Software Estimating, People, and the Smell of Popcorn:

An Interview with Dr. Randall Jensen

CrossTalk: What are the foundational benefits of software cost estimating?

Randy: Without a software estimate—a cost and schedule estimate—you can't manage the program. Fundamentally, that's what the problem is. People estimate off the top of their heads very optimistically and, from the moment the project starts, it's behind—never does get caught up, gets a bad reputation, and lots of times fails, simply because they had no idea going in what the estimate should have been.

CrossTalk: Are we attacking the "same old problem?"

Randy: I think the problems are multi-faceted—it's a horrible word to use—but we have two things occurring. One is within the military and DoD projects: People move into the estimating profession, they're there for three years, then they go onto another profession because there is no future and really no promotions possible as an estimator. Then there are those who only develop an estimate once every four or five years, so they really never develop the skill. What we get are a lot of people who pick up an estimating tool, look at the instruction manual for 15 minutes, punch some numbers into the computer, and the computer tells them something that they interpret as truth—that is the estimate that people typically use. Real professional estimators, the people who master and use the skill for a long period of time, are very good and worth seeking out or utilizing.

CrossTalk: With that said, is there still a software crisis, and would this be the crux of the problem as you see it?

Randy: Yes. The software crisis included problems like we couldn't manage the costs, we couldn't estimate the schedule, we couldn't maintain the software, and we couldn't modify the software. There were eight characteristics of software that caused the term "software engineering" to even happen. And the experts thought by changing the name that would make the problems go away, but the managers didn't go away, and the problems didn't go away either. We still can't maintain the software, the software still has errors, and the software is still not delivered on time. We've tried.

The first really big effort I saw to get the problems under control was in the late '80s with the CMM®. We thought if we got the processes right, then the productivity would be better, and we would make fewer errors. We tried to fix the problems with structured programming, and then structured design, and then structured analysis.[1] But we've heard the same mantra over and over: *Each of these things will cause errors to disappear and productivity to improve by an order of magnitude*. What we have now is an organized process doing things in an orderly way—and producing the same stuff we did before. I always refer back to a phrase that said, "When processes are optimized, people are interchangeable." If we get the process right, it doesn't really matter who the people are. But it's the people who are the problem. It's also the people who are the answer.

CrossTalk: What I know of tools and your teachings is that you factor in that people issue quite heavily. What is the people issue, as you see it?

Randy: I put it into two categories. The first category comes down to their ability to communicate with each other. If we go back to the old Skunk Works® that Lockheed built up millions of years ago, they had a tremendously high productivity of turning out good products in a relatively short period of time, and it worked. The scheme was to give everybody on the project access to everybody else. If you had a question you could go ask somebody a question. Oddly enough, the first cubicle was introduced by the Skunk Works, only it was a mobile workstation that they could move around so they could be where they needed to be when they needed to solve problems.

Someone else found they could maximize the number of programmers per outlet if they put all the cubicles side-by-side. And we have little walls and places where people could work without interfering with anybody else—they could work by themselves. This was exactly the wrong thing to do, I think, from that point of view.

Managers in general look at programming the same way their mentors taught them: You sit down, you code, you work by yourself, and you are in a very productive environment—which is exactly false. We learn in kindergarten, grade school, and high school that if you work with somebody else you are copying and cheating. So we are taught as children to work alone. We don't learn about working with somebody else until we get older, and, by then, habits are developed and people think they need to work alone—and that's not really true. Managers come up and follow the same idea: If people are talking to someone else, they are wasting time. If people go to the coffee machine, they are wasting time. They should be working, and that's what we're paid for. So if we work on a problem and can't solve it, even though it may take three or four weeks, we *will* finally ask someone for help. But that interaction should be going on *all the time*.

One of the keys of the agile movement—not that all the keys of the agile movement are good ones—is that people are more important than process. If managers recognize the need for their teams to work together, they will. Software Skunk Works are not entirely unique. They've been tried and tested, and they pay off very well. The management response to those organizations is typically, "That's not the way we do business."

Software is not accounting. You're not looking at a column of numbers and trying to make them balance. It's not like almost any other activity I can think of. It's very creative. Every piece of software you write is new and unique, and sharing that development is one way to really improve both productivity and experience of the people who are involved in the communication. It makes the whole organization better.

CrossTalk: I've heard you state that it's good that everyone understands software cost estimation, but who specifically on a software project team really needs to know the ins and outs of cost estimation?

