An Enterprise Strategy for Introducing Agile

By Kane Mar

When Agile software development practices are introduced into an organization, they often spread faster than they can be controlled.

Without a clearly defined standard to guide their implementation, numerous Agile management approaches can emerge simultaneously—sometimes in direct conflict with one another. That's not necessarily a bad thing, but traditional, top-down management structures typically experience some growing pains as they transition to the more egalitarian Agile approach. So how does an organization integrate this new Agile framework without exposing itself to undue risk?

Over the last two years, I've worked closely with several clients as they have rolled out Agile practices throughout their enterprises. In nearly every instance, I observed that these organizations moved through extremely similar stages and, consequently, encountered very similar problems. Anticipating these problems and minimizing their impact is possible, but only if there is a clear overall strategy in place. Put another way, if you were driving through unfamiliar territory, you'd rely on a road map to ensure you head in the right direction. And while no two organizations follow the exact same path, knowing what to expect can help organizations remain oriented. Just think of this overview of the trends and patterns I observed as a compass.

A TRIP DOWN THE PATH

At the very start, we have a non-Agile enterprise: an organization where Agile practices have yet to be introduced and the prevailing attitude is that Agile methods will not improve upon the status quo. Often, this skepticism is based on little or no understanding of Agile methods or the benefits they can provide.

Agile methods can be introduced into an organization in a number of ways. Most commonly, developers begin implementing a handful of practices into their work after becoming acquainted with the framework through existing literature. Piquing the interest of other developers and management, ideas gradually spread. As momentum builds behind the new methodology, the organization may proceed with experimentation on one or several small projects. This phase of Agile adoption is usually limited in both scope and budget. It's an intermediate step that allows enterprises to safely experiment with new ideas and techniques, while limiting risks.

Naturally, these problems can create clashes between proponents of Agile integration and those outside the team, who fear their positions are threatened.

These experimental projects are, in essence, trials, yielding both valuable and hard lessons. If the enterprise's experience with the new practices proved to be, more or less, positive, it will often accept that there are considerable advantages to Agile management and initiate a more systematic rollout. This Pilot phase will be more formally structured, but remains limited in its scope. Seeded through the group or department, these projects up the ante on the experimental projects in many ways, including budget, team members, and visibility to both senior management and external clients.

One of the strengths of the Agile methodology is its ability to bring impediments to the fore. The nature of these impediments is constantly evolving, but, at least initially, they are often issues that directly affect the development team: poor tools, inefficient build, and source code management issues. Once project teams become increasingly experienced, issues pertaining to quality crop up, which typically include test-driven development, continuous integration, refactoring, etc. At a mature stage of Agile adoption, issues on the organizational level emerge, such as project staffing, individual compensation, and promotion (see the following page). Naturally, these problems can

create clashes between proponents of Agile integration and those outside the team, who fear their positions are threatened.
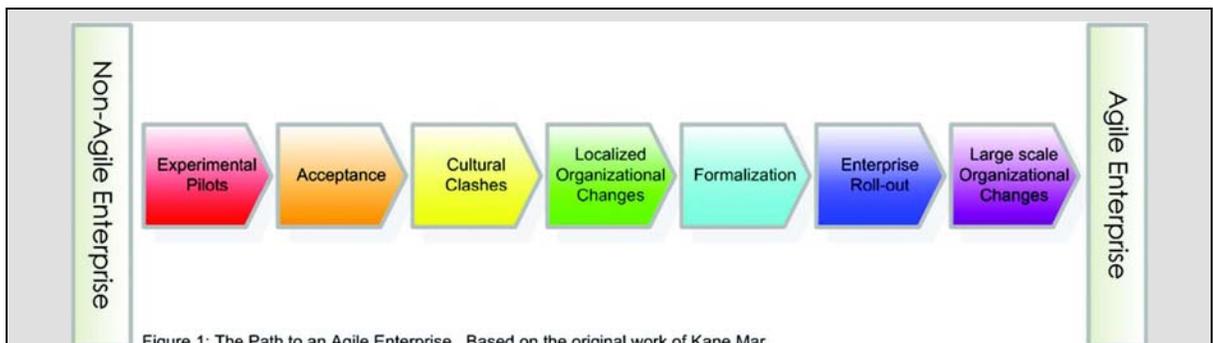

Figure 1: The Path to an Agile Enterprise. Based on the original work of Kane Mar.

