

Breaking the Facade of Truth: An Introspective View into and a Case Study About the “Apparent Truths” of Agile

by David Spann, Senior Consultant, Cutter Consortium

Each of the lessons, or “apparent truths,” presented in this *Executive Report* was discovered by one or more firms who actually went through an agile implementation process. The lessons include a need for a single definition of agile, a focus on agile as a quality process applied to software development, the anecdotal evidence that companies are drowning in “technical debt,” a few lessons related to leadership skills, and a caution about using Scrum.

Executive
Report

Access to the Experts

Cutter Consortium is a unique IT advisory firm, comprising a group of more than 100 internationally recognized experts who have come together to offer content, consulting, and training to our clients. These experts are committed to delivering top-level, critical, and objective advice. They have done, and are doing, groundbreaking work in organizations worldwide, helping companies deal with issues in the core areas of software development and agile project management, enterprise architecture, business technology trends and strategies, innovation, enterprise risk management, metrics, and sourcing.

Cutter offers a different value proposition than other IT research firms: We give you *Access to the Experts*. You get practitioners' points of view, derived from hands-on experience with the same critical issues you are facing, not the perspective of a desk-bound analyst who can only make predictions and observations on what's happening in the marketplace. With Cutter Consortium, you get the best practices and lessons learned from the world's leading experts — experts who are implementing these techniques at companies like yours right now.

Cutter's clients are able to tap into its expertise in a variety of formats, including print and online advisory services and journals, mentoring, workshops, training, and consulting. And by customizing our information products, training, and consulting services, you get the solutions you need while staying within your budget.

Cutter Consortium's philosophy is that there is no single right solution for all enterprises, or all departments within one enterprise, or even all projects within a department. Cutter believes that the complexity of the business technology issues confronting corporations today demands multiple detailed perspectives from which a company can view its opportunities and risks in order to make the right strategic and tactical decisions. The simplistic pronouncements other analyst firms make do not take into account the unique situation of each organization. This is another reason to present the several sides to each issue: to enable clients to determine the course of action that best fits their unique situation.

Expert Consultants

Cutter Consortium products and services are provided by the top thinkers in IT today — a distinguished group of internationally recognized experts committed to providing top-level, critical, objective advice. They create all the written deliverables and perform all the consulting. That's why we say Cutter Consortium gives you *Access to the Experts*.

For more information, contact Cutter Consortium at +1 781 648 8700 or sales@cutter.com.



The Agile Product & Project Management Advisory Service Executive Report is published by the Cutter Consortium, 37 Broadway, Suite 1, Arlington, MA 02474-5552, USA. Client Services: Tel: +1 781 641 9876; Fax: +1 781 648 8707; E-mail: service@cutter.com; Web site: www.cutter.com. Group Publisher: Kara Letourneau, E-mail: kletourneau@cutter.com. Managing Editor: Cindy Swain, E-mail: cswain@cutter.com. ISSN: 1536-2981.

©2008 Cutter Consortium. All rights reserved. Unauthorized reproduction in any form, including photocopying, faxing, image scanning, and downloading electronic copies, is against the law. Reprints make an excellent training tool. For more information about reprints and/or back issues of Cutter Consortium publications, call +1 781 648 8700 or e-mail service@cutter.com.



Rob Austin

Ron Blitstein

Christine Davis

Tom DeMarco

Lynne Elyn

Jim Highsmith

Tim Lister

Lou Mazzucchelli

Ken Orr

Mark Seiden

Ed Yourdon

Breaking the Facade of Truth:

An Introspective View into and a Case Study About the “Apparent Truths” of Agile

THIS MONTH'S AUTHOR



David Spann
Senior Consultant, Cutter Consortium

IN THIS ISSUE

- 4 The Apparent Truths
- 6 Lessons Learned
- 14 Case Study
- 17 Summary
- 18 Dedication
- 18 Endnotes
- 18 About the Author

During the past seven-plus years, I have been blessed with opportunities to observe, train, coach, and reflect on the agile software development movement and its implications for greater organizational change. My experience in this movement began in late 2000 and early 2001 when I had the opportunity to facilitate and observe a group of developers using the European version called the Dynamic Systems Development Method (DSDM). They took on an almost impossible corporate need, deployed their beta test within the world's largest retailer, and built such a joyous and enlivening work community that the entire firm (from increased profits to significantly improved moral) was positively affected. Then, in the spring of 2002, I hosted the very first one-day *Agile Software Development* conference at Westminster College's Gore School of Business, just one year after the Agile Manifesto was created. More than 250 people came to hear the panel discussion, which included Cutter Senior Consultant Alistair Cockburn and Cutter Fellow Jim Highsmith, Ron Holiday from Fidelity, Michael Patten from Lowery & Associates, and Russell Stay from Semantec. (Note: this conference was held in the graduate business school, not by the IT program.)

Since that wonderful beginning, I have helped host the first three *Executive Summits* at the 2003, 2004, and 2005 *Agile* conferences, joined the Cutter Consortium team of Senior Consultants, and taught and coached agile management for Westminster's Graduate Business Program, large and small companies, government, and nonprofits throughout North America. And the reason I keep doing all of this is because I get to see groups of people discover a more humane way to solve problems, build solutions, and profitably change their organizations. Through the agile values, principles, and practices, my clients and I get to continuously experience what it means to organize around a common purpose, offer our valuable talents, and thereby jointly enjoy the results of our work.

The purpose of this *Executive Report* is to look back, as if we were at the end of an agile project's first iteration,

reflect on the things I've learned, offer some recommendations for change, and provide a case study illustrating many of these recommendations. The intent in this report, therefore, is to bring forward a few "apparent truths" to consider in an agile implementation effort, to provoke action where needed, and to offer recommendations where appropriate so that those of us in the "agile movement" might debate these topics more clearly and focus the corporate attention where it needs to belong: on the organizational outcomes created by collaborative work and adaptive technical practices.

The six lessons that have emerged from this reflection were discovered by one or more of my clients who went through an agile implementation process. The topics include what it means to be agile, the fact that agile is really a quality process applied to software development, the anecdotal evidence that most companies are in way over their head in "technical debt," a few lessons related to leadership skills, and what some companies are experiencing trying to apply Scrum.

Before I get into the specifics, I want to share a story about a prospective executive client who was trying to go beyond the agile rhetoric and get to the real technical, business, and behavioral issues, but was caught in a common organizational trap: unable to express publicly, except to me, the apparent truths related to becoming an agile organization. After a fairly normal conversation about customer involvement, iterative technical design, development and testing, and the need for collaborative team interactions, this prospective client surprised me by saying he understood the agile intent but faced fundamental organizational problems that seemed insurmountable: the first problem was that the firm was very risk averse and fearful of any process change (or any change for that matter); the second, more important, problem was that the organization was living within a "facade of truth," meaning the inability to say what needs to be said regardless of the impact on the individual or the firm. The company was enacting its

risk-averse nature and living behind this facade of truth to ensure that "what was said" was "what was expected." No one actually knew whether the opinion being stated was actually truthful in the view of the person giving it or if it was simply an act to gain validation and support for some other initiative.

Since the Agile Manifesto calls us all to focus on "individuals and interactions," I offer the dialogue below to remind us that even when we're talking about agile values, principles, and practices, we need to consider the apparent truths supporting, or as in this case, standing in the way of our progress. Similarly, the story is offered to remind us to be vigilant in looking behind those truths for lessons that will help us move forward and to reflect every so often on our experiences and potentially design new strategies for our joint future.

Here is how I remember the conversation:

"The first problem we have," my prospective client said, "is that our organization is conflict-averse. We will avoid disagreements at almost any cost, even if resolving that conflict will move our business forward in a significant way. For instance, just last week we had to decide whether to outsource some of our development staff, but instead of creating a business case with pros and cons for each alternative and developing a consensual outcome, we adjourned the meeting at the first sign of discontent. The decision was then made by our CIO and published in a memo (we later learned that he'd already made the decision before the meeting and was hoping for a quick and supportive meeting on the subject)."

"Unfortunately, he and others have a tendency to call meetings where it looks like a consensus decision is preferred, but when it comes down to our ability to conduct a hearty debate, we fail to do the hard work to reach agreement and support for each other."

I confirmed that I'd seen that problem or its corollary (i.e., when the hearty debate rages on without any resolution so everyone can feel good about being in the debate without having to commit to a certain outcome; in either case, there is no resolution and no buy-in to an agreeable outcome) in many different types of organizations. In addition, I said, "like the old statement that says a company can be judged by the complexity of its technical systems, I believe that the potential for organizational success can be judged on how well a company's meetings are managed: is there a stated and meaningful purpose, are the right people involved and present, is there something to work on, and can the attendees act in concert with each other once the meeting is finished? We seem to be trained to disagree, but the really successful leaders and their organizations find ways to bring disparate opinions to a supportive resolution in a timely way."

The second problem wasn't surprising except, like a bad aftertaste, I've been thinking of his comments ever since:



“Our second problem,” he said, “is that we have a ‘facade of truth’ that permeates almost every layer of this organization. If we want something accomplished or if we want to be seen as successful, we weigh what the other might be thinking before we speak. And even then, we may not speak our truth, but instead say the truth we think the other wants to hear so that we can position ourselves to get what we need or want. For instance, I found myself doing that very thing just yesterday.”

