200 words each. All submissions are subject to editing for clarity, style, and space. Unless otherwise stated, bylined articles and departments, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in *IEEE Software* does not necessarily constitute endorsement by the IEEE or the IEEE Computer Society.

Circulation: *IEEE Software* (ISSN 0740-7459) is published bimonthly by the IEEE Computer Society. IEEE headquarters: 345 E. 47th St., New York, NY 10017-2394. IEEE Computer Society Publications Office: 10662 Los Vaqueros Cir., PO Box 3014, Los Alamitos, CA 90720-1314; (714) 821-8380; fax (714) 821-4010. IEEE Computer Society headquarters: 1730 Massachusetts Ave. NW, Washington, DC 20036-1903. Annual electronic/paper/combo subscription rates for 1999: $27/34/44 in addition to any IEEE Computer Society dues, $49 in addition to any IEEE dues; $93 for members of other technical organizations. Nonmember subscription rates available on request. Back issues: $10 for members, $20 for nonmembers. This magazine is available on microfiche.

Postmaster: Send undelivered copies and address changes to Circulation Dept., *IEEE Software*, PO Box 3014, Los Alamitos, CA 90720-1314. Periodicals Postage Paid at New York, NY, and at additional mailing offices. Canadian GST #125634188. Canada Post Publications Mail Product (Canadian Distribution) Sales Agreement Number 0487805. Printed in the USA.