The Path to an Agile Enterprise

The diagram above summarizes the path that organizations follow when transitioning from a non-Agile environment to an Agile one. [The different stages in this diagram are arbitrary. By that, I mean that there was no standard measure by which companies were judged when moving from one state to another. As a result, this diagram is certainly up for discussion and I'd be interested in your thoughts and comments.]

Mapping integration progress from left to right, the diagram is broken down into stages, each of which presents its own problems and challenges. An organization can fail at any point along the path. An enterprise's ability to avoid bumps in the road depends on its willingness to meet the issues that arise head-on and its openness to modifying its culture. I'll return to these two survival skills later in the paper, but, for now, I'd like to stay on the path, as it were.

For enterprises that have witnessed Agile's benefits and garnered support from senior management, many of these issues will be addressed. The changes deemed most urgent to the success of the Agile teams will be dealt with first, including the use of a simpler build process, the introduction of continuous integration, or the reduction of mandatory team meetings. These initial changes to the organization are typically limited in scale (confined to a particular group or department) and perceived risk.

As the success of the Agile teams leads to a growing sense of acceptance throughout the enterprise, more teams will eagerly implement the same methodologies in their own work. This accelerated spread of the framework typically results in a sudden increase of not only Agile projects, but in the many ways teams integrate these methods. Invariably, teams adopt different practices or implement similar ones, but use them in slightly different ways. For example, one team might measure progress in two-week iterations, while another might use four-week iterations. Likewise, some teams will hold daily team meetings, some will not. Different teams will estimate story points in different ways. With little or no enterprise-wide consensus over how Agile methods will be instituted, the next step in the process is formalization. This effort is typically led by a project management office, which will determine how Agile works "within the context of the enterprise" by defining practices such as terminology, length of iterations, reporting, and metrics.

The enterprises that will most successfully navigate the path to Agile integration will be those that can manage constant change within an organization; that remains closely attuned to its customers' needs; that reacts to changing business conditions at incremental cost (rather than exponential cost); and that employs individuals who are committed to constant education and innovation.

Once Agile practices have been standardized, the enterprise will follow formalization with a large-scale rollout, implementing Agile methods across several departments of the organization. This is the real moment of truth for Agile in the enterprise, in which more ambitious projects, such as Scrums-of-Scrums, will be undertaken. By this stage, Agile implementation is highly visible to senior management and often involves increased corporate risk—even the potential failure of a large project.

This transition will inevitably place great strain on the enterprise. During the course of these large-scale projects, not only will the usual local problems crop up (tools, build times, quality issues, etc.), but also a new set of enterprise-wide issues, including compensation, promotions, and roles and responsibilities. Because these new issues will challenge the organization's culture and require its employees to alter their day-to-day practices, they will be the most difficult to solve. Moreover, some individuals will perceive that their position and authority within the enterprise are jeopardized. Therefore, any risk (whether perceived or real) related to the introduction of Agile development must be resolved promptly by senior management.

The enterprises that will most successfully navigate the path to Agile integration will be those that can manage constant change within an organization; that remain closely attuned to its customers' needs; that react to changing business conditions at incremental cost (rather than exponential cost); and that employ individuals who are committed to constant education and innovation.

