

Impediments to Enterprise 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 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?

In the paper Enterprise Strategy for Introducing Agile, I attempted to answer that question and offer a road map to Agile adoption for enterprises embarking on that journey. What that map did not consider are the many roadblocks they encounter along the way. Actual Agile adoption seldom resembles a Sunday drive. Typically, there is very strong resistance to change within an organization, which is hardly surprising. Any deviation from “business as usual” is scary, but Agile represents such a radical shift in how an organization operates that it can often result in a kind of “culture clash.” A company culture can dictate how an organization's employees identify themselves, so reorienting employees to a new way of doing business is like asking employees to re-imagine who they are as well. Consequently, it is common for employees to cling to old rules out of habit, others to refuse to accept the change altogether, and some to view the change as a threat to their security. In fact, an aversion to Agile methodologies almost always stems from an organization or individual's perception that it will threaten their position, authority, or compensation—or more directly, their power and money. As such, the barriers to smooth Agile adoption are almost always cultural and organizational.

Frankly, any project can succeed in Agile; what dictates if it takes off or stalls out are the organizational and cultural constraints of the enterprise.

These impediments can be classified in four main categories: people; processes; technology; and teams, management, and culture. But before I discuss these four major impediments, it's important to consider how and why the organization and culture of an enterprise play such a critical role in determining the outcome of an adoption effort.

WHICH ENTERPRISES WILL SUCCESSFULLY ADOPT AGILE?

I'm frequently asked: “What types of projects are most suitable for applying Agile practices?” But it's even more important to consider the organizational setting where the work gets done. Frankly, any project can succeed in Agile; what dictates if it takes off or stalls out are the organizational and cultural constraints of the enterprise.

Lean Software Development guru Tom Poppendieck sums this idea up perfectly (italics mine):

The constraints on when Agile approaches can be used are primarily organizational and cultural, not project types. Some organizations and some contexts are incompatible with Agile/Lean thinking. When these organizations eventually come up against a strong competitive threat, they will fail to meet it unless they change their values and mindsets. Lean/Agile is, at its foundation, the fourth industrial paradigm, the first being Craft Production, Factory Production with machine tooling, Automation, and Taylorism. These come along every hundred years or so and take a few decades to work through. Each paradigm includes the preceding one and makes it dramatically more productive.

There is no need to sell Agile except to organizations that want to survive long term. If they don't see the threat/opportunity, they cannot succeed with Agile or Lean nor can they sustain economic viability in the long run.¹

When I am working with an organization, the best indication that it will be able to make the transition is a willingness to listen and change accordingly. Many organizations insist that they listen to their employees and foster an Agile culture, but that claim is frequently disproved when Agile values conflict with organizational values. Such a culture clash reveals just how deeply embedded waterfallish mentalities are. Frequently, enterprises are unwilling to break from habit, even when short-term difficulties will yield long-term improvements.

Consider an organization with a lot of up-front analysis and design. How do its employees respond when asked to write User Stories? Do they insist that Use Cases are better than User Stories? Or do they ask to be shown how to write User Stories? Clearly, the employees who are open to the new paradigm will transition to Agile faster and with better results. Other issues, though, are more tightly woven into the fabric of the organization. Consider an organization that traditionally provides individual offices for its developers. How does management respond to the suggestion of common team rooms? Do they justify the status quo or are they willing to experiment with alternatives? In short, for organizations willing to listen and change, there are no issues that cannot be resolved; it is only a matter of time and effort.

That is not to say that change is not disruptive. On the contrary, change of this order sends shockwaves throughout the entire enterprise. When an organization first introduces Agile practices to a team, the team's difficulties are limited to the immediate adoption of the practices and how they alter their familiar working methods. But after the team acclimates to these new practices, the challenges shift to a larger, team-wide scope. This is how concerns over Agile tend to develop: by shifting from local, individual problems to over-arching, organizational ones. As Agile practices spread throughout an enterprise, the scope and nature of the problems mirrors that growth. Although these issues cannot be anticipated in any absolute sense, it is possible to classify them as groups and highlight some of the most typical challenges organizations encounter, which I have divided into the following four categories: people; processes; technology; and teams, management, and culture.