Randy: The estimator, who should be an expert with the tool. There are half a dozen tools that estimate projects in an equal number of ways. They approach problems differently, but those estimators who are good at using those tools will get nearly the same answers using different approaches if they understand what they are doing. You don't hand an accountant a scalpel. There has to be somebody in the organization who really knows how to estimate.

On the other hand, it won't hurt the manager to take an estimating class and understand the meanings of the parameters. By understanding the meanings, they can understand quantitatively what the impact of their decisions will be. If I force people into cubicles, my productivity is going to go down about 20%. If you count the analyst as well that's 40% that we've lost. If you give the team tools that don't work, that's another 10 to 12%. If you convert your organization into a well-functioning Theory-Y[2] Skunk Works, you might double or triple productivity. You can look at the individual things that can contribute to cost, and you say, "Yeah, that makes sense, it will work."

I have one good example from my prior life. The project manager took an estimating class. He learned the tool and he did the homework. When he started the next project, he said, "I'm going to do everything the estimating model tells me to do." He located an unused cafeteria as a working Skunk Works. His programming team cleaned up the tables and the floors and he brought in a microwave (he said he bought a lot of popcorn). His task was to keep other people out of the way, which he did. The manager's role was to support the team and keep them moving forward with the development. What he gained was the highest technology rating (I have a numerical way of rating technology)[3] I have ever seen, as well as achieving productivity that had never been seen by that organization. It was about 150% better than the norm.

CrossTalk: DoD is supporting a massive effort to obtain data leading to a new estimating tool. You've looked at data, what are your thoughts on quality of data?

Randy: I've spent almost a year analyzing a total of 960 some-odd data points that had been stored in a DoD software product database. At the end of the year, I found about 15 data points that matched what I would consider reality.

The analysis showed that when you included all the data, the development effort was independent of size. When we filtered the data (down to the final 15 points), we realized the "data base" was actually a data repository, not the database as advertised. Most databases are, in reality, repositories. The two should never be confused. We turned in a report and we haven't heard another word from them. I don't expect we will. I don't think the analysis was what they wanted to hear.

There is a real concern with data quality: We can't seem to get our hands around cost and schedule data. There's the CHAOS report[4] showing that about a third of projects are never completed within a reasonable schedule and cost. And that problem will go on and on.

CrossTalk: Do you think the industry has warmed up to the idea of the people factor? Do you think it is generally accepted?

Randy: No. Again, it's not the way they do business, or should I say, it's outside their organization culture. I've been studying this for about 30 years. If you look at publications on the people issues versus the tool and process issues, you'll see the vast majority focus on technology while only a few look at people issues.

I remember one collaborative project where the error rate was three orders of magnitude less than normal and productivity was much higher than that organization had ever done before. When it was presented to their management staff, one of the project managers said, "If we forced our senior people to work with someone else, our senior people would all quit." Now, I looked at the numbers (sitting in the back of the room), and said, "You know, if all of their senior people quit, the productivity would increase."

Anyway, the company was determined they couldn't do it because they thought that people don't like to work together—and I think that's absolutely wrong. I've never seen that in a team-oriented environment.

CrossTalk: Along those same lines then (these people issues), are we able to accurately estimate and measure human components?

But "socialize" is exactly what they want to do. Even when you're "wasting" time, you're communicating, and in wasted time there are usually work problems that come into it—you end up talking about what you're stuck on.

Randy: Yes we can. It's not a hard science, mind you. It is like physics in some ways. The major estimating models dating back to 1980 have all predicted the effect of the human components.

We have mapped out a group of very strong indicators of what we call a capability rating. Capability ratings are not just I.Q. and programming skill and experience: It includes motivation, management style, and the ability to communicate and to work with others—your personality issues and the ability to talk to somebody. To take one from contemporary lore, Sheldon on *The Big Bang Theory*[5] is a perfect example of one of those people who would be very low on the capability rating: In spite of being terribly intelligent, he can't communicate that intelligence to anybody else. And that's important.

So one of the first questions I ask myself when looking at an organization is, "Are they talking?" The first test that I use when I walk into a work area—and look over the sea of cubicles—is the noise level. What you hear makes all the difference in the world.