“In an executive leadership meeting, I was asked what I thought about the new marketing plan.” [Note: this prospective executive client was the firm’s CIO.] “Instead of telling the group I thought the plan was off base and too indirect, I complimented the CMO on the amount of work he and his team must have put into the effort and said how much I looked forward to working with him on implementing the plan.”

“Sitting here today, I can see that my motivation was clear: marketing has never really driven the messaging in this company, our sales department does that. And I needed his support to purchase a new enterprise application I was going to advocate during the next agenda item. While that strategy worked — he did support me and my department will be able to expend more resources on this new technology — I am now hoping I didn’t sell out the firm’s overall best interests for the sake of optimizing my department’s needs.”

Once we talked about these problems and how agile’s transparent team practices, as well as how my approach to executive coaching and organizational transformation could unveil many of these challenges, he decided to take some time and think about the implications. Ultimately, he followed the cultural pattern he had described to a tee by first ignoring my followup phone calls and e-mails and then officially postponing the decision to implement agile in his firm (I got a “cc” to the official e-mail — no personal phone call).

While I was disappointed, his comments and his behaviors made me think about risk aversion and its effect on “truth telling” generally. The more provocative inquiry, however, came when I thought about my personal facade of truth, specifically focused on those things in my agile consulting practice that I know are true but have previously been unwilling to disclose. Why wasn’t I talking about those things and just letting the “chips fall where they may?” The simple answer is I thought that by disclosing these issues, I’d lose potential clients, just as I did with the CIO described above.

In an attempt to overcome that apparent fear, peel back at least one layer of my metaphorical “truth onion,” and speak more honestly about agile, I decided to write this report about some of the more provocative lessons my client companies and I have learned about the organizational dynamics within an agile experience. Each of the

apparent truths was discovered by firms who actually went through some agile implementation process (even if they didn’t discover the issue until the consultants had all gone home).

For example, most firms want to know what it means to be agile, but outside the stated values and principles from the Agile Manifesto and Declaration of Interdependence,¹ the agile expectations are not clear; there is no single definition for agile, and there is no agreed-upon methodology. Are there some operating practices that should be called out and used to judge the corporate competency in agile, and if so, what are they?

Likewise, even though we talk about how important executive support is to the success of an agile implementation, we hardly ever define how much involvement (versus mere support) is actually necessary to shift how the organization accomplishes its work. The fact is that taking on an agile implementation process is tantamount to an organizational change effort and needs to be managed as such.

In terms of specific management practices, in this report I share what I’ve learned about the importance of technical debt, overselling capacity, executives sweating the small stuff, decision making, and many more issues I call “agile deal killers.”

Of all the recommendations I discuss in this report, I suggest two practices for immediate implementation. The first is for every level of management to implement daily stand-ups. Imagine every team meeting for 10 to 15 minutes every day discussing what is most important and then getting on with their work. That one practice, assuming the results are transparent to the organization, will force everyone to be clear about the business drivers and the overall strategy. Likewise, it will force leaders to be more transparent and collaborative in their efforts.

The second recommendation is similar. To focus people’s work on purposeful outcomes, I would post a sign on every conference room door (and require the same be included on every electronic meeting invitation) that says: “If the meeting you are about to attend has no stated *purpose*, please return to something that does.”

If the meeting you are about to attend has no stated *purpose*, please return to something that does.

As far as I know, these topics are not commonly shared in agile training sessions or coaching scenarios, but if they were, they might help broaden the purview of agile from being basic software development to having a more quality-focused, organizational impact. To that end, I have gone back over my notes from prior consulting efforts and identified the most compelling statements clients have made about agile, what it means to them, and the impact they discovered based on that knowledge. Even though these statements seem true to me, I am going to represent them as “apparent” truths because I’m sure that behind these statements is yet another layer of truth to be unveiled and discovered.

But if the process of “going agile” isn’t managed well, the organization’s “antibodies” will emerge to thwart the invasion of any new idea.

THE APPARENT TRUTHS

The six lessons, or apparent truths, presented in this report, their underlying stories, and my recommendations are offered as a reflection on what has worked well, what might be improved, and the things organizations may want to consider in their agile implementation effort. In fact, I am convinced that without understanding these stories and the techniques necessary to ensure successful outcomes, organizations will continue to try applying the next new agile technique, one after the other, always hoping for better results. But if the process of “going agile” isn’t managed well, the organization’s “antibodies” will emerge to thwart the invasion of any new idea — even if the agile cure is logically more appropriate.

May the lessons learned here better ensure your agile journey is enjoyable and productive.

Apparent Truth #1: There Is No Single, Agreed-Upon Definition of Agile

In 2002, Jim Highsmith included the following methodologies in *Agile Software Development Ecosystems*²: XP, Crystal Methods, Scrum, DSDM, lean development, Adaptive Software Development (ASD), and Feature-Driven Development (FDD). Even though each of these approaches includes collaborative, iterative, and quality-based practices, there is no standard on which to say whether they or any other methodology is more

agile than the others. While this reality assures an individual or an organization the ability to pick and choose the practices that work best for them and call them agile, it also results in some organizations trying to be faster and cheaper without spending time and resources on being better. This report offers a set of ability statements (aka “stories”) that define a barely sufficient level of technical, customer, and business expectations for those organizations that want to be known for their “agility.” These expectations will also be used throughout the rest of the report to set a standard for testing, decision making, leadership, cross-functional involvement, and even the type of methodology an organization might want to choose.

Apparent Truth #2: Agile Is a Quality Initiative

One of the first things I’m interested in exploring as I conduct a firm’s agile readiness assessment is its focus on quality assurance (QA). When I was conducting one of these assessments and making suggestions about ways the firm could move its QA practices upstream into the development cycle (and even back up into the sales department), the leader of QA said: “This isn’t new; it’s exactly like any other quality improvement program I’ve seen.” He was right. The ideas of continual improvement are baked into every one of the agile methodologies, as is the idea of meeting the customer’s expectations, which are two of the foundational components of any quality initiative. In fact, the intent of most agile philosophies and practices is found in other well-known, quality-based methodologies, such as lean manufacturing and design-build (for the construction sector).

Apparent Truth #3: Companies Need to Invest in Reducing their Technical Debt

During the third day of one of my agile management classes, after a discussion about iterative testing expectations, one student said, “I’m sorry, but you just don’t understand our situation. It takes us at least five months to do anything that might be construed as integration testing, so how can we ever reach your definition of being agile?” Another client wanted to know how agile would fix its inability to adapt a highly integrated code base written in COBOL that had poor documentation and only one of the original developers remained on staff. In both cases, and in many other cases similar to these, my clients are finding they are in fact choosing to use agile methods and practices to build their way to “software hell” faster than they did with a traditional waterfall approach.

Apparent Truth #4: Your Organization Will Change if You Adopt Agile

Much of the discussion about agile has in the past been about creating “working software,” team dynamics, technical practices, and even the project management required to get effective and acceptable code delivered and/or deployed. What teams quickly find out is that they depend on other organizational dynamics outside their purview, such as securing customer involvement, issue escalation, and resource allocation; even facility management becomes important. When these issues arise, unless they are in a small enough company where there is only one team working, they will need to become aligned with the rest of the organization to ensure the best collaboration possible.

However, what most leaders discover is that by taking on agile as their development methodology, they come face-to-face with organizational change. Change will be provoked by needing to cross previously undisputed functional boundaries to involve customers, stakeholders, and other managers, entering into agreements for resource sharing, and potentially renegotiating work processes with other departments, such as sales, contracting, finance/accounting, and human resources. To have a reasonably orderly transfer from a traditional or function-focused work method to one that is more agile and adaptive requires behavioral modeling, management, and an ongoing “enablement team,” as if you were managing any other purposeful change initiative.

There is no magic pill: being faster, better, and cheaper requires collaborative work, even at the organization’s leadership/executive levels.

Agile implementation in an organization requires an executive enablement team to manage the following details associated with almost every agile implementation. The team must:

- Consider issues related to integration and release expectations
- Assess and resolve technical debt
- “Sweat” the right level of detail
- Coordinate sales’ commitments with development capacity
- Negotiate appropriate delivery dates
- Reduce organizational insanity with fewer, well-defined projects/initiatives
- Make sure there is a clear definition for decision making (empowerment)

- Make meetings purposeful and productive
- Develop a rapid issue resolution process
- Address other topics as they emerge

Apparent Truth #5: Agile Needs Supportive and Determined Leaders for Long-Term Success

An agile implementation, like any organizational change, takes approximately 18 months to ensure the new practices become inculcated (and that timeline depends on a focused enablement team being in place and managing the process). In that time frame, I have noticed that long-lasting agile initiatives have required supportive, determined, and involved leadership from at least two people. Without them, I have watched too many well-intentioned agile initiatives go up in flames.

Typically, there are two types of leaders who make the agile transformation possible. The first is an executive sponsor who has the resources and patience to stay the course while others get used to the uncertainty and chaos before they begin to enact the agile practices as their own. The second leader is like a “chief of staff” over the agile implementation process. This person leads the enablement team, champions the cause when others are ready to throw in the towel, and makes sure the major organizational issues are resolved and/or brings in the executive sponsor as necessary. While these can be the same person, the risk of failure goes up if the organization depends on one individual to serve both roles and that person leaves for any reason, whether by promotion, retirement, severance, and so on.

