Print Article Close Window

From: www.cio.com

Agile Project Management Lessons Learned From Texas Hold'em

– David Taber, CIO

January 13, 2014

I can hear you sighing — and I'm not surprised. I had always considered Texas Hold'em Poker to be a game dominated by blind luck, bluffing and testosterone poisoning. What could be more different from the cold rationality of IT project management?

Actually play the game, though, and you'll see that skill is definitely involved — and there are clear right and wrong responses to most situations. You'll also notice several weird parallels with software projects:

- At the outset, you have surprisingly little concrete, actionable information. What you *do* have is a lot of hope and supposition; you can make decisions based only on general probabilities.
- Most of the time, making an irreversible commitment early in the project is about as effective as going all-in before the cards are dealt.
- As things evolve, prior decisions are sometimes invalidated by subsequent information. Significant changes in strategy may be required, including cancelling part of the work.
- In certain situations, winning will mean spending more than you originally budgeted.
- The biggest costs often come at the end, even though the chances of a great result haven't gotten that much better.
- Being stubborn really costs you. Alacrity and flexibility pay off over time.

Now, these observations aren't exactly another revelation on par with the Mythical Man-Month, but there are still some interesting corollaries for software project management in the enterprise.

You Only *Think* You Know What It Takes to Win

In software projects, the impulse is to create a detailed requirements document to guide the project team before anything starts. In the extreme case, that requirements document, rather than business value, is the basis for the business case. That can work for software *products*, but with business systems — particularly customer-facing ones or CRM software — producing a thorough requirements document is a fool's errand.

[Commentary: Top 10 Ways to Blow Up an Agile Project]
[More: 5 Ways to Send a Custom Software Project Off the Rails]

Why? Even before the ink is dry, requirements documents just aren't accurate enough. During the requirements writing, one group or another won't be able to put its "A Players" on the job long enough. There will always be something more important to do than this busywork assignment — so the document will have invisible gaps, wild assumptions and incorrect priorities.

That's not all. The "requirements tome" approach is based on two assumptions that, all too often, prove to be false in CRM projects. One it that it's possible to specify exactly what's needed before the project

begins. The other is that a successful project delivers exactly what's specified, but *only* what's specified.

OK, then. What are the odds of the following?

- Your users will state their requirements with such thoroughness and precision that you won't misunderstand them.
- You understand your business processes, data quality and cross-system integration issues so well that there won't be any surprises.
- Users won't change their minds about details and priorities during the course of the project.
- There won't be a reorganization that changes priorities or even nullifies some of the requirements.
- Your product or service offerings won't change in substantial, relevant ways.
- Your channels and competitive environment won't change during the life of the project?
- You're able to foresee all the government regulations, customer contractual stipulations and internal standards you'll need to comply with.

In the immortal words of Clint Eastwood's Dirty Harry, "Are you feeling lucky?"

[Tips: How to Deal With Software Development Schedule Pressure]

These mid-project discoveries often mean that initial requirements must be recast or re-prioritized. In other words, the team will need to place different bets than what was contemplated in the document.

Since the initial requirements tome will have to be rewritten anyway, the agile solution is to do discovery at the outset of each sprint — typically, every two to four weeks — to deeply understand the tactical requirement at the instant the implementation begins. In many agile methodologies, these up-to-the-minute detailed requirements are even called "cards."

See the parallels with Texas Hold'em?

You Only *Think* You Know Your Budget

Classic software budgeting is done almost entirely on the basis of cost models: Requirements > Statement of work > Work breakdown structure > Estimates. Thanks to the effects discussed above, the cost estimates are going to be out of whack. Count on it.

[How-to: Move Beyond Project Estimates and Provide Better Value]
[More: 'No Estimates' in Action: 5 Ways to Rethink Software Projects]

There's an additional twist. In business software projects, some of the biggest cost elements are internal staff time, meetings and other sunk costs. All are often ignored in the budgeting process, with twin nasty consequences.

First, the "invisible, free resources" (often, sleep-deprived employees) are over-allocated, over-consumed and cause big project delays. Second, when things get serious, the explicit, costly resources (often, sleep-deprived consultants) come in and cause big budgetary surprises.

So there's a significant chance you'll underestimate the costs. However, the smart business case doesn't focus as much on costs as on business value. Who cares what you think it will cost as long as the final cost is outweighed by the value achieved?

In the course of the project, one squeaky wheel or another will pull rank on priorities and resource allocations. When push comes to shove, the business will discover all kinds of business value that wasn't properly included in the original business case.

Since the entire business case is at best a model — which may well be overcome by new facts — the agile solution is highly incremental budgeting. Each sprint gets its own time-boxed "not to exceed." As the team gets better, the estimation errors start to cancel out. Essentially, the agile model is to make a series of small bets, not an initial "all in" bet.

Of course your finance department wants a fixed price — but in custom software, that's an illusion based on fallacious assumptions. Just in case there's any doubt, any serious CRM effort is going to involve custom software.

David Taber is the author of the Prentice Hall book, Salesforce.com Secrets of Success, now in its second edition, and is the CEO of SalesLogistix, a certified Salesforce.com consultancy focused on business process improvement through use of CRM systems. SalesLogistix clients are in North America, Europe, Israel and India. Taber has more than 25 years of experience in high tech, including 10 years at the VP level or above.

Follow everything from CIO.com on Twitter @CIOonline, Facebook, Google + and LinkedIn.