A PLAN OF ACTION

With Agile's recent rise in popularity, organizational integration of its methodologies is already becoming more common. The question remains: How can enterprises make the transition as smooth and secure as possible? The two best-known approaches are Top-Down, in which senior management initiate the introduction of Agile, and Bottom-Up, in which developers and testers spur the integration. The truth is neither approach is flawless. The most effective strategies I've seen have combined aspects of both. Because Agile software development practices force such large cultural ramifications for an enterprise, coordinated change needs to occur throughout the organization — not just at the top or the bottom. The most seamless transition will execute a formalized plan that addresses concerns for both those who do the work and leadership who makes the strategy a reality.

According to the *American Heritage Dictionary of the English Language*, strategy is defined as "a plan of action ... intended to accomplish a specific goal." In this case, the specific goal is the complete rollout of an Agile methodology to an entire organization.

For our purposes, I've broken this plan of action into three phases, separated based upon scope of influence ever-increasing outward, like ripples on a pond. In the Pilot phase, only a limited number of individuals are directly affected and the projects are limited in scope and risk. In the Launch phase, though, an entire department (many people and multiple projects) may be affected, but the rollout is still confined to the department. The final phase is the Enterprise Rollout, in which individuals across multiple departments are affected and projects involve significant budgets and risk to the organization.

Each phase is outlined below with a description and a list of activities that are commonly performed. For each of the activities, I've also included a number of questions that need to be addressed. These questions will be helpful in guiding each organization on its own unique path to adoption. You'll notice that no answers are provided. Because every organization is different, its strategies will differ accordingly. Still, the questions merely represent possible routes to success. Some organizations will have to answer many of these questions in order to succeed, while others will only need a few.

Strategy: strat•e•gy n. pl. strat•e•gies

1.1 The science and art of using all the forces of a nation to execute approval plans as effectively as possible during peace and war.

1.2 The science and art of military command as applied to the overall planning and conduct of large-scale combat operations.

2. A plan of action resulting from strategy or intended to accomplish a specific goal. See synonyms at plan.

3. The art or skill of using stratagems in endeavor such as politics and business.

(From The American Heritage® Dictionary of the English Language, Fourth Edition)

PHASE 1. THE PILOT

The Pilot phase is primarily concerned with the immediate rollout of Agile practices to a known team. Along the way, questions arise about how to integrate the Agile framework most seamlessly. How granular should backlog items be? How does one write stories? Where does Quality Assurance fit in? The Pilot phase is an opportunity for an organization to hammer out these details and determine what works best within the context of the enterprise.

The Pilot phase typically lasts between six months and a year. During this time, only a small number of experimental projects (between five and 10) are involved. Generally, these projects share several attributes. Namely, they are limited in budget (and, consequently, duration and staffing), scope, and risk. They also tend to focus on delivering functionality to an internal client, thereby limiting external visibility, and have few, if any, dependencies. What follows is a list of activities that are typically undertaken during this phase, paired with a list of questions to help identify how best to approach them.

Introduce Agile software methodologies to several small teams using coaches:

- Should the coach be CSM-certified?

- Is it possible to certify an existing project manager and then assign him/her to act as ScrumMaster?

- How many teams can a ScrumMaster manage at a time?

- How large should those teams be?

The Launch phase focuses on how best to roll out Agile methodologies to a much wider audience in a consistent manner.

Introduce Agile practices and terminology to the teams:

- What terminology should the team use — Scrum, XP, or some combination of both?

Identify likely cultural issues and organizational impediments:

- Is there some group or individual who feels most threatened by the introduction of Agile methods?

- Is there a Methodology or Software Development Process group?

Identify tool issues:

- Are the tools quick, efficient, and reliable?

- Do they leave the code base in a known state?

- Or are the tools cumbersome and require extensive babysitting?

- Do the tools meet the needs of the team?

- Or is there an alternative solution which better meets their needs?

- Is the choice of tools made by the developers or by some third party that is not responsible for delivering code?

Identify likely IP issues (open source tools, GPL code):

- Does the organization have a fear of GPL code?