Apparent Truth #6: Scrum Is a Great Agile Management Methodology (But Don’t Forget the Technological and Product-Focused Practices)

Scrum, as a management methodology, is elegant in its design: offering just enough practical advice to agile teams so they do not overcomplicate the development lifecycle with too much ceremony and documentation. However, it is a management methodology, not a software development methodology; therefore, it assumes such matters as sufficient architectural design, appropriate technical environments, and protocols for development, testing, and production are in place. Similarly, it assumes someone or some group is managing the customer/product community so that the product owner can represent a unified and prioritized description of the business need. Unfortunately, my experience says that firms that take on Scrum do not understand these two issues and thereby focus on doing things faster and cheaper but can and often do miss doing things better.

LESSONS LEARNED

During most retrospective workshops, we ask participants to acknowledge the lessons they've learned in the last iteration and discuss what might be done to improve in the next timebox. The following section enlarges the introductory conversation above by offering new insights and recommendations for those interested in launching or improving their agile implementation effort.

There is no one accepted definition of agile that encompasses all the techniques and processes currently under the agile banner.

Apparent Truth #1: There Is No Single, Agreed-Upon Definition of Agile

Most corporate executives want to know what agile is and the differences between it and what they currently do. Unfortunately, there is no one clear way to answer these inquiries, because there is no one accepted definition of agile that encompasses all the techniques and processes currently under the agile banner. In *Agile Software Development Ecosystems*, Highsmith does a great job of describing the processes (including XP, Scrum, DSDM, lean software development, ASD, Crystal Clear, test-driven development (TDD), and FDD).³ (Note: I would add Industrial XP, which was introduced in 2003 by Cutter Senior Consultant Josh Kerievsky and the folks at Industrial Logic.)

Even though each of these approaches includes collaborative, iterative, and quality-based practices, there is no standard on which to say whether these processes or any other methodology is more agile than another. While this reality affords an individual or an organization the ability to pick and choose the practices that work best for them, it also results in some organizations trying to choose those practices that seem to be faster and cheaper without spending time and resources on implementing basic quality practices that would help them be better over the long term.

The closest we can come to a definition are the Agile Manifesto and its supporting 12 principles created in 2001 and the Declaration of Interdependence created in 2005.⁴ As good as these two documents are in defining values, principles, and including some specific practices, such as iterative development and customer involvement, they are not clear as to what is and what is not agile. For instance, can teams that are not colocated be

called agile? If “stories” are documented in a business requirements document, does that mean the project is automatically not agile? If iterations take longer than four weeks, is that agile? If the preproject planning takes more than eight weeks and includes multiple prototypes and architectural design sessions, is that agile? In some agile methodologies, the answer to these questions may be yes, and in others it is a definitive no.

So what are the minimal building blocks upon which agile is formed? What are the acceptance criteria for “being agile?” What are the expectations an organization should have if it is considering being agile? In an attempt to provide this level of definition, I compared most of the methodologies and in the process discovered a minimal subset of ability statements (aka “stories”) that define a barely sufficient level of technical, customer, and business expectations for those organizations that want to be known for their “agility.” These expectations will also be used throughout the rest of the report to set a standard for testing, decision making, leadership, cross-functional involvement, and even the type of methodology an organization might want to choose.

Recommendation

While I know this may be tantamount to heresy for some readers, I offer the following list of seven statements as my own definition for what is and what is not agile. If you are not ready to do these things — all of them — you aren't ready to be agile. This list also acts as a foundation for the rest of the report and, I hope, will instigate further definitional consolidation within the agile community.

Agile is an organization's ability to:

1. Set up every initiative with sufficient knowledge about the business case (including customer and ROI expectations), technical architectural design, development protocols, business process impacts, and other team needs, such as basic working agreements, risk management, facilities planning, and resource sharing.
 - **Acceptance Criteria:**
 - The cross-functional team of developers and stakeholders understands enough to begin and manage the project through at least the first iteration.
 - The executive sponsor feels confident enough to expend organizational resources on the initiative (at least through the first iteration).
2. Collaborate with major stakeholders (e.g., customers, technical team, and management) throughout product design, development, and delivery/delivery.

■ **Acceptance Criteria:**

- o An agreed-upon process for communication, including what will be communicated, when, and how is established between the development team and the stakeholder community.
- o Those representing the customer community are available on an as-needed basis to the development team, which may require a full-time person dedicated to a development team.

3. Develop a highly collaborative (i.e., cross-functional) work environment in which people enjoy successful outcomes.

■ **Acceptance Criteria:**

- o Teams and appropriate stakeholders take time to celebrate success during and after the initiative is completed. Celebrations and other types of reward are focused on the team's success.

4. Work in short, iterative timeboxes lasting two to six weeks that include planning and design ("barely sufficient to go forward"), development, testing, and reflection.

■ **Acceptance Criteria:**

- o Iterations end on their planned date (they are not extended for any reason).
- o Teams stop at the end of an iteration to demonstrate their work and reflect on what worked well, what could be improved, and things they want to try differently.
- o Short (no longer than 20 minutes) debriefing sessions held daily to cover accomplishments, upcoming work, and major roadblocks or issues.
- o Issues are escalated and resolved quickly.
- o Meetings are designed with a clear purpose, managed for effectiveness, and adjourned as soon as possible.

5. Complete all acceptance tests within an iteration and throughout the lifecycle.

■ **Acceptance Criteria:**

- o All functionality, features, or stories pass three tests for that item to be considered "done": unit, integration, and customer acceptance.
- o Customer acceptance is based on the functionality meeting its expectations.

6. Develop, create, and build the most important customer/end-user needs first (which may require a

corollary ability to understand the business value within the context of corporate strategy and project portfolio decisions).

■ **Acceptance Criteria:**

- o Stakeholders representing the firm's business interests and those representing the customer need have an agreed-upon process for prioritizing the workload.
- o Development team members can recite the project/initiative business purpose and high-level objectives at all times.
- o Review of the business objectives is included in every reflection or retrospective discussion.

7. Iteratively reflect upon, and then adapt to, lessons learned.

■ **Acceptance Criteria:**

- o Reflection or retrospective discussions are held at the end of every iteration to cover what worked well, what could be improved, and any changes the team might want to consider in the next iteration.
- o Lessons learned are communicated as appropriate to other teams and interested parties in the organization.

The "agile game" is really very simple: someone, with the best intentions possible, gathers together a small group of like-minded but differently skilled people. They envision the possibilities, put their trust in each other, and begin to work collaboratively. Agile, as I've come to know it, simply calls out sufficient protocols for these groups to work well together: designing and building small increments of the work, testing outcomes with customers/end users in short, iterative time frames, and adapting as best they can.

Is there really much more to it than helping well-intentioned people work together toward a common goal? Doesn't it come down to a group of people who remain present to the circumstances, curious about the possibilities, supportive of each other, and willing to continually learn about what might work better the next time?

Apparent Truth #2: Agile Is a Quality Initiative

The idea of a continual improvement process is baked into every one of the agile methodologies simply because development is expected to occur iteratively along with technical testing, customer acceptance, and a rigorous process of daily and iterative reflections about

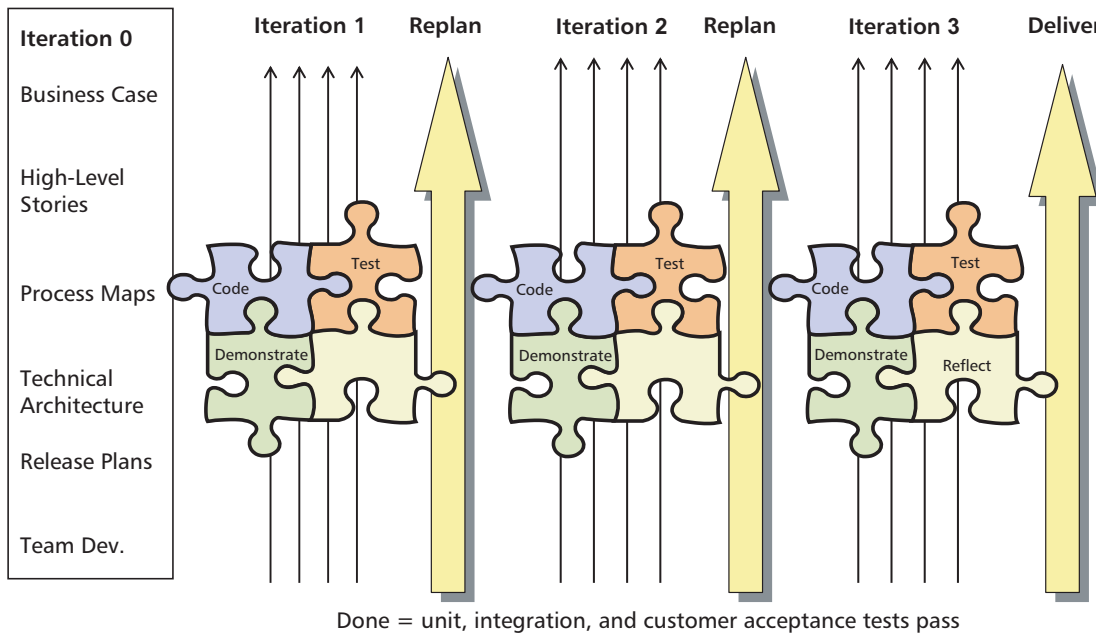


Figure 1 — The agile game.

the health of the project/initiative (see Figure 1). These simple practices assure both the product’s technical quality and that customer expectations for that product are met, which are the two foundational components of any quality initiative. However, too many firms are taking on agile not as a quality initiative, like their counterparts in lean manufacturing and design-build, but instead in a way that focuses solely on finding a faster and cheaper method of delivering an end product.

