Presents
An IT Metrics and Productivity Journal Special Edition

Focus on Steve McConnell, Author of *Software Estimation: Demystifying the Black Art*

A CAI State of the Practice Interview
August, 2006

Biography of Steve McConnell

Steve McConnell is CEO and chief software engineer at Construx Software, where he oversees software engineering practices, teaches classes, and writes books and articles. Steve is the author of the computing industry classics *Code Complete* and *Rapid Development*, both winners of *Software Development* magazine's Jolt award for outstanding software development books. He is also the author of *Software Project Survival Guide* and numerous technical articles. Steve was editor-in-chief of *IEEE Software* magazine from 1998 to 2002. His newest book is *Software Estimation: Demystifying the Black Art.* Our interview between Steve McConnell and Michael Milutis, Executive Director of the IT Metrics and Productivity Institute, took place in April of 2006.

◆ ◆ ◆ ◆ ◆ ◆ ◆ ◆ ◆

CAI: Could you tell us a little bit about yourself, the path your career has taken and what you are working on today?

STEVE MCCONNELL: I've been working in the software industry for 22 years now, and like a lot of other people I started out as a programmer. I initially majored in philosophy believing that I would end up working towards a master's degree in that field, but I was also minoring in computer science.

As I entered my senior year, I started filling out grant and fellowship applications. The questionnaires I had to answer helped me clarify for myself that I really enjoyed

computer science. Consequently, I decided to defer my entrance into graduate school and try my hand at programming for a while. I worked through the summer and into the fall and was enjoying my job very much. One autumn morning I woke up in cold sweat. I realized that if I had accepted any of those fellowship offers, I would be sitting in graduate school – studying philosophy! This did not sound like a lot of fun.

In short, I got into software through the back door, which is probably the way half of the people in this industry have gotten into the field.

After roughly 10 years, I published my first book – *Code Complete*. Shortly thereafter, I went back to hands-on programming. After a few more years, I published my second book – *Rapid Development*. After publishing *Rapid Development*, I started my own company- Construx Software. We just celebrated our 10th anniversary.

The last few years have been a mixture of writing, working on my company, and developing training and consulting offerings for our clients. I find that I like the variety. I like the ebb and flow of focusing on different aspects of the business from time to time.

CAI: What inspired you to write your first book, *Code Complete*?

STEVE MCCONNELL: Kids don't really fantasize about growing up and writing a book on software development. However, like many people with a liberal arts education, I did hope someday to write something. I took some advice from Mark Twain: "If you want to write, write about something you know about." Well, software was something I knew about.

At first, my plan was to publish an article about software construction practices. However, as I started the background research, I simply could not find any decent books that described software construction practices in depth. I eventually discovered that such a book did not exist.

It was my subsequent research for this article that formed the backbone for *Code Complete*. I started collecting notes in 1989 and the book came out in 1993. The entire book grew organically out of that original desire to write a single article.

CAI: How is it possible that as recently as 1989 there was not a single authoritative text on software construction practices?

STEVE MCCONNELL: There were several, fragmented offerings but most of these were quite specific to various programming languages. For instance, Brian Kernighan and P. J. Plauger had a nice book – *The Elements of Programming Style* - but it used a slightly arcane language called Rational Fortran (RATFOR). Moreover, it had not been updated in quite a few years. There was also an interesting series by Henry Ledgard – *Programming Proverbs* – but it was not up to date either. The examples Ledgard gave were not even in current programming languages. Neither of these books could function as a comprehensive and authoritative reference tool for software construction.

That was really my goal; to write an authoritative reference book on software construction.

CAI: How has the quality and quantity of software development literature improved since you first wrote *Code Complete*?

STEVE MCCONNELL: The literature has improved dramatically. Back in 1993, when I published *Code Complete*, I think it would have been possible for a motivated professional to read every significant book in print that was related to the field of software engineering. In 2006, however, it would probably be impossible for a motivated individual to read every significant book *within a single specialty area*. There has been an explosion in good software development literature over the past 15 years.

CAI: You recently came out with a new book - *Software Estimation: Demystifying the Black Art*. What did you set out to achieve with this?

STEVE MCCONNELL: I wrote my first book - *Code Complete* – mostly because the topic had not been written about. I noticed a gaping void and I believed that I was capable of filling that void. The motivation for writing *Code Complete* was not my burning desire to write about software construction.

In contrast, I have had a passion about software estimation since the very first years of my career. It is the topic that originally spawned my interest in developing more systematic approaches to software construction.

Why are some people better or faster programmers? Why are some people creating higher quality code? What made it easy for me to estimate my individual work, but more difficult to estimate the entire project? These were the kinds of questions that had originally captured my interest.