- Is there a tendency for the organization to redevelop tools that already exist in the marketplace?

Identify management issues:

- How do functional managers (i.e. QA managers, software analysis managers, etc.) fit into an Agile model?

- Who should the Product Owner be and what should the team do if the Product Owner does not want to engage the team?

Identify and resolve reasons for initial failures:

- What made some of the initial Agile projects successful?

- And how should the failed projects be addressed?

COLLABNET

Physical location and layout:

- How important is collocation to Agile projects?
- Can teams retain their offices and communicate via IM and/or email?
- What about teams in different locations or time zones?

Present Agile to interested parties and senior management:

- Who should know about the benefits that Agile software development can bring?
- How should they be educated — in a series of lectures or by presentations from the team?

Have senior management promote Agile SW.

PHASE 2. THE LAUNCH

The Launch phase focuses on how best to roll out Agile methodologies to a much wider audience in a consistent manner. For this to occur, the organizational understanding of Agile must be clearly defined and codified throughout the enterprise. It's a process of drilling down into how the Agile framework will affect day-to-day operations and answering questions as they arise. How long will iterations be? What tools will teams use? What formats should teams use to report progress?

Projects in this phase are usually much larger than those in the Pilot phase in terms of budget, scope, and risk. A larger budget means these projects will also have a larger number of staff and longer durations. Projects addressed in the Launch phase typically represent a cross-section, but are undertaken within a single department. They commonly address various aspects of the organization's business, including new functionality, software and database maintenance, and reporting.

Once an organization reaches the Enterprise Rollout phase, its attention turns to solving the most difficult aspects of Agile integration.

Again, what follows is a list of activities typically undertaken during this phase, paired with a list of questions to assist in determining what steps to take to achieve them.

Codify the organization's understanding of Agile. This includes establishing:

Usage of common terminology:

- What does a sprint or an iteration mean?
- What is the Scrum equivalent of Iteration 0?

Usage of common metrics:

- What scale should the teams use for estimating story points — a scale of one to 10, a scale of one to five, or something else?
- What is the meaning of "velocity" and should the organization compare velocities between two different teams?

Usage of common tools:

- What source control tool should the teams use?
- Should all the teams be using some form of continuous integration? If so, which tool?
- What about IDEs and code coverage tools?

Usage of common reporting formats:

- Should the teams present their results as Burn Down charts? Is there any value to Gantt charts or Microsoft Project?

Formally establish a coaching model:

- What is the organizational coaching model?
- How should coaches be brought on board and what career path should they follow?

Establish an office layout/collocation policy:

- Are teams collocated? Is there sufficient space to create team rooms?
- Are there expenses involved with collocating teams?

Establish Agile forums within the organization:

- What are the best ways to ensure different teams are communicating their experiences?
- Should this be an informal event or should there be some ceremonial process?

Establish Agile project selection and project scoping (size and cost):

- What projects would be suitable for Agile software development?
- What criteria should be applied to the project selection process?
- How should you use Agile methods to estimate the size and costs of these projects?

Present Agile methodologies to interested parties and senior management.

- Who within the organization would help promote Agile methods?
- Who should be educated about the benefits that Agile methods can bring to the enterprise?

PHASE 3. THE ENTERPRISE ROLLOUT

Once an organization reaches the Enterprise Rollout phase, its attention turns to solving the most difficult aspects of Agile integration. How will teams in different departments communicate on shared projects? How will multiple Scrum teams be organized and managed to complete large-scale projects? How will an Agile framework affect compensation and promotion?

In this phase of adoption, projects are typically large, expensive, and potentially very risky. Therefore they require ample persuasive and political skills to negotiate solutions. Given these factors, this phase is usually the longest of the three. Whereas projects in the previous two phases could be initiated and completed within a six- to nine-month period, the Enterprise Rollout can last as long as one to two years.