As one of my clients said, “Agile is really all about applying common sense”; to which I amended, “Agile is all about translating common sense into common practice.” It all comes down to how well people work together, how they can better use their time, and how the outputs of their work are directly proportional to

their interest, knowledge, and willingness to continually improve.

To that end, when you look at agile as a work process instead of as a philosophy or set of value statements, it is like any other quality initiative: people interested in reducing waste, increasing productivity, and improving working relationships. It is all about focusing on purpose and then iteratively, in short bursts of time, developing solutions, testing their efficacy, reflecting on lessons learned, and then adapting to those changes as appropriate. In other words, agile is just like the plan, do, check, act (PDCA) model for continual improvement originally defined by Walter Shewhart, made popular by W. Edwards Deming, and codified by the American Society for Quality (see Figure 2).

The PDCA model can be thought of as follows:

- **Plan.** Identify an opportunity.
- **Do.** Implement on a small scale.
- **Check.** Analyze the results and determine whether the change made a difference.
- **Act.** If the change was successful, implement it on a wider scale and continuously assess your results. If the change did not work, begin the cycle again.

While most agile methods spend quite a bit of time on “barely sufficient” steps to meet the “plan” and iterative development practices to meet the “do,” the piece that really defines whether a firm is agile rests in the “check” or testing portion of the model. This is important

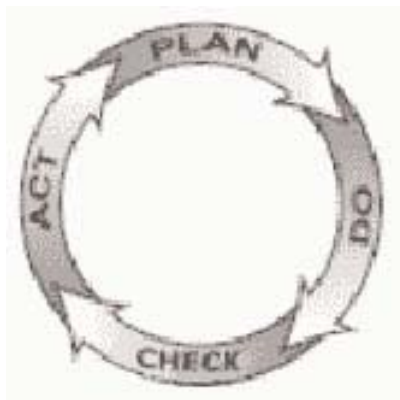


Figure 2 — PDCA model.

because if an organization cannot conduct the customer acceptance, unit, and integration tests inside the same iteration in which the code was completed (see definition for technical “doneness” above), then either the development team will need to go back at some later time and take its eye off the current development objectives or turn it over to a support or maintenance team to fix the inherent bugs, which requires a whole host of additional resources, time, and technical documentation.

More important than when bugs get resolved is the implication that when QA is at the end of a development cycle, there may be more pervasive architectural or systemic design problems. Even though I’m not an architect or even a technologist, I know from experience that these types of situations point toward a highly integrated and fragile code base. In fact, my experience says that when QA takes weeks and even months to complete its tests, the underlying code base is so highly integrated that a development team using agile management practices without resolving this issue is simply adding to the problem faster than it would in a more traditional waterfall approach. (See the next apparent truth relating to technical debt.)

The main point here is that agile must be seen as a quality initiative that requires investment in technical quality, meeting or exceeding customer expectations, and reducing time of delivery. The quality movement has been successful in driving home the need for these investments in manufacturing and in construction, but for some reason firms are less inclined to make similar types of investments in the processes leading to a technological solution. However, there is no getting by with great management practices, such as customer involvement, product backlog prioritization, demonstration, and retrospective workshops, without having a similar interest in assuring the technical quality through automated unit and integration testing.

Recommendation

Firms that take on an agile implementation should understand the total cost of an agile implementation effort, including the costs of potentially refactoring their code base and of automating technical tests, as well as organizational costs associated with cross-functional participation, especially as it relates to involving the customer/product owners.

Apparent Truth #3: Companies Need to Invest in Reducing their Technical Debt

If I could rewrite the Sarbanes-Oxley Act, I’d add one clause to protect investors, which would read

something like, “The firm’s code base must be sufficiently compartmentalized/modularized so that automated testing can be applied and results returned within a 24-hour period.” While I agree that this would be too prescriptive, the intent would be to ensure firms invest in reducing their enterprise system’s fragility and reducing the risk that the firm collapses from a technical glitch. Unfortunately, because of this problem, way too many of the firms I know — even if they are interested in adopting an agile methodology — are in serious jeopardy of collapsing under their own technological debt.

There are two corporate examples I use to describe this situation more clearly: the first has to do with a highly integrated code base that forces a firm to spend lots of money replacing its entire enterprise system, and the other relates to another firm spending equally huge sums of cash trying to force fit a new enterprise system into a highly integrated code base written in COBOL.

In the first example, the CEO showed up on the first day of my agile project management class and said, “Next Monday this firm is officially agile; I’m staking my reputation on it, so learn well and let’s get started.” (Note: this executive didn’t want to spend time on an agile readiness assessment but wanted to go straight to training, and, unfortunately, I let him convince me this was a prudent approach.) When he left you could feel the participants’ anxiety but I couldn’t get anyone to describe what was going on. So I proceeded until almost the end of the third day, when someone had enough courage to describe the situation by drawing a fishbone diagram (see Figure 3) and telling the following story:

“Imagine,” the individual said, “that the main horizontal line represents one of many product lines supported by this company’s system. Then imagine that for the largest 150 international customers, we create a separate but integrated branch of applications for each of them. And, then imagine that for every one of those branches we have at least 20 additional and again highly integrated applications

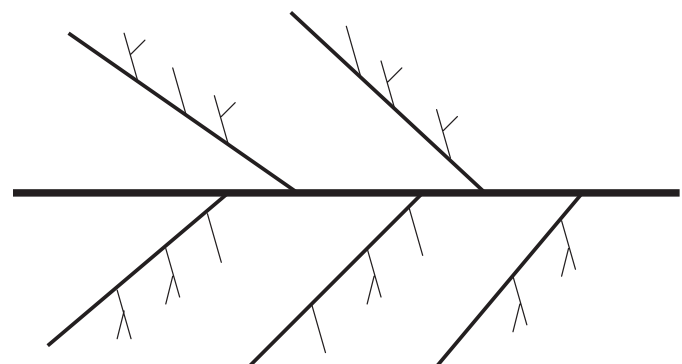


Figure 3 — Fishbone diagram.

that each meets some specific customer's needs. The kicker is that because this code base is so highly integrated and complicated, and because we do testing manually, all of our testing is left until development is completed."

So, I asked, "How long does it take for you to complete the testing phase on a normal-sized application project?"

She responded, "Five to six months! Now you see why most of us in this class don't understand how we are going to be agile by next Monday."

In the end, the executives with this firm, even though they were interested in assuring customer involvement through stories, user acceptance testing, and demonstrations, were not interested in developing automated unit and integration testing protocols, because it would force them to invest in refactoring and/or replacing their overly complicated system. See Figure 4 for more on the costs of technical debt.

The second corporate example I use is similar to this one, except that I spent almost three months with a client trying to help them understand that they needed technical expertise more than they needed management expertise. They were trying to use an agile approach to implement a new enterprise-wide application, written in .NET, into a highly integrated code base written in COBOL. To make matters worse, there was so much technical documentation on that original system that no one understood it, and only one developer was left on staff who was around when it was created. Their only saving grace (or curse) was they had more than

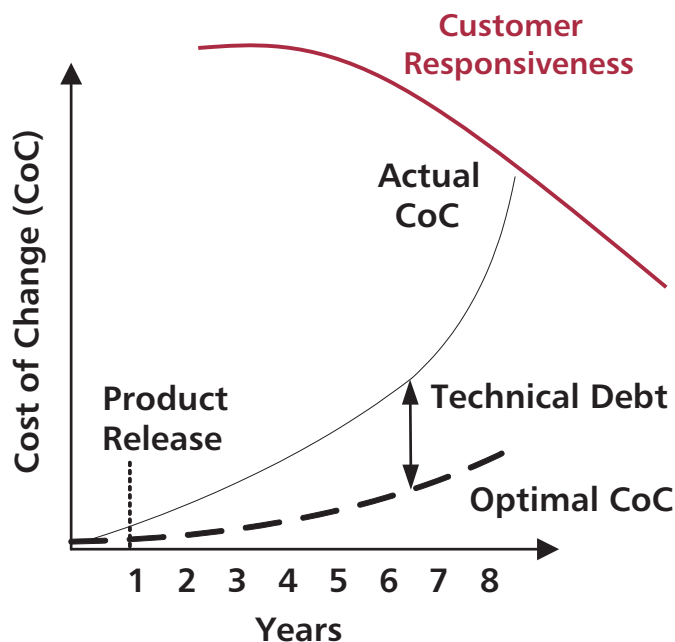
\$2 billion in "free cash," so that no one had any sense of urgency, and tough decisions were being postponed and/or avoided daily. Eventually, I had to fire myself, because I saw no way to transform the organization without a strong leader who would integrate the business processes as well as invest in the time it would take to refactor the current system and/or replace it.

