ARTICLE

Why Is Agile Development So Scary?

By Preston Smith
Posted April 30, 2005 in Business Technology & Digital Transformation Strategies, Business Technology & Digital Transformation Strategies

Agile methods are not appropriate for certain development projects that have special requirements. But in many cases, organizations resist using an agile approach on any project. Why?

I believe that there are two reasons for organizations' fear of agile development, and both, unfortunately, are inherently tied to the agile style. Thus, it may be difficult to overcome these obstacles, but those who understand them can surmount them. Let's explore the two reasons for organizational resistance to agile development.

CHANGING REQUIREMENTS

As flight attendants on Southwest Airlines are fond of cautioning passengers when opening the overhead compartments upon landing, "Shift happens." A fundamental motivation for going agile is that customer requirements will shift before you can deliver your product -- if indeed you even have a firm grasp of the requirements at the outset. But for decades, we have preached that "best practice" is to thoroughly establish requirements before beginning any design work -- a say-what-you'll-do-then-do-what-you-say approach. This idea is appealing and reduces wasted effort, but it doesn't fit with reality. Donald Reinertsen has collected data on requirements change for several years; some of his findings are the following [3]:

- Out of hundreds of projects, there is no case in which requirements remained stable throughout design.
- Of more than 200 product developers, fewer than 5% had complete requirements before beginning design.
- On average, design commenced with only 58% of requirements specified.

Since Reinertsen collected the data, his database has more than doubled, but his conclusions remain the same [2].

If requirements are going to change, wouldn't we be better off acknowledging this and building our processes to accommodate it -- precisely the purpose of agile methods -- rather than denying it? I've found that when confronted with the reality of changing requirements, managers accept the situation much better than developers. Developers seem burned out from years of incomplete requirements and seemingly arbitrarily changing requirements. They wonder why their counterparts in marketing can't stabilize the requirements and save developers from frustrating rework.

Thus we arrive at the first of two reasons why adopting agile development is difficult. Developers know they need a plan and a process; hacking is unacceptable. So they expect complete "correct" requirements before they begin (though the pressure of business precludes waiting for this completeness, as Reinertsen's statistics above illustrate). In an effort to protect themselves, some developers adopt a contract mentality: "If the requirements change on us, it voids all our prior schedule commitments." Unfortunately, such predictability violates the fundamental premise of agility: shift happens.

To overcome this dilemma and thus adopt an agile approach, developers -- and their counterparts in marketing and management -- must adopt the new viewpoint that requirements will always be uncertain and subject to change. The solution is to develop something and get it into users' hands as early as possible, then be prepared to react to the feedback quickly: that is, user-led development. This concept won't come easily for those who believe that the user can be captured completely through requirements documents. Nor will developers with a great deal of technical training accept the idea easily; they believe that they know how to write software best and that users should accept this and appreciate their leading role.

What can we do about this situation? First, the solution is not to return to hacking. Agile methods provide a great deal of process and control, but the sequence of the process is arranged to allow constant user feedback and corresponding adjustments of the product under development. Second, developers will be more open to changing requirements if they have firsthand exposure to actual users. They need to develop an appreciation for the fact that users seldom read user manuals, frequently receive inadequate training, and sometimes try to use products in ways never intended by the developers. Yes, cultivating this user sensitivity does take developers out of the office and "distract" them from their primary job of writing code, but it is essential in creating developers who can balance initial product requirements with marketplace realities.

AN UNCERTAIN OUTCOME

The second reason that agile methods are unsettling pertains more to management than to developers. Conventional projects have a well-defined outcome, and they are managed toward that outcome. In contrast, agile projects posit only an initial guess at an outcome, and management regularly reassesses what the actual outcome should be. Thus, managers must live with the uncertainty of not knowing exactly what the project will deliver. All they know is that if the project goes well, it will deliver good value for the effort expended.

This happens because an agile project regularly reassesses where it is currently and where it's headed next. In every iteration, features are reprioritized. A feature planned initially may be bumped repeatedly and never make it into the final product. Agilists believe that this process is advantageous because it leads to the best value for the effort expended. In some cases it leads to early completion of the project because management decides that the project has achieved most of its business value and further effort would not be rewarded commensurately. (This is based on the principle that effort expended on a project reaches a point of diminishing returns, as Figure 1 indicates.)