The best way to minimize or prevent these problems is to create an environment that fosters open communication and emphasizes teamwork over individual heroism. But even under the best conditions, there are a number of scenarios that frequently occur.

PEOPLE

That organizations have issues with their employees is hardly a problem exclusive to the introduction of Agile. But Agile methods accelerate the rate of change, exposing problems that had previously remained hidden. The best way to minimize or prevent these problems is to create an environment that fosters open communication and emphasizes teamwork over individual heroism. But even under the best conditions, there are a number of scenarios that frequently occur:

- **Overly Specialized Skill Sets**

Overly specialized skill sets require work to be handed off several times before it can be described as complete. For example, an analyst will describe the problem before handing the product off to a database administrator, who will then work with the database before handing it off to a developer, who will then write the functionality before handing it off to a tester, and so on. The act of handing off partially completed work is very expensive and should be kept to a minimum.

Ideally, there would be no handoffs and cross-functional teams would work together on every step of a product's lifecycle. But since this is seldom an option, the next best thing is to ensure that all handoffs remain informal and that teams pair at every stage of development. Encouraging teams to work in small chunks fosters more interaction and communication between the teams.

- **Lack of Ownership by the Team**

When Agile is first introduced, team members used to receiving assignments often have a difficult time adjusting to the concept of self-prioritization. And with a learned habit of waiting to be told what to do, these individuals may lack ownership of the work to be done. Similarly, employees who have been micromanaged often abuse the freedom of self-organization. Without a regimented schedule or constant supervision, some employees will simply do nothing at all. Addressing this apathy is a considerable challenge for organizations

attempting to transition to Agile. Given that pilot initiatives often receive a great deal of visibility for managers and executives, unmotivated individuals jeopardize the project with low productivity and provide a negative first impression of Agile's potential for success.

To build team-wide accountability, teams need to communicate openly. Strategies to encourage this include allowing the team to work directly with the customer, ensuring that technical conversations are not shut down, and soliciting input from all team members.

- **Refusal to Interact with the Team**

It is far too common for a team member to refuse to participate in the Agile process. Armed with any number of rationales for remaining detached from the team, such individuals "have too many prior commitments," "can't break their work down into increments amenable to Agile development," and "work better on their own." These are excuses that, in the end, deliberately create a division in the team dynamic.

This mentality is extremely common in the public sector, in which employees do not face the same threat of dismissal they would in a private sector setting. To resolve this type of issue requires either a united effort among team members to work with those who intentionally separate themselves or a firm manager who is unafraid to send them away, if necessary.

For Agile methods to succeed, every member of the team needs to contribute. If one or two members of the team continue to put themselves above the team, it is the responsibility of the management to address the problem quickly, usually with the prompt removal of the individual(s).

- **Mixed Signals from Senior Management**

Nothing undermines an Agile project more than a few carelessly chosen words from senior management. Senior management must decide prior to introduction, if it is committed to Agile or not. Regardless of a manager's personal perspective, he or she has an obligation to publicly support the effort.

Conversely, Agile teams and customers must also communicate their experiences to the wider organization or their successes will remain unacknowledged. For Agile to take hold in an organization, teams must interact frequently with senior management to clearly articulate how it is improving operations and what obstacles are impeding continued success.

One of the most appealing attributes of Agile methods is that it is very process-light, which is why it's often described as a framework rather than a process. Still, for something that is conceptually simple, a lot of projects have a hard time keeping it simple.

PROCESSES

One of the most appealing attributes of Agile methods is that it is very process-light, which is why it's often described as a framework rather than a process. Still, for something that is conceptually simple, a lot of projects have a hard time keeping it simple. Here are some common process-related problems:

- **No Single Owner (Product Owner) Can Be Identified**

The inability to identify a single owner is a common problem when there is a large separation between the business units and the software development departments or in situations in which several different groups have an equal interest in the success of a project.

In my experience, most business customers really want their projects to succeed, but they do not understand how to interact constructively with software projects. Methodologies such as waterfall or RUP have reacted to negative customer involvement by distancing the customer from the development teams, whereas Agile integrates the customer into the development process, but with a better understanding of the interaction.

In a situation in which a single customer cannot be identified, it is necessary to identify the project sponsor, the single person who approves the funding of the project. By working with the sponsor and explaining the project team's need for a single business representative, this situation can usually be resolved quickly.

