Term Project

ON

Title: Agile Project Management & Transformation Strategy

Submitted By:

Hitesh Sharma

Roll No: 2K14/MBA/508

Under the Guidance of:

Mr Vikas Gupta

DELHI SCHOOL OF MANAGEMENT

DELHI TECHNOLOGICAL UNIVERSITY

BAWANA ROAD DELHI 110042

January-May, 2016

DECLARATION

I **HITESH SHARMA** student of **EMBA 2014-2016** batch of Delhi School of Management, Delhi Technological University, Bawana road, Delhi-42 declare that term project "Agile Project Management & Transformation Strategy" submitted in partial fulfilment of Executive MBA programme is the original work conducted by me.

The information and data given in the report is authentic to the best of my knowledge.

Name of candidate with sign

Place: New Delhi

HITESH SHARMA

Date:

ACKNOWLEDGEMENT

This is to certify that the Project Report title "**Agile Project Management & Transformation Strategy"** submitted to DSM, Delhi technological University, in partial fulfilment of Executive MBA programme, was carried out by "**Hitesh Sharma**" under my guidance.

Dr Vikas Gupta

Asst Prof, DSM

TABLE OF CONTENTS

Contents

Introduction

Experience has shown that there cannot be a prescriptive, one-size-fits-all solution for all organizations turning to Agile to improve product development. Just like Agile and Scrum itself, the process of transitioning organizations to Agile is best achieved using empirical techniques. The Path to Agility is coupled with Services Phases. Each Service Phase is crafted to help the organization move down the Path to Agility as effectively as possible.

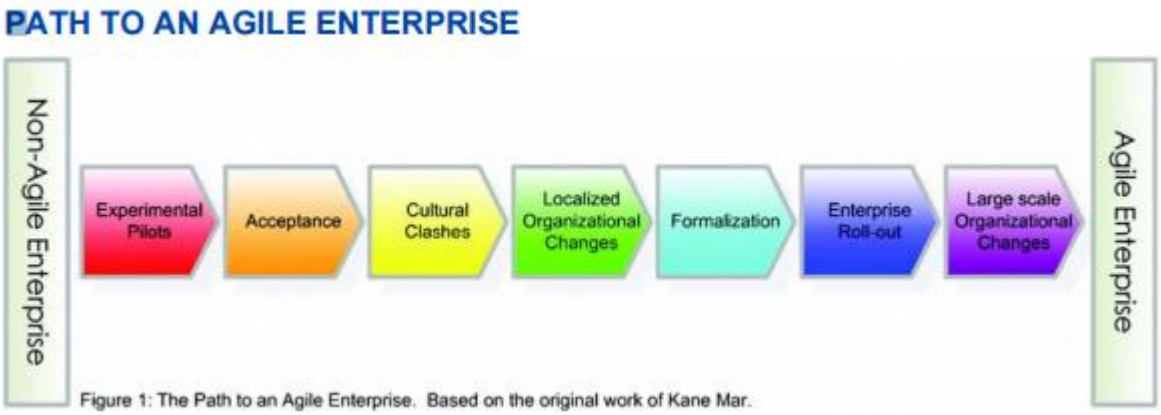


Figure 1: The Path to an Agile Enterprise. Based on the original work of Kane Mar.

Waterfall V Agile

Agile Project Management	Traditional Project Management
Teams are self-directed and are free to accomplish deliverables as they choose, as long as they follow agreed rules.	Teams are typically tightly controlled by a project manager. They work to detailed schedules agreed at the outset.
Project requirements are developed within the process as needs and uses emerge. This could mean that the final outcome is different from the one envisaged at the outset.	Project requirements are identified before the project begins. This can sometimes lead to "scope creep," because stakeholders often ask for more than they need, "just in case."

User testing and customer feedback happen constantly. It's easy to learn from mistakes, implement feedback, and evolve deliverables. However, the constant testing needed for this is labor-intensive, and it can be difficult to manage if users are not engaged.	User testing and customer feedback take place towards the end of the project, when everything has been designed and implemented. This can mean that problems can emerge after the release, sometimes leading to expensive fixes and even public recalls.
Teams constantly assess the scope and direction of their product or project. This means that they can change direction at any time in the process to make sure that their product will meet changing needs. Because of this, however, it can be difficult to write a business case at the outset, because the final outcome is not fully known.	Teams work on a final product that can be delivered some time - often months or years - after the project begins. Sometimes, the end product or project is no longer relevant, because business or customer needs have changed.

Project Background & Brief

There are several ways in which Agile methods can be initially introduced. Most often, developers seeking to improve their immediate situation read existing literature and implement some Agile practices in their small groups. As these developers become more successful, they attract the interest of colleagues and management and the ideas gradually spread. Once sufficient momentum has been established, the organization may want to proceed with experimentation on one or several small projects. This phase of Agile adoption is usually limited in both scope and budget. By choosing smaller projects initially, the enterprise can experiment with this new way of doing things in a way that feels safe, minimizing risk.

Experimental projects help staff and management identify the advantages and disadvantages of Agile and, more importantly, teach everyone lessons on how Agile will be best implemented in the specific organization. We generally find that with demonstrated advantages and a refined and customized approach, most enterprises are ready to move forward with further transitions of more and larger teams and more complex and expensive projects.

Evidence from the experimental projects creates buyoff from all levels of staff, building enthusiasm and willingness to participate in the next Pilot Phase. The Pilot Phase is more organized, but still limited in impact. Pilot projects are seeded throughout the group or department. These pilot projects are more significant than the previous experimental projects in terms of budget, team members, and visibility to both senior management and external clients.