Recommendation

My recommendation when I hear about long testing cycles and very little automated testing in place is to suggest the firm conduct an assessment to analyze its technical debt and the steps necessary to reduce it sufficiently to apply automated testing techniques. Typically, if a firm is in this type of trouble, it has only three choices: (1) continue to overlook the problem and hope no one calls your bluff until after you're gone; (2) spend time to refactor parts of the system in concert with the current system; or (3) replace the system with something more modular and testable. Remember, quality initiatives such as agile require investments in the organization's processes and technology to ensure long-term system sustainability.

Apparent Truth #4: Your Organization Will Change if You Adopt Agile

Much of the discussion about agile has, in the past, been about creating "working software," team dynamics, technical practices, and even the project management skills



- Once on far right of curve, all choices are hard
- If nothing is done, it just gets worse
- In applications with high technical debt, estimating is nearly impossible
- Only three strategies:
 1. Do nothing, it gets worse
 2. Replace, high cost/risk
 3. Incremental refactoring, commitment to invest

©2008 Information Architects, Inc.

Figure 4 — Technical debt. (Source: Jim Highsmith.)

and behaviors required to get effective and acceptable code delivered and/or deployed. While this is a very prudent place to begin, most teams quickly find out they depend on other organizational dynamics, such as technical architectural issues (see the technical debt discussion above), securing customer involvement, developing processes to resolve major issues, finding ways to obtain scarce resources, and making timely budget requests; even facility management becomes important. These management issues do not go away when you adopt agile; in fact, they become more complicated because organizations are rarely used to working collaboratively and quickly across functional boundaries.

So what most leaders discover is that by taking on agile as their development methodology, they are actually involved in, but completely unprepared for, dealing with some degree of organizational change.

While it may not be obvious initially, organizations that want to implement agile will need to:

- Find ways to cross previously undisputed functional boundaries to involve customers, stakeholders, and other managers
- Ensure their technical infrastructure supports agile development practices
- Craft new human resources policies
- Potentially renegotiate work processes with such departments as sales, contracting, and finance/accounting

All of this may require significant organizational change and therefore a sufficient level of leadership involvement through the change process.

Recommendation

The key to success, if you want a reasonably orderly transfer from a traditional or task-focused work method to one that is more agile and adaptive, is to establish an executive-level enablement team (ET) and expect it to apply agile practices, such as iterative planning, rapid issue resolutions, demonstrations, retrospectives, daily stand-ups, customer stories, and so on, as it manages the agile transformation process. There is no magic pill; being faster, better, and cheaper require collaborative work, even at the organization's leadership/executive levels. Let's look more closely at specific elements of this enablement team:

- **Purpose.** The purpose of an ET is to ensure the organization defines and accepts its own set of agile values and principles and applies a set of organizationally appropriate agile practices; it must also make sure the

organization does this through a reasonable transformation process.

- **Leadership.** The ET must be led by an executive with sufficient organizational influence to keep people's attention and inspire them when the going gets tough. The risk of the agile implementation failing goes down when this leadership role can be shared between multiple top-level executives.

The third six-month period will be all about consolidating the lessons learned into a new system of doing business within the firm.

- **Team members.** The ET should be made up of people who are well situated in the firm and have sufficient positional power to get things done and make changes as necessary. You will probably want someone representing the customer/end-user community, the technology community, and those interested in the firm's business strategy. In addition, you will probably want to include someone from human resources, operations, and sales and potentially someone from audit and security.
- **Duration.** Recognize that most organizational change efforts require approximately 18 months or longer to ensure the new values, principles, and practices become commonplace. The first six months will be the most chaotic and potentially the least productive as everyone gets a handle on what is expected and all of the issues that need to be resolved. The second six months will be substantially more productive as more teams come on line and get used to the agile expectations and as many of the issues are resolved. The third six-month period will be all about consolidating the lessons learned into a new system of doing business within the firm. Be aware that if any major perturbation occurs within this 18-month period (such as a strong agile champion being replaced with very little notice), the firm will easily revert to its previous operating approach.
- **Responsibilities.** Generally, an enablement team will need to manage all of the major issues related to transforming the organization including but not limited to the following list of expectations:
 - **Define the roles and responsibilities.** Depending on the type of methodology you choose, there may be very specific roles, such as ScrumMaster, team

leader, product owner, and technical lead that need to also be well-defined within your organization's human resources system.

- **Establish appropriate coordination and communication protocols between business units.** Define how your firm will coordinate the interests of your customer/product, technology, and management communities. For example, most firms will have several business or functional units interested in the functional outcomes of a single project. And if you are using a product owner or customer representative to set the project priorities (because it's too hard to get everyone in the development room consistently enough to help the team), you'll want to define a process to ensure the voice of those stakeholders is understood and transferred appropriately and in a timely fashion. The ET will also need to ensure this level of coordination occurs within the IT department (e.g., between architects, developers, testers, and technical writers), as well as ensuring good communication between managers and levels within the hierarchy to assure progress is transparent, lessons are shared, and issues are resolved as quickly as possible.
- **Negotiate appropriate delivery dates.** If there is one role the ET has to perform, it's the role of mediator/negotiator between executive expectations for delivery dates and the ability of the development team to produce. In the early months of an agile implementation, all will be trying new things to increase their productive velocity, so the ET's role is to make sure the development team doesn't overpromise and the executive level doesn't overexpect final results.
- **Reduce organizational insanity with fewer, well-defined projects/initiatives.** When Jim Highsmith visited my Westminster College agile project management classes back in 2000 or 2001, one of my students asked him what he thought was the most important concept in his ASD approach. With very little thought, Jim said, "reducing staff fragmentation" with fewer numbers of projects focused on the most important business value. Some of us can manage more than one or two significant projects in our life, but we begin to lose connection with our teams when we have to focus on three or more projects. The reality is that most of my students report working on seven or more projects at any one time; that's insanity, and the role of the ET is to reduce that insanity by focusing everyone's attention on the projects with the most important technical and business value.
- **Make sure there is a clear definition for decision making (empowerment).** Bill Gore, the founder of W.L. Gore & Associates, creator of GORE-TEX, and namesake for the Gore School of Business at Westminster College, believed in allowing his teams to have the authority to make decisions as long as the boundaries for those decisions were set in advance. To make that point, he used a metaphor of a boat sitting in the water, where if the crew for some reason needed to drill a hole in the side, they were completely within their power to drill that hole anywhere above the waterline. However, if the hole needed to be drilled below the waterline, then they needed to escalate that decision appropriately. The challenge for the ET is to ensure the "waterline" is defined in advance and as clearly as possible.
- **Address issues related to integration points.** Some organizations need to coordinate how they will integrate completed functionality between multiple projects and/or between software and product development initiatives. The good news is that if managers will sit down in advance and discuss their plans, they can generally find easier ways to adapt their development cycles using agile and lean practices to prioritize, develop, and deliver specific functionality than they might be able to do with more traditional waterfall techniques. The ET needs to ensure this type of pre-planning occurs.
- **Assess and resolve technical debt.** (Note that this is very important; see discussion above.) The main job of the ET is to make the business case to reduce the organization's technical debt and establish appropriate development and testing protocols to ensure the debt is minimized over time.
- **Address issues related to release planning.** The ET needs to ensure all the nondevelopment tasks, such as sufficient documentation, training, and end-user communications, are transformed into iterative cycles and included in the development team's planning processes. For example, technical writers who are documenting an application for future training classes can develop those materials iteratively so that their writing follows the process of development. In so doing, they can get advice from developers through an ongoing relationship instead of at the end when everyone is ready to move on and ignore the need for good training

documentation. The ET's role is to ensure this coordination occurs and that everyone gives enough time to complete the development and other release planning details as needed.

- **Create a reasonable prioritization process.** One of the most surprising issues for me is that many of the top-level executives I work with want to “sweat the small stuff.” In fact, many firms with entrepreneurial leaders want to know how they are going to get their pet projects interjected into the development queue even after they have experienced great success by focusing on accomplishing the highest business and technical value first (going so far as to pay an additional bounty to developers who will work on their initiatives). The role of the ET is to establish a clear prioritization process and ensure appropriate governance to ensure those priorities are aligned, from corporate strategy/vision through to individual tasks, and that everyone, including those at the top, follow those priorities.
- **Coordinate sales' commitments with development capacity.** I'm never surprised to learn that the sales department has oversold the IT department's ability to deliver even in the most productive agile environments. Really good salespeople will sell what the customer wants and look for solutions to fit the sale later. The key, and the role of the ET, is to ensure a good communication pathway between the sales and IT departments so that neither surprises the other unnecessarily.
- **Make meetings purposeful and productive.** In most cases, I challenge ETs to post the announcement I introduced earlier in this report that says: “If the meeting you are about to attend has no stated *purpose*, please return to something that does.” Likewise, the ET should monitor meeting feedback from participants describing what worked well and what could be improved and consider providing some form of facilitation training to project managers to meet those expectations.
- **Develop a rapid issue resolution process.** If the team cannot resolve some of the issues that emerge, that is probably acting as roadblocks to the team's success, so the ET needs to provide a rapid process for issue escalation and resolution. Generally, I suggest a small group of executives be on call at all times to meet within 24 hours of an issue being elevated, but that would mean the team has exhausted all other possible solution sources.