So I had been thinking about writing a book on software estimation for quite a while. I was also motivated by the need for a more practitioner-oriented book. I felt that this was missing. At the same time, I wanted to deliver some fairly simple but effective techniques within the context of a short book. And I wanted it to be accessible to as many software professionals as possible. In short, *Software Estimation: Demystifying the Black Art* was designed to target software practitioners who wanted to improve their estimates but who were not necessarily interested in becoming estimation experts.

CAI: Could you explain why you chose to describe software estimation as a "black art?"

STEVE MCCONNELL: It was deliberately chosen. And I mean it literally. In my experience, most people have very little exposure to systematic estimation practices. They honestly cannot understand why some estimates are accurate and some are not. Consequently, one of my objectives in writing this book was to reveal the mechanisms behind estimation and to show that it is not all that mysterious. Certainly, estimation is not simple. Nor is it straightforward. Nevertheless, with an understanding of just a handful of techniques and concepts, estimating accurately and consistently becomes dramatically easier.

CAI: In your book, you referenced a Standish Group study which concluded that 70% of all software projects were coming in over time, over budget or not at all. What factors might be contributing to these failures?

STEVE MCCONNELL: One factor, certainly, is poor estimation. Many people out there are using an estimation process that is either inadequate or characterized by weak execution.

A second factor might be poor project performance. In other words, the estimates might have been on the money, but the project managers might not have implemented the project very well.

A third possible factor is the common, erroneous usage of the word "estimate." I believe that when most people use the word "estimate," they are actually referring to a "business target." A project leader, for instance, might stress the importance of being ready in time for a trade show, or being ready by a regulatory date, or having the software on the shelves in time for the holiday season. While those might be important business goals, they do not - strictly speaking - constitute estimates. An estimate is an analytical assessment of the probable outcome of a project, independent of the desirability of that outcome.

It is unclear to me how many of these projects surveyed by the Standish Group were actually estimated in any kind of analytical manner. It is only speculation on my part, but I would guess that a much smaller percentage overran their estimates, whereas a much larger number might have only overrun their business targets.

CAI: What are the most important guidelines to keep in mind when formulating estimates?

STEVE MCCONNELL: The first piece of advice is to differentiate between estimates, targets, and commitments. These are three distinct ideas. Merging these ideas together will degrade the quality and accuracy of your estimates while reducing your

ability to define clear business targets and fulfill commitments. In short, when you are estimating, make sure you are actually estimating.

If possible, you should also base your estimates on historical data for similar projects conducted within the same organization. Using data from past projects within the same organization allows your estimates to account for a wide variety of organizational influences.

A final piece of advice to keep in mind is that there are times when meaningful estimates may not even be possible. In my most recent book, I talk about the "cone of uncertainty." What the cone of uncertainty demonstrates is that, during the earliest stages of a project, estimates are susceptible to a high degree of variability. This is because there are many details – some of which may be very significant – that have not yet been pinned down. Until some of these details are solidified, it is theoretically impossible to obtain accurate estimates. One must come to terms with this intrinsic degree of uncertainty.

CAI: There are many different estimation techniques to choose from. Could you offer some advice on how organizations can best determine which techniques are appropriate for them?

STEVE MCCONNELL: Rather than focus on specific techniques, I would simply recommend the formulation of estimates using a systematic approach.

There are many good ways to create estimates, but the best way to estimate will always be a decision that is contingent upon context. Some techniques will work well

in certain contexts, others will not. Moreover, the specific estimating method you choose will vary considerably depending on the size of a project. Your choice of a method will also vary if your project is iterative as opposed to sequential.

CAI: Where can people turn for guidance when they need to make decisions regarding specific techniques for a given scenario or context?

STEVE MCCONNELL: The actual meat of my book – about 300 pages worth – is devoted to descriptions of various estimating techniques, complete with scenarios in which each of these techniques apply. In parts two and three of my book, I have included detailed reference tables – in the front of each chapter - that demonstrate the applicability of the techniques being discussed. They include information on what size project the technique would be useful for, which development stage the technique applies to, whether the technique would be more applicable to iterative or sequential projects, and what level of accuracy is possible with the technique.

CAI: Could you address the some of the most common difficulties encountered with estimation? How might these difficulties vary depending on the size of an organization?

STEVE MCCONNELL: In my experience, the difficulties do not seem to vary much between large and small organizations.

The biggest and most common difficulty encountered is the confusion of estimates with targets or commitments – a topic that I discussed earlier. It applies to the smallest

organizations, as well as the largest organizations.

A second difficulty involves the cone of uncertainty. At what point is it too early to estimate? At what point does it make logical sense to conduct an estimate? People frequently give themselves a false sense of accuracy when they conduct estimates in the wide section of the cone. For example, if you are making estimates before pinning down all of your detailed requirements, you might wind up attaching a 25% margin of error to your estimate when you should really be factoring in a 200% or 400% margin of error. Most organizations don't understand the level of variability that can exist in estimates when they are formulated during the early stages of a project.