The Pilot Phase amplifies the successes and lessons of the experimental phase and begins to bring organizational impediments to light. Organizational impediments can be defined as circumstances, structures, policies, and ways of doing things, sometimes outside of the scope of the project, that limit project success. Impediments rise to the surface over time.

Initially, they are issues that directly affect the development team: poor tools, inefficient builds, and source code management issues. As project teams become more experienced, quality issues come into focus, including test-driven development, continuous integration, refactoring, etc. Even further into Agile adoption, organizational issues start to emerge, such as project staffing, individual compensation, and promotion .

Naturally, the emergence of these problems can create clashes between Agile team members (who are pushing for change) and those outside the team (who feel that their position is threatened). But, ultimately, resolving these issues results in a lasting change, rather than the “band-aids” that may have previously been applied, resulting in increased effectiveness and better products. If the enterprise is serious about Agile adoption and has gained support from senior management, some of the issues will be addressed.

The first changes will be tactical; those changes that are seen to be immediately needed to support the Agile teams. This might include changes to use a simpler build process, the introduction of continuous integration or reducing the numbers of meetings that teams are expected to attend. These initial changes to the organization are typically limited in scale (confined to a particular group or department) and perceived risk. With growing acceptance of Agile in the enterprise, more and more teams will be interested in adopting Agile on their projects, usually resulting in a sudden increase in the number of Agile projects. This sudden increase in Agile teams will result in different teams taking up different Agile practices, or adopting similar practices, but implementing them in different ways.

For example, some teams will do two-week iterations, some will do three-week iterations, and some teams will do four-week iterations. Some teams will hold daily team meetings, while others will not. And different teams will estimate story points in different ways. The response to these variations will be greater formalization of the Agile methodology.

This formalization will usually be undertaken by a project management office (PMO) who will want to understand what is meant by "Agile within the context of With growing acceptance of Agile in the enterprise, more and more teams will be interested in adopting Agile on their projects, usually resulting in a sudden increase in the number of Agile projects. In order to make a successful transition to an Agile enterprise, any risk (either real or perceived) associated with the introduction of Agile development needs to be resolved promptly by senior management.

The PMO will try to define Agile practices such as terminology, length of iterations, reporting, and metrics. Following the formalization or standardization of the Agile process, the enterprise will be ready to attempt a large-scale rollout. This is a rollout of Agile methods across several departments or organizations.

The types of projects attempted will be far more ambitious and may include Scrums of-Scrums, significant visibility to senior management, and increased corporate risk (associated with the potential of large project failure). These enterprise-wide projects will raise enterprise-wide issues (compensation, promotion, roles and responsibilities) in addition to the usual local problems (tools, build times, quality issues, etc.).

These challenges will be the most difficult for the enterprise to address because they will directly challenge the culture of the organization and will require changes to employees' behavior and dayto-day practices. In addition, some individuals will perceive that their positions and authority within the organization are at risk. In order to make a successful transition to an Agile enterprise, any risk (either real or perceived) associated with the introduction of Agile development needs to be resolved promptly by senior management.

The enterprise that has successfully negotiated this path is an enterprise that is able to manage constant change within the organization; an enterprise that is closely in tune with its customers; an organization that can rapidly change according to changing business conditions at incremental cost (rather than exponential cost) and that has employees who are constantly learning and innovating.

Transition Approach & Strategy

As Agile methodologies gain wider acceptance, they will be taken up by larger numbers of organizations. The question of how to introduce an Agile methodology into an enterprise with the least amount of risk will become more and more common. The two approaches that are typically talked about are top-down, where senior management takes the initiative to introduce Agile, and bottom-up, where developers and testers take the initiative to introduce Agile.

Both of these approaches have flaws. The successful approaches we have witnessed use some combination of both the top-down and bottom-up introduction. Agile software development practices can force large changes in the corporate culture, and making the change to an Agile organization is only possible if there is support from all parties involved.

Ultimately, there needs to be some coordination of both these efforts. There needs to be some planned approach to deal with concerns raised by those who do the work, as well as those who lead the organization and make the strategy a reality.

Strategic Alignment

A key indication of a move towards the Strategic Alignment phase of Agile adoption is that the organization begins to push Agile practices and values out to the entire organization. Leaders in the enterprise value the transparency and predictability that the iterative approach brings to projects, and they begin to realize that the next step in adoption is to push that approach out to the strategic planning process. The graphic below, often referred to as the Planning Onion, illustrates this well: An Agile approach such as Scrum does an excellent job, from early pilots projects onward, of linking the three inner layers of the Planning Onion.

This means all members of a team now Strategy Portfolio Product Release Iteration Day The Coordination phase is marked by experimentation with tools and practices; the Process Definition phase is a formalization of that work. The enterprise must have in place, at this point, tools that allow multiple teams working on the same project to coordinate and integrate work easily.

Naturally, this causes conflict sooner or later. This is when the organization must have and must exercise the skill and knowledge to begin linking the tactical and strategic planning efforts. It should be noted that there is essentially only one group of people who —live in‖ both the tactical and strategic worlds in an Agile organization, and that is the product organization.

With one eye on the current iteration and the other on long-range planning, it is the Product Owners who act as a catalyst to unite the planning process. Working closely with the development teams and upper management, they are the glue that brings together disparate groups to work towards a uniform goal. As the planning effort matures, other processes in the organization must follow suit.

One area that needs to be addressed sooner rather than later is Human Resources. Agile organizations find that old job titles and descriptions no longer meet their needs. New job descriptions, for roles such as ScrumMaster and Product Owner, must be created. New ways of judging performance must also be considered. There is hardly a single role in an enterprise that is not affected at some level by a move to Agile values and practices. Teams, managers, and the product organization must help the Human Resources group find ways to support this new way of working, so the organization can continue to grow and prosper.