- **Address other topics as they emerge.** This is a catch-all subject to ensure the ET is always looking for issues that are getting in the way of a great agile experience; for example, how to resolve fixed contracts with vendors or suppliers or how to coordinate with offshore resources (see Martin Fowler's recommendations for more on this⁵). Most of these issues will come from departments experiencing agile for the first time and wondering what they need to be doing differently.

Firms that take on agile transformation initiatives have two very different types of leaders involved in the process.

Apparent Truth #5: Agile Needs Supportive and Determined Leaders for Long-Term Success

In my experience, firms that take on agile transformation initiatives have two very different types of leaders involved in the process. The first is an executive sponsor who has the resources and patience to stay the course while others around that individual get used to what it takes to be agile. This person's main job is to help people discover their personal reasons for a more collaborative environment, to allow them to vent when the uncertainty and chaos becomes uncomfortable, and to model the facilitative yet determined behaviors the leader wants others to enact.

The second leader is like a chief of staff for the agile implementation process. This person leads the enablement team, champions the cause when others are ready to throw in the towel, and ensures the major organizational issues are resolved and/or brings in the executive sponsor as necessary.

While these can be the same person, the risk of failure goes up if the organization depends on one individual to serve both roles and that person leaves, whether by promotion, retirement, severance, or another reason. In a very extreme case, a large organization tried applying agile practices to a very significant enterprise-wide initiative. While it took some time to get the work processes aligned between the product and development communities and to establish appropriate development and testing protocols, the overall effort was moving forward very well (measured by the continually increasing velocity of completed functionality being produced), until, during one week, both the executive sponsor and the champion unexpectedly resigned from the company.

Almost instantly, the agile detractors started to affect the process by putting in more controls and process steps, so that it began to look more and more like their traditional development lifecycle.

Recommendation

Make sure leadership is shared between at least two people covering the executive sponsor and champion roles. If possible, make sure highly influential people are invited to participate in, and share, those leadership roles through the ET.

Within three iterations, the top executives became incensed when they found out a PM was directing the firm's prioritization process.

Apparent Truth #6: Scrum Is a Great Agile Management Methodology (But Don't Forget the Technological and Product-Focused Practices)

Scrum, as a management methodology, is elegant in its design, requiring only three roles (i.e., product owner, ScrumMaster, and self-organized team), three ceremonies (sprint/iteration planning, daily Scrum/debrief, and sprint review meetings), and three artifacts (product and sprint backlogs and the burn-down chart) — just-enough practical advice so agile teams do not overcomplicate the development lifecycle with too much ceremony and documentation.

However, because it does not include any direction on development, testing, or even coordination in the product community, it is normally seen as a methodology to manage software development initiatives versus a software development methodology. For example, it assumes such matters as sufficient architectural design and appropriate technical environments and protocols for development, testing, and production are in place. Similarly, it presumes someone or some group is managing the customer/product community so that the product owner can represent a unified and prioritized description of the business need.

In many cases, firms choose to apply XP protocols within a Scrum management framework to cover both the technical and management issues. However, this still leaves out the issue of coordinating the needs and interests of the product or customer/end-user community. From my experience, this is where agile, especially in firms that choose to implement Scrum only, starts to run headlong into organizational issues, especially if

those organizations have multiple business/functional units involved in setting the firm's business priorities. In fact, in the most severe case I've seen, a project manager (PM) was "knighted" as the new product owner, with all the rights to make business priorities for the internal product community. Within three iterations, the top executives became incensed when they found out a PM was directing the firm's prioritization process. Unfortunately, they didn't simply express their outrage; they demanded this PM be fired because he kept saying, "Only the product owner gets to make priorities; in addition, you're a chicken who has no voice in this process."

Recommendation

If you decide to apply Scrum as your management methodology, please make sure you incorporate appropriate development and testing protocols, such as the ones you'll find in TDD, XP, and Industrial XP (please refer to the section on technical debt above). Likewise, make sure you spend time developing a collaborative community of customers/end users, so the product owner can represent their joint views.

CASE STUDY

While the following case study is real, the firm's name and the names of the people involved have been left out to protect those involved.⁶

Background

Firm X, an international staffing agency focused on marketing and creative services, needed an easier way to translate client, talent, and internal needs into useful functionality in its enterprise Web applications. Bugged down in endless meetings about priorities, scope creep, and resource allocations, it turned to agile for a simpler and more effective way of developing, updating, and delivering solutions the company needed (yesterday). It decided to create several agile teams, each dedicated to one of the company's business constituencies; for example, individual teams were dedicated to serving the needs of the talent (the marketing and creative professionals), the client (the firms looking for the talent), the back office (the internal group managing payroll, benefits, etc.), and operations (internal users, such as agents, recruiters, and sales managers).

The company also created a team focused exclusively on the internal development infrastructure to maximize the productivity of the other streams. Each of these teams operates in an ongoing stream of iterative development, prioritizing and adapting the capabilities

deemed most valuable to the business in a given iteration. The end result is a company constantly changing, improving, and looking for the next highest business value it can create.

Assessment

Firm X is a fast-growing company headquartered in Boston, Massachusetts, USA, and like many other companies, needs to quickly adapt to the changing business landscape. In the past, Firm X's enterprise information services group (EIS) used a traditional prioritization process in which all of the company's projects competed to be the next in line, followed by a normal waterfall development cycle. An agile readiness assessment clarified the firm's three greatest challenges:

1. "We need to produce business value instead of just fixing or building something someone wants."
2. "Because of the push to release something on time, we have a tendency to release without complete testing."
3. "Though we spend a substantial amount of time planning (four to six weeks), after six to eight months of development, the business still wonders, "What is this"? We are not delivering what it needs, but instead what used to be needed last year."

Based on this poor experience, Firm X recognized the need to get out of packaging the company's development priorities into large projects (whether they were agile or not) and do something much more transformative. So instead of just instituting an agile project management approach for delivery, Firm X decided to set up ongoing streams to identify and prioritize the most important business value as it emerges.

The Firm's Agile Approach: A "Stream" Metaphor

Firm X uses "stream" as a metaphor describing a process in which there is no real beginning or end (except for a given iteration cycle), but there is a consistent flow of value being built, tested, and released. The capabilities to be delivered in an iteration are not determined until that iteration begins. Each iteration begins with a planning session in which selections are made from a large "unsequenced capability" deck according to each capability's business value. The stream chooses only enough of the most valuable capabilities to fill the development time in this next iteration. The rest are left for prioritization in subsequent iterations. Thus, there is no defined "project" the stream must complete; rather, it must simply deliver the most valuable capabilities for the overall

business at any given time and declare success only as business goals are met.

Four business streams now operate in the following business areas, as introduced above: talent, client, back office, and operations.

Working and Coordination Agreements

Each team is made up of a business owner, user advocate, project manager, and one or more developers. The business owner represents the strategic needs of Firm X and prioritizes features based on their business value. The user advocate represents the tactical needs of the users. The project manager is a facilitator and consensus builder who ensures the game is played well. Finally, the developers are the ones actually creating the business and technical solutions. The business owner, user advocate, and project manager are all jointly responsible for acceptance testing of new functionality. Roughly speaking, the business owner decides what should be built, the user advocate decides how it should work, the developers build it, and the project manager ensures it is done well.

Due to the distributed organization at Firm X, most of the teams are not entirely colocated. The developer(s) and project managers are in the same space, but the user advocate and business owners are often in other cities in North America. While this caused some delays in the early days of implementing the stream approach, several technical tools have helped the team record new functionality, prioritize and estimate appropriately, and track progress. A product/service called CardMeeting has proved effective as a shared virtual desktop on which distributed users can create, modify, and move capability/story cards in real time. This, along with voice conferencing, e-mail, and 37signals' Basecamp product help keep the teams in sync.

Technical Debt

One key to the success of this stream approach was the creation of an infrastructure stream focused on the company's development infrastructure. Instead of asking the business streams to take on the significant investment needed to automate testing, clean up the existing "technical debt," and rationalize the purely technical features, Firm X chose to create this small team focused on nothing but continually improving the company's technical infrastructure.

The customers of this stream are the other development teams working on the business needs. First among the priorities for this stream was the development of an

automated testing infrastructure. Reducing the regression testing from six to eight weeks in the waterfall approach to one week to match the agile iteration cycles proved crucial. Additionally, code refactoring and API development simplified the code base and enabled quicker, easier, and safer development in the other streams. With much of the work done, this stream has now been suspended and the members reallocated to other, business-focused streams. Firm X has recognized the probability that the infrastructure stream may be resurrected if the need arises.

Back-office owner: "I can weigh the needs of the organization and choose what's most important to the business as a whole. The bureaucracy is gone."

Best Practices from Lessons Learned

After a year and a half of experience practicing and adapting techniques to its growing, distributed development community, Firm X has quite a few best practices and lessons to share.

The Kickoff

As each new stream is created, there is a kickoff series to ensure all the team players and the executive sponsor are aligned. In this series, the team crafts its mission statement and business objectives, the working agreements necessary to be agile and remote, an initial catalog of capabilities to consider, and a high-level iteration plan to guide the team through the first few development iterations. This ensures each team learns from the previous stream experiences and has a baseline of operational norms with which to start.