The third difficulty encountered is the off-the-cuff estimate. There is naturally a difference between an estimate given while in line at lunch, versus an estimate given after sitting down for 15 minutes with a spreadsheet and looking through historical data. Organizations that accept off-the-cuff estimates make themselves susceptible to large scale errors. If someone is asked for an off-the-cuff estimate they should respond, "Why don't you give me some time to go back to my desk. I want to check some notes about past projects. I can send you an e-mail or call you in about 15 minutes." The petitioner may still be receiving a back of the envelope estimate, but it will at least be based on documented facts rather than personal memory. Those 15 minutes may not give you an estimate with a 5% margin of error; however, they will almost certainly help you avoid errors in the range of 200-400% or greater.

CAI: Estimation is interrelated with process and measurement. In light of this, what must organizations first have in place on the process and metrics side, before they can expect any significant return on their estimation efforts?

Computer Aid, Inc.
1390 Ridgeview Drive
Allentown, PA 18104
(610) 530-5000

STEVE MCCONNELL: I doubt organizations need to have that much in place before at least seeing minimal returns on their estimation efforts. On the other hand, the more they have in place in terms of process and measurement, the better their ability will be to estimate accurately.

Regarding process specifically, an organization that has chaotic development practices can employ certain techniques that will improve their estimates. Nevertheless, there will still be inherent limits to the accuracy they can achieve due to the chaos in their processes.

Metrics are slightly different. Good data from past projects can serve as a basis for better estimation of future projects. This goes back to the discussion we had earlier about historical data. An interesting thing to note, though, is the general correlation between measurement and process. If there is a lot of chaos on the process side, for instance, then there is probably going to be a lot of noise in the data. As the processes stabilize, there will be less noise. The data will be more accurate, and the estimates derived from them will be more meaningful and more accurate as well.

CAI: What role - if any - do tools play in the integration of process, measurement, and estimation?

STEVE MCCONNELL: As you get into some of the more powerful approaches to estimation – particularly those based on historical data - it becomes extremely useful to have the support of dedicated software estimation tools. This is what I call "the science of estimation." Quantitative and statistically-oriented estimation becomes computationally intensive rather quickly. It also becomes completely impractical to use a hand calculator for such math-intensive techniques. At this point, tool support becomes very important.

My company actually offers an estimation tool called Construx Estimate that can be downloaded for free from www.construx.com/estimate. It is a superb tool – especially considering that it is free of charge.

However, there are limits to what you can get in this world for free. If you need a tool that goes beyond Construx Estimate's capability, there are plenty of other tools out there on the market, but they will need to be purchased.

CAI: Your book features a chapter on politics, negotiation and problem solving. What did you want IT executives and software practitioners to learn from this?

STEVE MCCONNELL: Typically, negotiations refer to discussions between parties with competing interests. There is a pie of fixed size and the point of the negotiation is to divide up the pie. In some cases, the negotiations are aimed at win-win solutions which try to make the pie as big as possible before it is divided. However, at the end of the day, the pie is still being divided with each party taking their own piece.

Over the last several years, I have had a difficult time viewing estimation discussions as negotiations. I simply cannot discern which pie is being divided. Estimation discussions typically occur between business executives and technical leaders. Interests do not compete here; indeed, they are one in the same. Everyone is jointly benefiting from the same pie. Either the project does well and supports a successful company or the project does poorly and undermines the goals of the company.

Therefore, the word "negotiation" turns out to be a misleading expression. I believe

the term "problem solving" is much more apt for describing these kinds of deliberations. Both business executives and technical leaders are trying to define the project in a way that will best support their own joint interests. The technical role is to bring hands-on information to the table, e.g. the scope of the work, the nature of the commitment, whether there is high or low risk, etc. The business role is to define the achievements: the importance of the timing, the prioritization of the specific features, how sales might be affected by reduced features, etc. By adopting a problem solving approach, all players can find their way to mutually beneficial commitments that accurately reflect both the integrity of the technical estimates and the business targets set by management.

CAI: Why is this distinction between "negotiations" and "problem solving" important enough to include in your book?

STEVE MCCONNELL: After the formulation, through a solid analytical process, of accurate estimates, those estimates are then brought to the business leaders. If everyone views these discussions as a kind of negotiation, then the business leader may cut down the estimates by 40%. On the other hand, if the technical team can shape these discussions into more of a problem solving exercise, then the estimates presented will likely be taken seriously and the resulting commitments agreed to will be more realistic and achievable.

Each department must respect the other department's expertise, however. This is critical. It is entirely appropriate for the business department to maintain ownership over the business goals of the company. Likewise, the business people are not in any position to make analytical assessments of technical information and need to respect the estimates provided by the technical leaders. Thus, the technical people should own the estimates, the business people should own the targets, and in this manner