Transformation Journey

The final stage of Agile adoption occurs when an enterprise has transformed. Agile Transformation looks a bit different in every organization that achieves it, but there are some common themes. Agile practices are no longer something the ―technical people‖ do – they form the core foundation of values the organization shapes all its actions and decisions around. All employees value and promote transparency, honesty, and making and meeting commitments.

Despite this, one thing an Agile organization is not is free of problems. Agile values do not transport an organization to a magic land where every decision made is automatically the right one, and every conflict disappears with a group hug.

Agile organizations will still have challenges. Impediments will be exposed. But the difference between an Agile organization and one still stuck in traditional management is that an Agile organization will see problems—and opportunities—much sooner, and, more importantly, they will see them for what they are. This vision allows an Agile organization to respond to such situations both faster and more effectively. Individuals in an Agile organization believe the right path exists, so the question then becomes not if something can be done but rather how. And that is a much more empowering approach to problem solving.

CHALLENGES TO AGILE ADOPTION

For some organizations, the transition to Agile is harder than it needs to be. This can be because of a lack of understanding of Agile principles, and a simple lack of experience. But it can also arise when an enterprise tries to take short cuts in their Agile adoption process. Following are some common pitfalls we see when clients run astray of Agile values and goals:

- Underestimating the amount of change that must occur – companies looking for a ―quick fix‖ from Agile practices often succumb to this problem.

  Leaders in such organizations paste an Agile title on their current practices in hopes that the new labels will somehow magically achieve new results. They give their project managers the new label of ―ScrumMaster‖ without ever taking the time to understand how fundamentally different these two roles are. Such efforts are doomed to failure. They will be Agile in name only.

- Ignoring the Product Organization – as mentioned earlier, with Agile approaches such as Scrum, the Product Owners prioritize work into a Product Backlog and ensure the organization is receiving the best return on value possible in its projects.

  A good Product Owner is a shrewd Individuals in an Agile organization believe the right path exists, so the question then becomes not if something can be done but rather how. And that is a much more empowering approach to problem solving, shuffling the priorities of the Product Backlog to tease out more value. But when an organization thinks of an Agile transformation as something only the technical staff participates in, they may neglect to fully engage the Product organization in the process. As a result, with a lack of understanding of either Agile principles or the value they bring, Product Owners are unable to effectively fill their roles and, as a result, projects fail or do not deliver adequate value.

- Neglecting distributed teams – whereas it used to be common for technical teams to be colocated in a single building, for many enterprises this has become the exception rather than the rule.

  Teams comprising individuals that are separated by many time zones are now the norm. When moving to an Agile approach, it is important to remember that these distributed team members need the same support that local staff do in order to be successful with Agile. They need training, and they need tools for collaboration and Agile project management. Too often, organizations give distributed teams a hasty explanation of an Agile approach like Scrum, no tools to help with organization project information, and yet are surprised and disappointed when these teams are not successful. Distributed teams can transition to Agile, but they need the same support given to local teams, and will need an Agile toolset much earlier.

- Failing to make full use of the empirical process – Agile approaches like Scrum control risk through frequent inspect-and-adapt points rather than relying on an extensive, up-front planning process.

  This is the essential nature of the difference between predictive and empirical approaches to project management. However, if an enterprise is still mired in the old way of thinking, it may attempt to still rely on up-front planning in its Agile projects and therefore neglect all the opportunities it has to change approaches that are not working. In Scrum, each structured meeting—the Daily Scrum, Sprint Planning, Review and Retrospective—is an opportunity to examine how well the current approach is working and make adjustments if necessary. Agile teams strive to —fail fast‖, attempting to identify approaches that don't work as early as possible so they can try something new. Everyone involved in the Agile process must learn this skill and be willing to overcome the formerly negative connotations of making mistakes.

- Forgetting to celebrate successes – an experienced Agile team is a low-drama organization.

  Because teams and Product Owners work together closely to make realistic commitments and meet them, much of the behavior that was once rewarded in an organization—for example, pulling all-nighters to reach an unrealistic release date—goes away. From the outside looking in, a good Scrum team can seem almost machine-like – they crank out work predictably, iteration after iteration, essentially forever. What enterprises must remember is that this lack of drama, this dependability is exactly the type of behavior that should be rewarded. Heroics are only great when they work, and too often when this kind of approach fails, it does so spectacularly. Even when the approach is successful, it can come at a terrible price, with whole teams experiencing burn-out and leaving the program or even the company at the first opportunity. It is important to remember that Agile teams are not, in fact, machines. They need to be rewarded and appreciated for the consistent approach to work that they provide. And they need to have opportunities to pursue other kinds of work outside the structure of the iteration that allow them to continue to grow as professionals and individuals. Key to avoiding these common missteps is to be aware how easily they can occur. This is where the benefit of an experienced but neutral coach can smooth the way for an Agile transition.

AGILE TRANSFORMATION STRATEGY

The approach to working with clients making an Agile transition is to provide support at each key stage of adoption. Clients may seek our advice at any stage of an Agile transformation but An Agile coach can help an organization identify new approaches and help them onto the path to Agile transformation more quickly. We often find that, as they proceed through the stages listed above, most clients will need help and guidance in one or more of the following areas:

- Assessment – for complete novices, and for those who have attempted some isolated Agile projects, an assessment can be helpful. During this effort, a coach will work with key members of the organization to observe the practices they have put into place to evaluate their effectiveness. Too often, without a rigorous approach to surfacing impediments and adopting Agile practices effectively, the enterprise can end up doing ―hybrid Agile‖ which is often just another term for getting the worst of both worlds from an Agile and non-Agile perspective. An assessment can help identify these compromises and make a plan to improve them.