Executive Support/Involvement

Each stream has some level of executive support, if not direct operational involvement. The "business owner" is actually someone who owns the outcome of the stream, not just a representative or proxy. This means, the team has access to strategic corporate direction and is confident its priorities are appropriate. It also ensures the highest business value is actually being produced. For example, Firm X's VP of finance (back-office stream) and VP of recruiting (talent stream) play the role of business owner and have direct control over the capabilities developed. The back-office owner says: "This new process has changed the way I look at development

work. Instead of shouting over every other voice to get 'my stuff' done, I can weigh the needs of the organization and choose what's most important to the business as a whole. The bureaucracy is gone."

Enablement Team

A team of three top executives representing technology, external clients, and internal processes agreed to take on major issues as agile was introduced and implemented. The team's job was to act as a "release valve" and help resolve any emerging issue the teams couldn't handle internally. This team provides a sense of comfort that executives truly support their efforts. Within the first week, the team resolved two significant procedural issues.

Infrastructure and Testing

A combination of regularly executed automated tests and manually executed regression tests at the end of each iteration supplement the user acceptance testing performed within each stream. Manual regression is confined to the weeklong release iteration that follows each four-week development iteration. This allows each stream to contribute to the overall system stability while focusing entirely on new functionality during the development iterations.

Stream Coordination

The business owners of each stream gather once a week in a "sync meeting" to discuss the major issues/topics affecting them. As this experiment goes forward, the teams are resolving higher-level strategic issues and ensuring stories are coordinated across streams to catch the dependencies and similarities between streams. Additionally, the streams are coordinating with other operational leaders from outside the development teams to manage the issues that emerge from across the business. Thus, the development process can reduce (instead of exacerbate) the impact on these other business units including engineering and operations, learning services, and marketing and back-office payroll.

Other Business Processes

As new business stream teams are established, those business units are beginning to apply agile principles in their work processes as well. As an example, the group responsible for training within Firm X recently went through an agile overview course so it could better serve the ongoing streams and so it could begin applying many of the iterative, collaborative, and productive techniques within agile to such issues as content development, customer involvement, and communications.

Colocation

These teams have discovered that as long as the user advocates and business owners are involved with daily stand-ups, resolving issues, story definition, prioritization, reflection, and iteration planning, then their physical presence is not a critical success factor. This does mean, however, that each team member is competent in using technical collaborative tools, such as CardMeeting, voice conferencing, e-mail, and 37signals' Basecamp. To better support these distributed teams, the development of a communication tool specifically designed to complement Firm X's agile approach is under consideration as well.

Infrastructure Team

The infrastructure team focuses on purely technical issues that sometimes get overlooked when developing solely "customer" functionality. It was able to increase the use of automation and refactor the code base to significantly reduce the time spent on regression testing, thereby enabling the other development streams to build solutions more efficiently and effectively.

Personnel Allocation

As with the infrastructure stream, Firm X has found it can more easily move people around from stream to stream, starting up new teams with people who have experience in a stream and decommissioning a stream when its business objectives have been achieved or it no longer creates business value.

Developer Engagement

In addition to enhancing the planning and prioritization process, this stream approach improves the developers' experience as well. Participating in the planning process with the actual business owners not only gives the developers a say in what is delivered and how, but also provides insight into the business goals of a capability. This greatly enhances their ability to deliver a valuable solution. Ideas emerge while developing that lead to an improved solution, which would never be possible if the developers were merely handed a functional specification and did not understand the reasoning behind a capability's selection. It also leads to increased satisfaction and pride in their work. As one developer explained: "I feel good when I build something that people find useful. Code is fun in itself, but without the context I get from the planning sessions, it can often become a 'just get it done' exercise. Now, I'm constantly working to build something that works better, not just something that's built better."

Case Study Conclusions

This stream approach seems much more appropriate for developing ongoing business solutions, especially within a larger company where there are multiple functional expectations. It focuses the prioritization process on the real business owners, which thereby ensures the highest value is developed first. Likewise, it ensures a greater transparency into what each of the development teams is doing and their results. By eschewing the notion of a "project," it allows the greatest flexibility and responsiveness to ever-changing business needs.

It has allowed the teams to focus their debates on things of real importance instead of nitpicking procedural questions that had them bogged down in their previous processes.

The future challenges will probably include:

- Ensuring the cross-stream coordination and communication doesn't slow down the system
- Developing an overarching strategy across streams to move the company forward appropriately
- Looking at other business process flows to ensure they are as agile as the development streams

The fact that development teams are now fully integrated into the business processes within Firm X and that executive leadership is involved has allowed these "streams" the opportunity to finally deliver quality business functionality the end user really wants and needs. More important, it has allowed the teams to focus their debates on things of real importance instead of nitpicking procedural questions that had them bogged down in their previous processes.

SUMMARY

The intent of this *Executive Report* is to bring forward a few apparent truths to consider in an agile implementation effort, to provoke action where needed, and to offer recommendations where appropriate so that those of us in the "agile movement" might debate these topics more clearly and focus the corporate attention where it needs to be.

While I know it is presumptuous to speak about "truths" in any situation, I am particularly cautious to

make such claims inside the agile community because of the passions they may unleash. However, I do think it is time the community expands its impact from the leadership, managerial, and development practices associated with creating software solutions into creating more agile and adaptive organizations where software is only a service rendered in pursuit of a business objective. To that end in this report, I called out the following six apparent truths to help organizations understand the implications of “going” agile:

1. There is no single, agreed-upon definition of agile.
2. Agile is a quality initiative.
3. Companies need to invest in reducing their technical debt.
4. Your organization will change if you adopt agile.
5. Agile needs supportive and determined leaders for long-term success.
6. Scrum is a great agile management methodology (but don't forget the technological and product-focused practices).

In response to these truths, I offered several recommendations, including my own list of “capabilities” that firms should possess if they want to be considered agile, reasons to invest in improving the firm's quality outcomes (including the reason firms must consistently work on reducing their technical debt), and suggestions for establishing an agile enablement team. In addition, I offered firms that are using or thinking of using the Scrum methodology a recommendation to ensure they incorporate the technical and customer communities in this wonderful management framework. I also presented a case study about a firm that applied these organizational practices.

Good luck in your agile journey.

DEDICATION

This report is dedicated to the memory of Jim Thrash, a friend, neighbor, and colleague in the USDA Forest Service.

ENDNOTES

¹Agile Manifesto (<http://agilemanifesto.org/principles.html>); Declaration of Interdependence (<http://pmdoi.org>).

²Highsmith, Jim. *Agile Software Development Ecosystems*. Addison-Wesley Professional, 2002.

³Highsmith. See 2.

⁴See 1.

⁵Fowler, Martin. “Using an Agile Software Process with Offshore Development,” 18 July 2006 (<http://martinfowler.com/articles/agileOffshore.html>).

⁶If you are interested in talking with this company directly, please contact the author.

ABOUT THE AUTHOR

David Spann is a Senior Consultant with Cutter Consortium's Agile Product & Project Management practice. He focuses on strategic planning, team building, executive coaching, and training for organizations wanting to be more agile and adaptive. Mr. Spann helped host the first *Agile Software Development* conference at Westminster College, Salt Lake City, Utah, USA, in 2002; cohosted the *Agile Executive Summit* in 2003, 2004, and 2005; teaches the only MBA course on adaptive project management in the US; is a Certified Professional Facilitator; and in past careers was a District Ranger in the USDA Forest Service as well as an MBA Director at Westminster College. He can be reached at dspann@cutter.com.

Index

●●● CUTTER CONSORTIUM

Agile Product & Project Management Advisory Service Published Issues

Upcoming Topics

- **Program Management**
BY ALEX RODRIGUES
- **Scaling Up Agile Adoption
by Scaling Down**
BY AMR ELSSAMADISY

This index includes Agile Product & Project Management *Executive Reports* and *Executive Updates* that have been recently published, plus upcoming *Executive Report* topics. Reports that have already been published are available electronically in the Agile Product & Project Management Resource Center. The Resource Center includes the entire Agile Product & Project Management advisory service archives plus additional articles, Webinars, and podcasts delivered by Cutter Consortium Senior Consultants on the topic of agile.

For information on beginning a subscription or upgrading your current subscription to include access to Cutter's Resource Centers, contact your account representative directly or call +1 781 648 8700 or send e-mail to sales@cutter.com.