When several different groups share equal interest in the success of a single project, the team still needs a single representative who can work with each of the different groups and prioritize their work accordingly. Again, working with a project sponsor will provide the path to a solution.

- **Combining Elements of RUP and Agile**

When management attempts to combine elements of different, often contradictory methodologies, teams receive the disadvantages of both without the advantages of either. Although there are many different ways to do this, the most common is to pair the RUP management process with XP engineering practices.

When management attempts such a hybrid, it is typically motivated by a desire to provide familiar reports without re-educating senior management or business partners. In reality, this approach underestimates senior management and does the organization a disservice. It is far better to educate management and business partners while adopting a single approach, whether RUP or Agile.

- **Refusal of ScrumMaster to Protect the Team**

The primary function of the ScrumMaster is to protect the team and to remove impediments. It takes courage to report problems to senior management and to make changes that many in the organization may view as a direct threat to their security. Even the best ScrumMasters will have their credibility questioned at some point.

But without an effective ScrumMaster, who will protect the team? Who will stand up to those who would distract the team with unrelated activities? Who will confront the Product Owner to explain that his or her expectations are unreasonable? The role of the ScrumMaster is essential to maintaining healthy communication between team members and to protecting them from distracting external influences. If a team is receiving inadequate support, it is more than likely due to an ineffective ScrumMaster.

There are a number of steps that an organization can take to increase the quality of its ScrumMasters. From a human resources perspective, the role of ScrumMaster must be a clearly defined career path that is valued as “real work” within the organization, rather than “administrative overhead.” As such, ScrumMaster positions should be filled by volunteers, rather than assignments. To help establish such a precedent, it is recommended that an individual with considerable credibility within the organization pilot this role. When an individual with a proven record of leadership and accomplishment fulfills this role, it is much easier for teams to view it as a vital ingredient to the organization’s success.

Process practices are common to Agile methods, but engineering practices are not. Extreme Programming (XP) differs from Agile in that it explicitly prescribes both engineering and process practices. So why would specific engineering practices be of interest and how is this different from other methodologies, such as waterfall and RUP?

TECHNOLOGY

Process practices are common to Agile methods, but engineering practices are not. Extreme Programming (XP) differs from Agile in that it explicitly prescribes both engineering and process practices. So why would specific engineering practices be of interest and how is this different from other methodologies, such as waterfall and RUP?

The benefit of blending engineering and process is that it can reduce the cost of producing a usable product on a frequent and repeatable basis. Iterations are integral to Agile methods and the outcome of those iterations is usually a usable product. It takes some considerable overhead to build, test, and package a product. To do this repeatedly (every iteration) in a cost-effective manner requires a significant amount of automation, hence Agile engineering practices.

Processes which either increase the cost of producing a usable product per iteration or do not actively reduce the cost per iteration should be immediately eliminated. Here are a few examples of wasteful practices:

- **Unreliable Build System and Process**

Every software development team is intimately familiar with the source control, build, and testing of software. It simply makes sense to make this process as efficient as possible. If it takes two hours for a developer to check code and complete a build, then automating the process would free up that time for more constructive use. Clearly, if the team is building and releasing code many times per iteration, any effort expended to create a reliable build and testing framework is well worthwhile.

- **Unaddressed QA Issues**

With more traditional development methodologies, testing is usually addressed at the end of a product's lifecycle. Since quality assurance is an ongoing process in the Agile model, it's not surprising to see teams that take up Agile not pay appropriate attention to defects and code quality. Code quality will quickly become a problem with each additional iteration, especially if the team is building new functionality on top of defect laden code. Moreover, delaying testing until the late stages of a project will result in the equally late discovery of unanticipated side effects. Producing high quality code from the outset should be the goal of every project. It is only by addressing quality issues early and proactively that the full nature of the software can be known.

Agile adoption represents a seismic shift in how an organization operates. To justify and endure the disruption and strain of the transformation, an organization must be experiencing the "pain" of repeated failure.

- **Ineffective Tool Mandates by a Committee or External Parties**

Large organizations often attempt to control the number and types of tools used by project teams. This can be for a number of very good reasons, such as licensing, IP and IP rights management, and security.