The list below includes many activities common to enterprises in this phase of adoption as well as several questions to generate solutions.

The path I've outlined is an idealized version of the process. As such, it cannot account for every challenge an organization will face, but it will give enterprises a map to help them anticipate landmarks along the road to Agile integration.

Encourage internal communication regarding Agile:

- What is the best approach for encouraging internal communication between different Agile teams?
- Is it an Agile forum? Or is an email list sufficient?

Anticipate change and have a plan to evaluate changing circumstances:

- If a new application is getting more traffic than anticipated, how can you exploit that to your best advantage?

Review and align compensation model with Agile teams.

- Is everyone on the team adding value?
- How should project managers who do not facilitate a team (Scrum or Scrum-of-Scrums) be compensated?

- Is an architect who mentors a team more valuable than one who does not?

Review HR and hiring policies:

- Are your existing hiring practices sufficient to find skilled staff that works well in an Agile environment?
- Does the team have any say in who joins or leaves the team?

Establish parameters around very large projects with Agile.

- What qualifies as a large project? At what point should a project be broken down into two or more sub-projects?
- Are there additional financial constraints that larger projects must meet?
- In a large project, who is the Product Owner?
- Should there be a single Product Owner or multiple ones?

Establish parameters around distributed projects with Agile.

- How experienced should the team be? How is communication between teams handled?
- How is it handled between teams and the Product Owner?
- Is the Product Owner a part of the business or the development team?
- Should the entire business be relocated to somewhere more cost-effective?

Establish promotion policies.

- How should successful individuals be promoted?
- Should the promotion model be based on merit, influence, or some combination of both?

Establish a training model for coaches and Agile teams.

- What are the training requirements of Agile teams?
- After doing some initial training, what else should Agile teams know?

Align funding with lines of business.

- Since funding for Agile teams is usually secured by the Product Owner, how does this affect software development groups that have previously used their own sources of funds?
- How will the management structure react to changes in the funding model for these departments?

SUMMARY

By now, you've seen just how complex and challenging an enterprise-wide adoption of Agile methodologies can be. It's a process that is further complicated by an infinite range of variables that influence the outcome. I've started several organizations down this path, but I've never witnessed any two arrive in the exact same place, over the same amount of time.

The path I've outlined is an idealized version of the process. As such, it cannot account for every challenge an organization will face, but it will give enterprises a map to help them anticipate landmarks along the road to Agile integration. It will be an even smoother transition, if organizational leadership has the right mix of experience, persuasive skills, and political collateral to react to the most unexpected crises. Still, for all the unpredictability inherent to the process, it's possible to predict some impediments the organization will almost certainly encounter. In my follow-up paper, Impediments to Enterprise Agile, I identify those common problems to equip organizations with the knowledge to capably resolve them.

COLLABNET

ABOUT THE AUTHOR

Kane Mar

Kane Mar is an Agile coach and certified Scrum Trainer, specializing in Scrum and Extreme Programming. Prior to his work in Agile software development, Mar spent 15 years as a developer and project manager for waterfall and RUP projects working with Java, Smalltalk, C, SQL, and PL/SQL.

ABOUT COLLABNET

CollabNet leads the industry in Agile application lifecycle management (Agile ALM) in the Cloud. The CollabNet TeamForge™ ALM platform, CollabNet Subversion software configuration management (SCM) solution, and ScrumWorks® project and program management software enable teams using any environment, methodology, and technology to increase productivity by up to 50% and reduce the cost of software development by up to 80%. The company also offers training, including Certified ScrumMaster training, software development process improvement services, and an innovative community management approach to driving enterprise development success. As the founder of the open source Subversion project, CollabNet has collaborative development for distributed teams in its DNA. Millions of users at more than 2,500 organizations, including Applied Biosystems, Capgemini, Deutsche Bank, Reuters, and the U.S. Department of Defense, have transformed the way they develop software with CollabNet. For more information, visit www.collab.net.