- Training – an organized and comprehensive training plan is required for most organizations to fully embark on an Agile transformation. For those enterprises using Scrum as their agile project management framework, Certified ScrumMaster (CSM) and Certified Scrum Product Owner courses are a logical choice. It is crucial that key members of those early Agile teams receive training so that they understand the goals of Agile, how to use the inspect-andadapt method effectively, and how to fulfill the roles they will play in the project. CollaNnet Certified Scrum Trainers can provide certified and non-certified training as the need arises, in both public and onsite private class settings.

- Coaching – after an organization attends its first wave of training, it can be very effective to make use of an onsite Agile coach. Beyond getting teams started on an Agile path, and/or providing structured training, an Agile coach will provide organizations with on the ground help, working directly with teams on their real, day-to-day tasks to improve their effectiveness. Holding effective daily stand-up meetings, helping teams learn to self-manage, and helping the Product Owners learn to write and manage their Product Backlogs more effectively are all examples of Agile coaching activities.

- Tools – at some point—sooner if the enterprise has distributed teams—the organization moving to Agile must evaluate and choose a toolset that supports the new collaborative environment. As Agile practices spread throughout the enterprise, however, an integrated suite of tools is required to scale Agile across the enterprise and enable effective coordination among the teams and management. It is very difficult to implement Agile practices at an enterprise level without the tools to support such an implementation.

Transformation Phases

This plan of action is broken down into three different phases — each phase in turn relies on a combination of both top-down and bottom-up introductions. The three phases are separated based upon their scope of influence as it ever increases 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 Formalization Phase an entire department (many people and multiple projects) may be affected, but the rollout at this point is still confined within the department.

The final phase is the Enterprise Rollout Phase where individuals across multiple departments are affected and projects involve significant budgets and risk to the organization.

Each phase is described below with an outlined list of activities that are commonly performed. For each of the activities, there are a number of associated questions that need to be addressed. The reason that the questions appear without the answers are listed is simply because every organization is different, with different problems, personalities and requirements. Some organizations will have to answer many of these questions in order to be successful, while others will only need a few.

Phase 1. The Pilot

The Pilot Phase is mostly concerned with the immediate rollout of Agile practices to a known team. Concern is often focused on how best to adopt Agile practices and answering practical Agile questions such as how granular should product backlog items be, how do we write stories, how dowe integrate QA, etc.

The Pilot Phase typically lasts between six months and a year and will involve only a small number of experimental projects (between five and 10). These projects will have similar profiles. They will be limited in budget (which implies limits in duration and staffing numbers), scope, and risk. These projects will typically be focused on delivering functionality to an internal client [limiting external exposure], and will have few (if any) dependencies.

Activities undertaken (and questions that need to be addressed) during this phase should include:

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 have he/she act as the ScrumMaster?
- How many teams can a ScrumMaster manage at a time? How large should those teams be?

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 that needs education?

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 baby-sitting?
- 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 isn't responsible for delivering code?