But when the authorization process or the tools themselves act as an impediment to the team making progress, something needs to be done. If a tool does not meet the needs of the developers, why should they use it? If the authorization process for introducing CruiseControl takes more than a few weeks, what does that say about the organization's attitude toward software quality?

In that case, the ScrumMaster and project team need to make their point of view known to senior management. They must explain why tools are relevant to the work, how their work should be supported, and how the right tools can drastically impact the quality of the product.

WHY ALL THE DISCUSSION OF AGILE IMPEDIMENTS?

Dealing with these kinds of challenges is a part of everyday work life. Unsurprisingly, Agile methodologies offer no new easy fixes. So what, then, is the value of discussing all the potential impediments to Agile adoption? The answer is that Agile brings problems that were previously ignored, hidden, or otherwise invisible to the surface. In some ways, the transition to Agile will not create more difficulty for an organization; it will simply reveal the difficulties that were there all along. Of course, the first step in addressing these problems is to acknowledge the dysfunction affecting teams and an organization as a whole. Once these difficulties are readily visible, teams can then share information to help management determine how to resolve them and, on a larger scale, help other software organizations facing similar problems.

It should be said here, though, that an Agile adoption is only recommended for organizations with nothing to lose. If an organization is delivering frequent, high-quality product, and consistently meeting deadlines with a waterfall approach to management, it has no incentive to change. Agile adoption represents a seismic shift in how an organization operates. To justify and endure the disruption and strain of the transformation, an organization must be experiencing the "pain" of repeated failure.

TEAMS

Because most Agile frameworks strongly advocate close, cohesive teamwork, teams seldom present impediments to the adoption process. It may take a long time for the team to acclimate to the new working methods, but that learning curve is simply the nature of change, not Agile-specific. Therefore, conflict is more likely to arise between teams rather than between members on a single team. Typical situations which might lead to this friction between teams would be if two teams use different methodologies (i.e. Agile vs. RUP) or if one team is dependent on another. In both situations, the standard solution is to prioritize dependent functionality early and code to an agreed-upon interface, but that is an overly simplistic appraisal. There is a dynamic relationship between dependent teams, which requires ongoing attention to ensure that strong communication is maintained. If an organization fails to acknowledge the risks inherent in dependent projects risks confusion, poor team performance, and, ultimately, failure.

Culture-specific impediments are often related to how an organization rewards its employees, specifically its compensation, promotion, and career-planning models. In an environment where performance is determined by specialization of knowledge, promotion and compensation models reward the compartmentalization of knowledge.

MANAGEMENT

"In the culture of management, the worst thing you can do is admit to anyone that you have a problem you can't handle by yourself. If you really do need help, you have to sneak it in somehow without admitting in public that there isn't any problem at all."

--Gerald Weinberg²

Educating a new client on how to build an Agile organization can be very slow and difficult work. To succeed, you must be able to speak persuasively to many different levels of the organization. Communicating with the team or individual team members is an effective first step because this is the level at which most consultants work. Still, there is a wider audience to consider, which includes functional managers, the PMO, and Human Resources. As I'll explain further below, speaking to the organization at all levels can empower an enterprise to make a smooth transition to Agile.

At the team level, employees often assume management can see changes for themselves and that the value of Agile methods is easily understood. In fact, management requires the same amount of education, coaxing, and convincing as anyone else. But they need to be persuaded in terms that are meaningful for them. For example, instead of discussing TDD, discuss Agile metrics (i.e. burn-down graphs over actual developer hours). Rather than mentioning Pair Programming, frame the conversation around the need for collocation. Also, when working with management, it's helpful to give extra attention to both Agile metrics (or lack thereof) and adaptive planning over predictive planning. The Agile approach to these is counterintuitive for most traditional project managers and requires constant reinforcing.

COMPANY CULTURE

As Ken Schwaber, co-founder of Scrum points out in his Certified ScrumMaster course, an organization's culture is not easily changed. There will always be employees who are against change on principle, explaining that "that's not how we do things here" or "we can't implement that here." As Ken Schwaber points out in his course, these are all "Yes, but..." arguments. There are no easy answers, especially for something as abstract as company culture.

"... logic and culture have nothing to do with one another."

