CIO

Print Article Close Window

From: www.cio.com

# 5 Ways to Send a Custom Software Project Off the Rails

– David Taber, CIO

**May 21, 2012**

The first two articles of this series discussed the relationship between two problematic good ideas—bid-to-win behaviors from vendors and Big Bang release planning from clients. In many a data migration and custom software project, these ideas lead to big overruns in cost and schedule.

This all happens in big cloud projects as well. There are many contributing factors to these bad outcomes—chief among them adversarial incentives, inappropriate metrics and lack of collaborative infrastructure—but those aren't the root cause.

It doesn't matter how well you collaborate and execute *if you are working on the wrong thing*.

## Custom Software Projects Beget Unnecessary Code

When The Standish Group conducted an analysis of big Y2K projects, it evaluated the old COBOL systems that were being converted rather than the success of those conversion projects. The bigger the original project had been, the firm found, the more of its code actually had no current function.

Some of this was correlated with age, of course, as those behemoth systems had evolved over more than 20 years. But that merely underscores the point that larger projects were more likely to be wasteful projects. Imagine maintaining a billing program with 20 million lines of source code when you're not exactly sure what half of it does.

In the group of mainframe projects analyzed, more than 90 percent of the function points in the

original programs didn't need to be recreated at
all in the replacement system. The key issue here is that, in those programs' pasts, every one of those function points was fought over, budgeted for and sweated over— all for a business purpose that was at best temporary and at worst imaginary.

Borrowing a metaphor from the construction industry again, we have to constantly apply value engineering to avoid overinvestment in custom software. At every turn in the project, we must ask, "Isn't there a simpler way to solve this problem?" This applies the lessons of *The Innovator's Dilemma* at the microscopic level by relentlessly looking for the "good enough" alternative to something sexy and elegant.

## Remember, You Can't Fight City Hall

In an organization of any size, political decision rules the day when there's any amount of stress and controversy. The way to improve the quality of decisions is to make sure that your project doesn't get into trouble in the first place.

The following five "great ideas" can send any custom software project off the rails.

**A tight specification makes for a good project.** Too often, this means an overly-detailed technical tome that is thin on business context and silent on economic goals. Unless you are lucky, you've specified something that is so hard to build that it'll cost more than it's worth—though you probably won't be able to see that until halfway through the project. The antidote here is a change of mind-set. Focus the business analysts on describing the business goals, and leave the software implementation details to the team.

**A tight specification makes for a good project, revisited.** Specifications tend to be written at the beginning of a project, during the analysis and scoping phase. As soon as the real project work starts, spec writing typically stops. However, the ramifications of requirements are often not understood until the spec writing is complete. Even worse, business requirements change—often dramatically—during the life of a project. Specs should be an evolving set of short documents, preferably living in a collaboration system with automatic change bars.

**Try to provide a functional equivalent for the existing system.** Since true greenfield projects are rare, most of the time the project is replacing, upgrading or rationalizing an existing system. The problem is, what already exists is not ideal—that's why you're replacing it, after all—and it's part of an ecosystem of business processes that is rarely well understood. Consequently, when you start to deploy that replacement system, mysterious side effects crop up. The antidote here is to conceive of the project as an upgrade to the business *process*, of which the business *system* is just a component. In the immortal words of W. Edwards Deming, "If you can't describe what you are doing as a business process, you don't know what you are doing."

**Use the business case and the budget as your guide.** Too often, the business case provides only an illusion that you know what the project is really worth to the business. The remedy here is deep and continuous involvement, with an active decision loop that is constantly reviewing the business case as it appears *now*, at every iteration of the project.

**Always say "Yes" to users.** As I've written endlessly, user-centered design isn't just for the UI; it's for the whole project. Agile works because users are intimately involved at every stage. But that's users as in individual contributors, not the vice presidents of the business. Of course, VPs represent the business interests—but they live in an information bubble that distorts the nuance and ignores the details where the devil lives. Certainly, it is painful to say "No" to politically powerful constituents. But "Yes" is the weakest word in the English language. Say it too often and you get silly architectures, overblown requirements, idiotic resource allocation and project failures. Nothing in the cloud or agile can counteract this issue.

## The Path to Custom Software Project Victory

Two simple ideas sit at the heart of the thing that wouldn't die. Both are dead wrong.

- Software projects can be managed as if the business environment isn't changing. This means you're building a fossil.

- Businesses can foresee the consequences of software systems before they're built. This means you're building a monster.

In a future article, I'll explain how to say "No" to users in a graceful and effective way.

*David Taber is the author of the new Prentice Hall book, "Salesforce.com Secrets of Success" 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, and David has over 25 years of experience in high tech, including 10 years at the VP level or above.*

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

© 2012 CXO Media Inc.