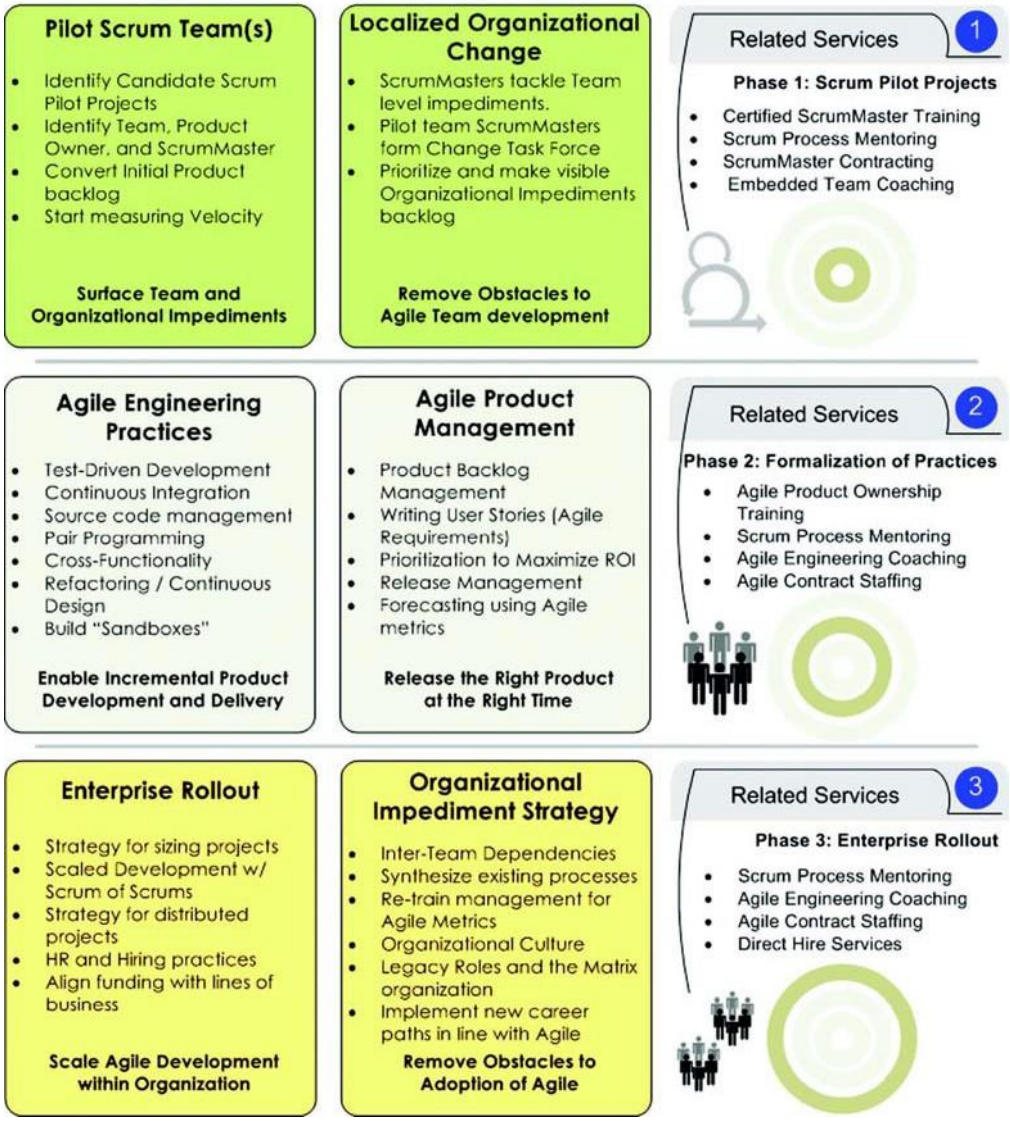


Figure 2: Agile Transformation Overview

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 baby-sitting?
- 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 isn't responsible for delivering code?

Identify likely IP issues (open source tools, code).

- Does the organization have a fear of GPL code?
- Is there a tendency for the organization to re-develop 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 be the Product Owner and what should the team do if the Product Owner doesn't want to engage with 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?

Physical location and layout.

- How important is collocation to the success of an organization's Agile projects?
- Can teams retain their offices and communicate via IM, email, video conferencing, or other tools?
- What about teams in different locations or time zones?

Present Agile to interest 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?

Why We Avoid Traditional Assessments?

In nearly all cases, Agile represents deep changes at nearly every
level of the organization, including sensitive areas such as roles and career paths, company culture, and team dynamics. The prospect of traditional "consultants" coming in and "assessing" an organization in preparation for the next big initiative is enough to make most employees disengage and withdraw from the process. Traditional assessment approaches like interviewing leave a lot to real impediments to change. Interviewees are rarely candid and most often act in their own self- interest or to protect valued initiatives.

Our approach to assessment leverages Scrum's built-in capacity to unearth root problems quickly and accurately. Our approach is simply to start doing Scrum with a pilot project. While pilot project selection is important, the exercise of asking a team to deliver working software in 30 days or less nearly always reveals many of the key team-level and organizational issues likely to block healthy, organization-wide adoption of Agile.

This approach is still valid if your organization has already started using Scrum or Agile. It is our experience that organizations moving to Agile without any professional coaching sometimes relax the Agile rules that cause the most pain to the team or organization. By doing this, the impediments and problems holding it back from realizing its full potential with Agile are masked (not made visible). When we identifies a pilot project in an organization and enforces all rules strictly, the effect is largely the same: the issues holding the organization back bubble to the surface.

Phase 2. Formalization of Agile Practices

The Formalization Phase is focused on how best to rollout Agile methodologies to a much wider audience in a consistent manner. The natural consequence of this is that the Formalization Phase is characterized by substantial codification of the organizational understanding of Agile. This codification will try to address issues such as how long an iteration should be, what tools Agile teams should use, and what formats teams should use for reporting progress. Naturally, these decisions will be influenced by the successes and failures of the Pilot Phase.

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 that these projects will also have a larger number of staff and longer durations. The types of projects addressed in the Formalization Phase will likely represent a cross-section from within a single department. They will likely address many different aspects of the organization's business such as projects to address new functionality, software maintenance, database maintenance, and reporting.

Activities undertaken (and questions that need to be addressed) during this phase should include:

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

- ***Usage of common terminology.***
  What does a Sprint mean? Or an Iteration? What is the Scrum equivalent of Iteration 0?

- **Usage of common metrics.**
  - What scale should the teams use for estimating story points, a scale from one to 10, a scale from 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, and 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 other traditional project management tools?

- **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 co-located?
  - Is there sufficient space to create team rooms?
  - Is there an expense 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 are 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 cost of these projects?

- **Present Agile methodologies to interest 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

The Enterprise Rollout Phase is characterized by focus on communication between projects in different departments; the management, organization and running of very large projects that comprises of multiple Scrum teams; and issues related to compensation, responsibilities, and promotion. These challenges will be the most difficult to resolve and will require considerable persuasive and political skills.

The projects that are typical in this phase of Agile adoption are large, expensive, and potentially risky.

They may be projects that have multiple Scrum projects and managed with a Scrum-of-Scrum, or may involve some element of distributed software development. The Enterprise Rollout Phase is usually much longer than either of the previous two phases. It is possible for projects in each of the previous phases to be initiated and completed within a six- to nine-month period (depending upon the number of people impacted, the types of issues raised, etc.). Projects in the Enterprise Rollout Phase, however, may last anywhere from one to two years.

Activities undertaken (and questions to be answered) during this phase should include:

- **Encourage internal communication regarding Agile.**
  - What is the best approach for encouraging internal communication between different Agile teams? An Agile Forum? Or perhaps an email list is sufficient.

- **Anticipate change and have a plan to evaluate changing circumstances.**
  - 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 don't facilitate a team (Scrum or Scrums-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 the team? Or who should leave a team?

- ***Establish parameters around very large projects with Agile.***

- What qualifies as a large project and 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 represents the Product Owner?
- Should there be a single Product Owner or is it okay to have multiple Product Owners?