- Gerald Weinberg²

Culture-specific impediments are often related to how an organization rewards its employees, specifically its compensation, promotion, and career-planning models. In an environment where performance is determined by specialization of knowledge, promotion and compensation models reward the compartmentalization of knowledge. Over several years, this divides the organization in

two ways. Firstly, only certain individuals can perform certain activities and, secondly, the organization is managed as a matrix.

The consequence of the first effect is obvious. To illustrate, if Alice is rewarded for her understanding of the security system, for example, then she will not develop her understanding beyond that point until the reward mechanism changes. Likewise, in a competitive environment, Alice has no incentive to share her knowledge with her co-workers, thereby impeding team-wide advancement.

The second effect is much less obvious, but is also a reaction to increased specialization. In order to circulate knowledge and information, groups are formed in which individuals share a common function. Typically, this results in an organization grouped by “types,” such as analysts, architects, developers, testers, etc. Project teams are then assembled by selecting individuals from each of the different groups. Ironically, this functional grouping of people further segregates information. Agile methodologies break down these arbitrary boundaries on a project-by-project basis by encouraging cross-functional teams. Long-term solutions should reward teamwork and breadth of understanding, not encourage individuals to guard it to ensure their own security.⁴

A small team of experienced XP developers and I were asked to help a client introduce XP into the organization. At the end of eight months, we had left the client with only minor success influencing other Agile projects. So what went wrong?

THIS IS WHAT YOU SHOULD NOT DO

When I was a child, one of my favorite books was *The Bike Lesson*,⁵ in which the phrase “This is what you should not do” recurred throughout the story. In the same spirit, I offer the following example of what not to do. To illustrate what happens when all relevant parties are not addressed during the Agile transition process, I will use an actual project of mine from several years ago.

A small team of experienced XP developers and I were asked to help a client introduce XP into the organization. We had some initial success and quickly brought the team up to speed. After a couple of months, we had a collocated team working in pairs to deliver tested software every two weeks. At the end of eight months, however, we had left the client with only minor success influencing other Agile projects.

So what went wrong? Looking back, I made several mistakes of omission:

- I failed to adequately address dependencies between two projects. The result was a delay in completing functionality and this directly impeded the team’s progress.
- I failed to recognize and address issues in regard to developer compensation. Developers were rewarded (by promotion and compensation) for both specialization and building large frameworks. Introducing Agile and encouraging generalization over specialization of skills was a frightening move because it threatened both their position and their year-end bonuses. By failing to address issues of compensation, I failed to influence the architects and, eventually, was unable to make the necessary recommendations.
- I failed to fully educate senior managers on adaptive planning. Consequently, they continued to try and identify fixed end-dates for deliverables, which became increasingly unrealistic with each passing iteration, and cast doubt on the other successes that the project team was achieving.

The transition to Agile is difficult, but strong leadership—and the unity it can command—alleviates some of the fear that accompanies major change.

My reason for providing this example is to make you aware that successfully introducing Agile methods into an organization is not a simple task. It takes a wide range of both technical and political skills. The individuals who are most qualified to lead an organization through Agile adoption are those who have both solid credibility within the enterprise, usually from a long-standing tenure, and a sense of urgency to inspire co-workers to commit to the change. The transition to Agile is difficult, but strong leadership—and the unity it can command—alleviates some of the fear that accompanies major change.

SUMMARY

In both *Enterprise Strategy for Introducing Agile* and this paper, I have attempted to plot out a road map that can lead enterprises toward their destination of Agile adoption. There are many routes an organization can take, but, hopefully, some of the impediments I have identified will help your company make it there. If you find yourself traveling down a similar path, I wish you every success and I would love to hear about your journey.

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.

REFERENCES

1. Poppendieck, Tom. "Why Can't You Use Agile?"
<http://finance.groups.yahoo.com/group/SellingAgile/message/662>
2. Weinberg, Gerald. Secrets of Consulting: A Guide to Giving and Getting Advice Successfully.
3. Ibid.
4. Poppendieck, Mary. "Unjust Desserts."
<http://www.poppendieck.com/pdfs/Compensation.pdf>
5. Berenstain, Stan and Jane. The Bike Lesson. The transition to Agile is difficult, but strong leadership--and the unity it can command--alleviates some of the fear that accompanies major change.