Agile Product
& Project
Management

Executive Reports

- Vol. 9, No. 12 Breaking the Facade of Truth: An Introspective View into and a Case Study About the "Apparent Truths" of Agile BY DAVID SPANN
- Vol. 9, No. 11 Scrum Today BY BRIAN J. DOOLEY
- Vol. 9, No. 10 Are You a Cook or a Chef? Succeeding in the Contemporary World of Project Management BY ROBERT K. WYSOCKI
- Vol. 9, No. 9 How Agile Projects Measure Up and What It Means to You BY MICHAEL MAH WITH CONTRIBUTION BY MIKE LUNT
- Vol. 9, No. 8 Agile Software Package Implementations BY SAM BAYER
- Vol. 9, No. 7 Refactoring in the Context of Enterprise Architecture BY SEBASTIAN KONKOL
- Vol. 9, No. 6 The Four Pillars of Agile Adoption BY NANCY VAN SCHOENDERWOERT
- Vol. 9, No. 5 Social Project Management BY DAVID COLEMAN
- Vol. 9, No. 4 Agile Adoption for Organizational Change: Improving Time to Market BY AMR ELSSAMADISY
- Vol. 9, No. 3 Moving the Herd: Facilitating Multiparty Project Teams Toward Common Goals BY MOSHE COHEN
- Vol. 9, No. 2 Is Design Still Dead? BY KEN ORR
- Vol. 9, No. 1 Transitioning to Agile Project Management: A Roadmap for the Perplexed BY SANJIV AUGUSTINE AND ARLEN BANKSTON
- Vol. 8, No. 12 Negotiating Resources, Deliverables, and Deadlines Within the Global Organization BY MOSHE COHEN

Executive Updates

- Vol. 9, No. 23 To Release No More or To "Release" Always: Part II — Toward a New Business Design for Software BY ISRAEL GAT
- Vol. 9, No. 22 Managing the Project Portfolio: An Agile/Lean Approach BY JOHANNA ROTHMAN
- Vol. 9, No. 21 To Release No More or To "Release" Always: Part I — The Myth BY ISRAEL GAT
- Vol. 9, No. 20 A Fresh Look at Software Project Estimation: Part II — The Logic of Special Agent Gibbs BY E.M. BENNATAN
- Vol. 9, No. 19 Managing Offshore Development: From Requirements Analysis to Customer Support BY STACEY BERLOW
- Vol. 9, No. 18 A Fresh Look at Software Project Estimation: Part I — Promises Impossible to Keep BY E.M. BENNATAN
- Vol. 9, No. 17 Stormy Skies: Forecasting in the Face of Uncertainty BY DAVE ROONEY
- Vol. 9, No. 16 Managing Offshore Development: Establishing Software Development Processes BY STACEY BERLOW
- Vol. 9, No. 15 Insecure Code: An Agile Look at the Debian Debacle BY LAURENT BOSSAVIT
- Vol. 9, No. 14 Software Project Planning: Part III — Divide and Conquer BY E.M. BENNATAN
- Vol. 9, No. 13 A Practical Guide to Crafting an Agile Adoption Strategy BY AMR ELSSAMADISY
- Vol. 9, No. 12 Flying the Friendly Skies: Navigating Agile Projects BY DAVE ROONEY
- Vol. 9, No. 11 Software Project Planning: Part II — Back to 1978 BY E.M. BENNATAN
- Vol. 9, No. 10 Software Project Planning: Part I — What Can We Learn from *Star Trek*? BY E.M. BENNATAN

SUMMIT 2009



Interact with the Experts in IT

Now, More Than Ever ...

Summit 2009

4–6 May 2009 | Cambridge, MA, USA

Why Attend Summit 2009?

IT is being challenged to scrutinize every expenditure and make sure that every dollar is invested — not just spent — on high ROI efforts. *Summit 2009* is one of those high-value items. Join us at *Summit 2009* and explore the issues and strategies you should be considering to ensure you're maximizing today's return and keeping ahead of the competition.

Summit 2009: Unlike Any Other IT Conference

You'll get advice that is insightful and forthcoming. You'll have a chance to voice your opinion and hear the opinions of international experts and of other IT professionals. You'll debate with, and listen to the debate among, some of the most experienced and clear-thinking business technologists today. And you'll return home with ideas to implement right away. Our only agenda is to support you, to give you a forum for thinking about your business world, outside of the box. Don't miss this opportunity.



The 5 Essential Habits of Appropriately Paranoid Business Technology Strategists

Keynote by Steve Andriole



Information Security: Zeroing in on a Moving Target

Keynote by Mark Seiden



Case Study: How to Avoid Getting Flattened by a Runaway Project

Taught by Rob Austin



Don't Downsize. Rightplace Instead.

Keynote by Vince Kellen



Where Do We Go from Here?

Keynote by Tom DeMarco



IT Managers Forum

Forum with Warren McFarlan



Collaboration for the Enterprise

Seminar with Mike Rosen and John Tibbetts

●●● Plus additional sessions and roundtables ...



Visit www.cutter.com/summit.html for an updated agenda

SUMMIT 2009



Interact with the Experts in IT

Jump ahead of the competition. Reserve your seat for Summit 2009 today!

"The Summit stretched my thinking in some new directions."

— Martin Malley
AVP, Information Systems
Union Pacific Railroad

"Engaging in a way that other conferences don't accomplish."

— David Smalley
IT Group Manager
Progressive

"Learned some new ways of looking at or managing things."

— David Sum
Director Product Platforms
Custom House

Register Today

To ensure a lively and interactive event, attendance for *Summit 2009* will be limited. Register early and guarantee a space for yourself at this exclusive conference.

Return this form by fax to +1 781 648 8707, or mail it to Summit 2009, Cutter Consortium, 37 Broadway, Suite 1, Arlington, MA 02474-5552, USA. Checks should be made payable to Cutter Consortium. To register by phone, call +1 781 648 8700. Or visit www.cutter.com/summit.html to register online.

Registrants Total

Summit 2009 Conference (4-6 May 2009)

- US \$1,995 each
(US \$1,795 if registered before 31 January 2009
— a \$200 savings)

_____ \$ _____

If you would like to register your team of 4 or more for Summit 2009, please contact your account representative for multiple-attendee discounts.

Name _____ Title _____

Organization _____ Department _____

Address _____ Mail Stop/Suite # _____

City _____ State/Province _____

ZIP/Postal Code _____ Country _____

Telephone _____ E-Mail _____

- Check enclosed (payable to Cutter Consortium).
 Please invoice my company.
 Charge MasterCard, Visa, AmEx, or Diners Club. (Charge will appear as Cutter Consortium.)

Account Number _____ Expiration Date _____

Name on Credit Card _____ Signature _____

Registrant #1 Name _____ Title _____

Registrant #2 Name _____ Title _____

Registrant #3 Name _____ Title _____

Registrant #4 Name _____ Title _____

Registrations for *Summit 2009* are transferable. If you find that you cannot attend, you may transfer the registration to another member of your organization at any time prior to the first day of the event. Cancellations after 31 March 2009 are nonrefundable. In the unlikely event that Cutter Consortium cancels the conference, Cutter Consortium is not responsible for any airfare, hotel or other costs incurred by registrants. Speakers subject to change without notice.

Agile Product & Project Management Practice

Cutter Consortium's Agile Product & Project Management practice provides you with information and guidance — from experienced, hands-on Senior Consultants — to help you transition (or make the decision to transition) to agile methods. Led by Practice Director Jim Highsmith, Cutter's team of experts focuses on agile principles and traits — delivering customer value, embracing change, reflection, adaptation, etc. — to help you shorten your product development schedules and increase the quality of your resultant products. Cutting-edge ideas on collaboration, governance, and measurement/metrics are united with agile practices, such as iterative development, test-first design, project chartering, team collocation, onsite customers, sustainable work schedules, and others, to help your organization innovate and ultimately deliver high return on investment.

Through the subscription-based publications and the consulting, mentoring, and training the Agile Product & Project Management Practice offers, clients get insight into Agile methodologies, including Adaptive Software Development, Extreme Programming, Dynamic Systems Development Method, and Lean Development; the peopleware issues of managing high-profile projects; advice on how to elicit adequate requirements and managing changing requirements; productivity benchmarking; the conflict that inevitably arises within high-visibility initiatives; issues associated with globally disbursed software teams; and more.

Products and Services Available from the Agile Product & Project Management Practice

- The Agile Product & Project Management Advisory Service
- Consulting
- Inhouse Workshops
- Mentoring
- Research Reports

Other Cutter Consortium Practices

Cutter Consortium aligns its products and services into the nine practice areas below. Each of these practices includes a subscription-based periodical service, plus consulting and training services.

- Agile Product & Project Management
- Business Intelligence
- Business-IT Strategies
- Business Technology Trends & Impacts
- Enterprise Architecture
- Innovation & Enterprise Agility
- IT Management
- Measurement & Benchmarking Strategies
- Enterprise Risk Management & Governance
- Social Networking
- Sourcing & Vendor Relationships

Senior Consultant Team

The Cutter Consortium Agile Product & Project Management Senior Consultant Team includes many of the trailblazers in the project management/peopleware field, from those who've written the textbooks that continue to crystallize the issues of hiring, retaining, and motivating software professionals, to those who've developed today's hottest Agile methodologies. You'll get sound advice and cutting-edge tips, as well as case studies and data analysis from best-in-class experts. This brain trust includes:

- Jim Highsmith, Director
- Sanjiv Augustine
- Christopher M. Avery
- Paul G. Bassett
- Sam Bayer
- Kent Beck
- E.M. Bennatan
- Tom Bragg
- David R. Caruso
- Robert N. Charette
- Alistair Cockburn
- Jens Coldewey
- David Coleman
- Ken Collier
- Ward Cunningham
- Rachel Davies
- Tom DeMarco
- Lance Dublin
- Khaled El Emam
- Sid Henkin
- David Hussman
- Ron Jeffries
- Joshua Kerievsky
- Bartosz Kiepuszewski
- Brian Lawrence
- Mark Levison
- Tim Lister
- Michael Mah
- Siobhán O'Mahony
- Ken Orr
- Roger Pressman
- James Robertson
- Suzanne Robertson
- Alexandre Rodrigues
- Dave Rooney
- Johanna Rothman
- David Spann
- Rob Thomsett
- John Tibbetts
- Jim Watson
- Robert K. Wysocki
- Richard Zultner