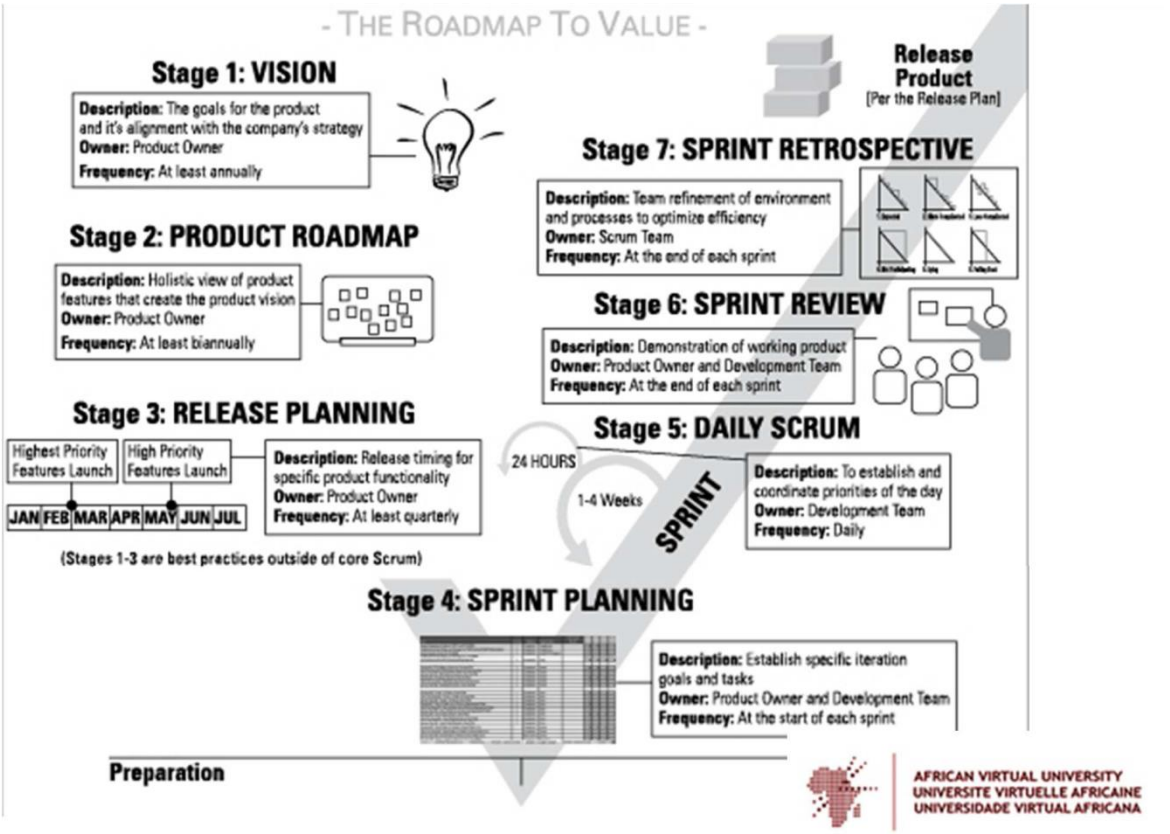
- ***Establish parameters around distributed projects with Agile.***
  - How experienced should the team be?
  - How is communication between teams handled?
  - Should the Product Owner be located within the business or with the development team?
  - Should the entire business be relocated to somewhere more cost-effective?

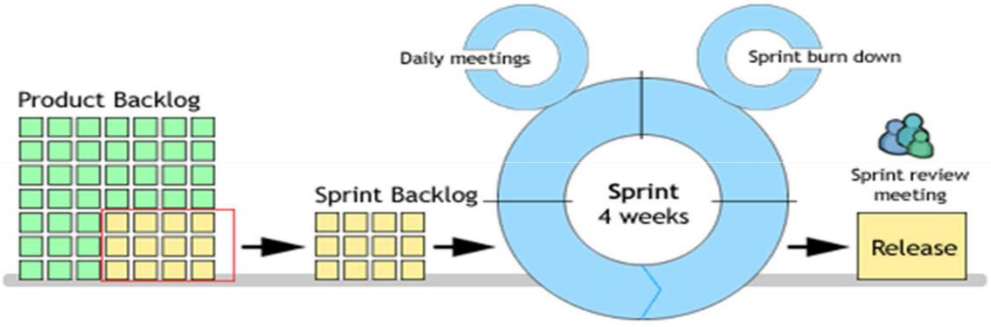
- ***Establish promotion policies.*** **How should successful individuals be promoted?**
  - Should the promotion model be based on merit, influence, or some combination of both?

- ***Establishing training model for coaches/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.***
  - Funding for Agile teams is usually secured by the Product Owner.
  - What does this mean for software development groups that have previously had their own source of funds?
  - How will the management structure react to changes in the funding model for these departments?