One in particular is a perfect example: They had the world's best-looking cubicle environment I had ever seen—all uniform, all had the same equipment, same manufacturer, and everybody had two displays to work with. The organization was set up so that the people who built the previous system, the old-timers, were all on one side of the room; the people building the new system were on the other side of the room. In talking with the older group, they told me that they had all of this experience they were transferring to this younger group. When we walked in the room, I stopped and said, "Just simply listen." We listened for a few minutes, and there was not a sound. I said, "Who's communicating in here … anybody?" The answer was, "Well, they're all doing it online—the Internet, IMs, e-mails, what have you." And walking through, programmers were at their workstations, quietly at their computers entering things. I told them that the *content* of the communication between people—in their case between the experts and the novices—(according to the study I referred to) is only about 7%[6] of the communication. So a lot is lacking without face-to-face discussion. They may as well be reading documents because the interaction they're getting is slow, unclear, and you can't quickly discuss it or argue about it. So, yes, that's an issue.

Another thing we ask is, "Are the cubicles big enough that somebody can come into your cubicle, sit down, and talk about a problem?" In this organization there was not. If they came to ask a question, they leaned over the wall—and there was no one doing that. You could have areas with conference tables and lots of white boards where they could discuss problems—and this organization didn't have any of those either.

One of the last things we looked for—and it turns out to be one of the most significant, I don't know why, it just works that way—is the smell of popcorn. Popcorn indicates that people are talking—you don't sit and eat popcorn by yourself. We had this conversation with one of the younger employees:

"If you get stuck on a problem, who do you go to?"
"Well, I can send an e-mail to so-and-so."
"Is there anyone you can talk to?"
"Well, we're not supposed to talk. We're not supposed to socialize."

But "socialize" is exactly what they *want* to do. Even when you're "wasting" time, you're communicating, and in wasted time there are usually working problems that come into it—you end up talking about what you're stuck on.

CrossTalk: So it's a hard science, but measurable nonetheless.

Randy: It's all communication. The whole thing. And it's very easy.

CrossTalk: It seems, at least through your experience, that many organizations seem to be having a hard time not just grasping the need for interaction but the actual quality of the physical environment in which employees work.

Randy: I went to this company that was having trouble delivering a system. Hypothetically, they were in Bozeman, Montana: cold, windy, and just plain miserable. We went to their engineering building, which was a recovered, retrofitted livery stable. Picture one of those old stables: big double doors where wagons went in and out, a stone building. There were rows of tables in there where they were assembling the components for the systems they were delivering. I asked, "Where are the software people? I don't see them anywhere." I was led through a door in the back into a lean-to on the back of the building. It had a corrugated steel roof and tiny factory windows—some of which were broken out. Inside there were rows of tables with people huddled over their terminals, blazing away at this software: They were all wearing overcoats, typing with gloves, no Internet, with the only heat being generated by their computers. And, as a capstone to this whole thing, the floor was *dirt*. I said, "This is insane, why are you here? I can't believe this." They said, "There's no place within 50 miles where we can find work. This is the place—if you want to program, you work here." They were all there, not delivering anything, freezing to death. I explained the observation to the vice presidents the next day saying, "These people hate this place." They said "Why? We're giving them all this opportunity." They could not understand that environment and motivation had something to do with their lack of success.

CrossTalk: One final question. If I'm a software project manager and I'm reading this interview, what are some of the things I can do today to make me better at cost estimation?

Randy: That's a real tough one. No one has asked me that question before.

One, I would review a cost estimating user manual and see what parameters are used in the estimate—focus in on the parameters that have the largest effect. Spend a whole day just reading the manual and understanding the cost impact of the decisions you're going to make on a project. That would be very worthwhile.

I think Jerry Weinberg said it best—he has a law that says that everybody has a problem, and it's always a people problem.[7] And most projects I've looked at bear that out.

I could go on all day talking about this.

Disclaimer:
® CMM is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.
® Skunk Works is registered in the U.S. Patent and Trademark Office by Lockheed Martin.

NOTES

1. See < http://en.wikipedia.org/wiki/Structured_analysis>.
2. See <http://en.wikipedia.org/wiki/Theory_X_and_theory_Y>.
3. For more on rating technology, see Jensen's co-authored CrossTalk article (with Lawrence H. Putnam Sr., and William Roetzheim) at <http://www.crosstalkonline.org/storage/issue-archives/2006/200602/200602-Jensen.pdf>.
4. See <http://www1.standishgroup.com/newsroom/chaos_2009.php>.
5. A sitcom on CBS. See <http://en.wikipedia.org/wiki/Sheldon_Cooper#Characteristics>.
6. With 38 percent being tone of voice and 55 percent body language, according to Albert Mehrabian's 7-38-55 Rule. See <http://en.wikipedia.org/wiki/Albert_Mehrabian>.
7. From Gerald W. Weinberg's Secrets of Consulting: A Guide to Giving and Getting Advice Successfully, specifically the First and Second Laws of Consulting (pg. 5).