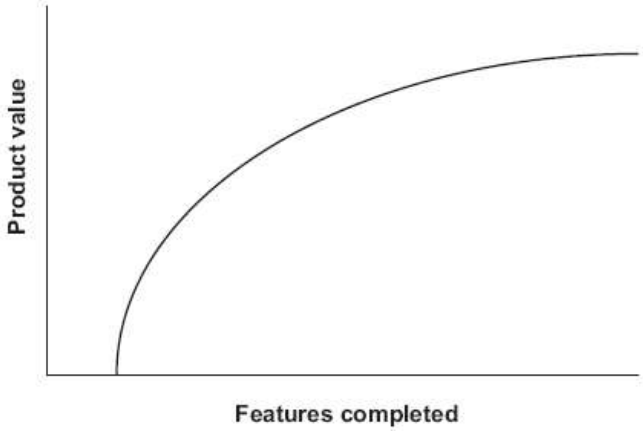

Figure 1 -- The diminishing return of effort on a project.

Some managers accept this notion and even appreciate that they are receiving the best value from their investment. Other managers and those working in other cultures that fit better with a command-and-control management style have difficulty with it. They believe that in order to manage effectively, they must know the final objective and measure progress and success relative to it. In fact, unless one's workers are intrinsically motivated, managing toward an established objective may be the best way available to achieve project success.

Most of us are trained, either formally or through on-the-job experience, that a project should have a clear goal and should be managed relative to that goal. Although the idea of receiving value is understandable, it is a softer concept and more difficult to apply in practice.

Another problem with the value-added approach, which agilists don't discuss much, is that it leaves most project decisions, including ones on which features are actually implemented, in the hands of a cross-functional team rather than under management's control. Management has to feel comfortable relinquishing this control -- and in some cases, it is actually unwise for it to do so.

This brings us to another difficulty that has received little coverage in the agile literature: agile methods depend on experienced, intrinsically motivated workers who know how to and are inclined to deliver value rather than simply check off tasks completed. In my consulting and training, I constantly encourage the development of such individuals. Nonetheless, in most companies, self- motivated employees are in short supply.

I write this *Executive Update* from Asia, where dynamic, developing economies encourage job hopping and dissuade managers from developing problem-solving and value-adding skills among employees with little allegiance to the company. In the current uncertain business environment in the US -- partly exacerbated by the migration of engineering jobs from the US to Asia -- many of the same factors encourage a similar command-and-control management style that discourages releasing developers to add value.

What can we do to encourage managers to move toward a value-adding style? Unfortunately, the process seems to be a slow one of education. The agile project management "Declaration of Interdependence" emphasizes adding value [1]. We need to move beyond the principles, however, and, with studies in hand, actually demonstrate to hard-pressed managers how a focus on value improves the bottom line. Beyond this, it would be nice to have case studies illustrating projects that were completed early and satisfactorily with corresponding savings.

Focusing on the project outcome rather than providing value is a symptom of putting business metrics before satisfying customers. If we involve customers more intimately in our development projects and include customer satisfaction metrics in managers' evaluation criteria, managers' emphasis is likely to shift from fulfilling requirements to satisfying customers.

REFERENCES

1. Anderson, David et al. "The Declaration of Interdependence." 17 February 2005 (http://pmdeclarationofinterdependence.org).

2. Reinertsen, Donald. Personal communication with the author, 6 May 2005.

3. Thomke, Stefan, and Donald Reinertsen. "Agile Product Development: Managing Development Flexibility in Uncertain Environments." *California Management Review*, Vol. 41, No. 1, Fall 1998, pp. 8-30.

ABOUT THE AUTHOR

Preston G. Smith is a management consultant and trainer specializing in accelerated development of new products. In his 20 years of guiding managers toward faster and more flexible development, he has noted a strange resistance to such techniques that seem so beneficial on the surface. This ExecutiveUpdate *opens a discussion on this ambivalence, which is often an obstacle to successful implementation.*

Mr. Smith is coauthor of both Developing Products in Half the Time *and* Proactive Risk Management. *He holds an engineering Ph.D. from Stanford University and is a Certified Management Consultant. He can be reached at preston@NewProductDynamics.com.*