Agile Overview



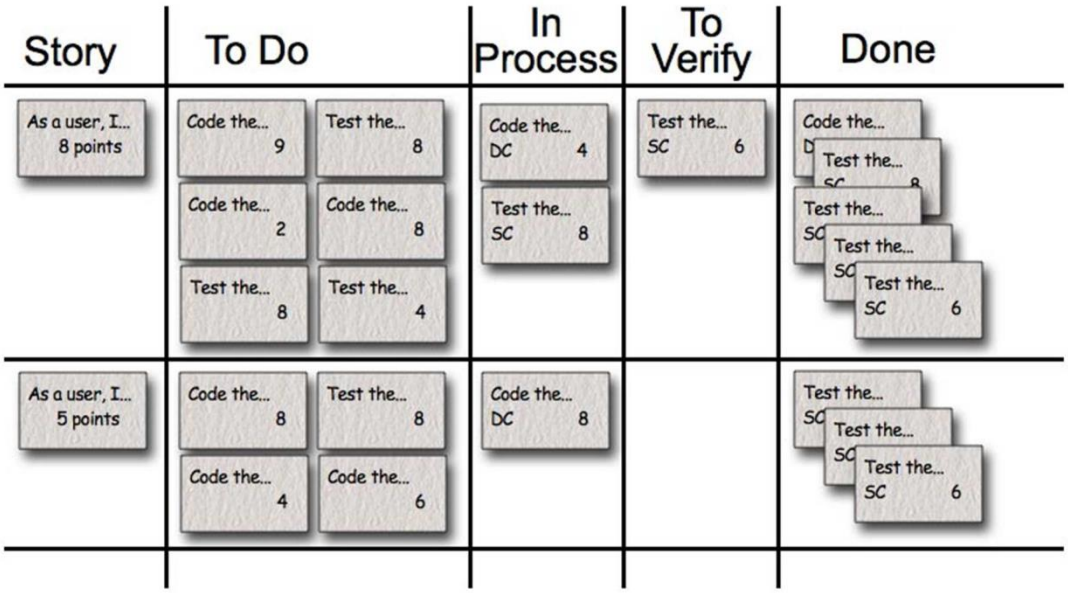


Agile Project Management Events

Most projects have stages. Agile projects include seven events for product development. These events are meetings and stages and are described in the following list:

- **Project planning:** The initial planning for your project.
- **Release planning:** Planning the next set of product features to release and identifying an imminent product launch date around which the team can mobilize. **On agile projects, you plan one release at a time.**
- **Sprint:** A short cycle of development, in which the team creates potentially shippable product functionality. Sprints, sometimes called *iterations,* typically last between one and four weeks. Sprints should remain the same length throughout the entire projects.
- **Sprint planning:** A meeting at the beginning of each sprint where the scrum team commits to a sprint goal. They also identify the requirements that support this goal and will be part of the sprint, and the individual tasks it will take to complete each requirement.
- **Daily scrum:** A 15-minute meeting held each day in a sprint, where development team members state what they completed the day before, what they will complete on the current day, and whether they have any roadblocks.
- **Sprint review:** A meeting at the end of each sprint, introduced by the product owner, where the development team demonstrates the working product functionality it completed during the sprint.
- **Sprint retrospective:** A meeting at the end of each sprint where the scrum team discusses what went well, what could change, and how to make any changes.

Delivery backlog



Agile Project Management Roles

- **Development team:** The group of people who do the work of creating a product. Programmers, testers, designers, writers, and anyone else who has a hands-on role in product development is a member of the development team.

- **Product owner:** The person responsible for bridging the gap between the customer, business stakeholders, and the development team. The product owner is sometimes called a *customer representative.*

- **Scrum master:** The person responsible for supporting the development team, clearing organizational roadblocks, and keeping the agile process consistent. A scrum master is sometimes called a *project facilitator.*

- **Stakeholders:** Anyone with an interest in the project.

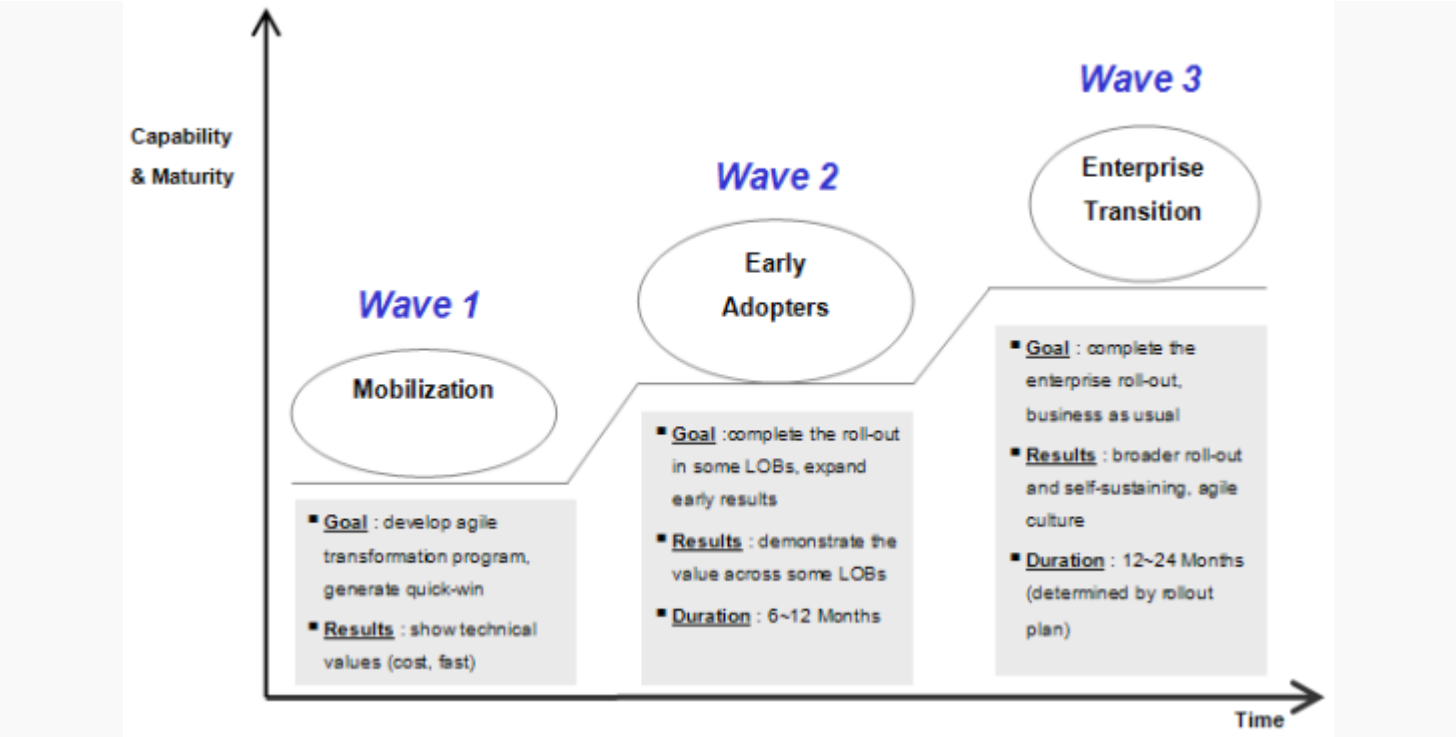
- **Agile mentor:** Someone who has experience implementing agile projects and can share that experience with a project team.

Agile Project Management Artifacts

- Agile project teams often use six main artifacts, or deliverables, to develop products and track progress.
- **Product vision statement:** An elevator pitch, or a quick summary, to communicate how your product supports the company's or organization's strategies. The vision statement must articulate the goals for the product.
- **Product backlog:** The full list of what is in the scope for your project, ordered by priority. Once you have your first requirement, you have a product backlog.
- **Product roadmap:** The product roadmap is a high-level view of the product requirements, with a loose time frame for when you will develop those requirements.
- **Release plan:** A high-level timetable for the release of working software.
- **Sprint backlog:** The goal, user stories, and tasks associated with the current sprint.
- **Increment:** The working product functionality at the end of each sprint.

Adopted Iteratively

Large transformation efforts especially agile transformations are not easy. But economics are driving all organizations to seek creative approaches for cutting costs, improving performance while delivering value to their clients in this competitive marketplace. More companies are moving towards taking advantage of Lean principles to help cut waste and using Agile for Managing IT and Non-IT project efforts for maximum ROI. At IBM we often recommend an iterative and incremental approach for strategic enterprise agile transformation. This high level pattern uses adoption “waves” instead of a rigorous step-by-step procedure (waterfall) which won’t account for the evolutionary nature needed for this transformation. During each wave the team overseeing the agile transformation focuses on different agile practices that correlate to the business values driving that wave.



Agile Adoption Waves

Wave One: (or Mobilization) Focuses on establishing a firm agile foundation and deliver initial value by *establishing a Center of Excellence (CoE)* staffed by resources responsible for the supporting the Agile Transformation. The CoE should be comprised of Agile Transformation team members; Mid-level managers and upper management guide the overall transformation. This team is responsible for

- Obtaining executive sponsorship and commitment
- Cultivating experienced agile mentors and coaches from within the organizations
- Building agile infrastructure (C0E, process, tool)
- Demonstrating measurable value by piloting agile practices on 1 or 2 process to show quick-wins and increase visibility

Wave Two: (or Early Adopters) Focuses on building on early success by helping a relatively small number of internal employees to become internal coach/trainers. The Agile Coaches must work with employees to develop a sufficient number of people who are capable agile facilitators (Scrum Masters). This team engages the trainers/coaches to do the following things roughly simultaneously:

- Conduct awareness sessions to introduce everyone in the organization to agile concepts
- Do initial cultural and process assessments to track progress over time
- Start _everyone_ in the organization using the agile meta-process
-

These are the people will slowly take over from the external coaches. They learn about agile methods more deeply: practices, principles, variations, techniques, and tools. They learn to be effective facilitators who have the trust of their co-workers. These facilitators then become responsible for ensuring that everyone else is using the agile meta-process for effective learning and simultaneously applying appropriate agile practices.

Wave Three: (or Enterprise Transformation) Finally, the coaches work with the Agile Transformation Team to help a relatively small number of employees to become internal coach/trainers. These are the people who will take over from the external coaches. As for ongoing assistance, the coaches should be working in a consultative capacity as the organization struggles with obstacles, restructuring, and the deeper culture changes. Like any change effort, there are five critical components: sponsorship, communication, training, support and strategy. The coaches should be advising the Agile Transformation Team and management on how these five components can best be handled for the agile transformation.

Lessons Learned

Lesson 1: Write down all your mustdo/firm requirements as soon as you know them, and do not plan to collaborate on them

Trying to collaborate on truly firm requirements will only frustrate your team and waste resources. I have witnessed this frustration on multiple occasions during this past year. When I use the term collaborate, I mean an honest consideration of alternatives and a willingness to give. I am not saying do not talk to your customer about the requirements, but I am saying if there is no room to give, then do not pretend there is. Collaboration – in the agile Lessons Learned Using Agile Methods on Large Defense Contracts While the agile movement began on small commercial projects, many contractors are employing these methods today (to varying degrees) on large defense contracts.

Lesson 2: We often confuse nice-to-have requirements with firm must-do requirements.

Lesson 3: Systems engineering is still required with agile development.

Lesson 4: We must get out of the sequential waterfall mentality – this is an outdated way of thinking and it does not work with agile methods.

Lesson 5: Agile does not require fewer written requirements.

It does require collaboration to identify the needed detail to implement what the team is focusing on now in this increment. The word flow-down implies an ordering – something occurring before something else. Systems engineering does its job before software developers do theirs. Systems engineering does the requirements. The developers wait for the handoff. This way of thinking will not work with agile methods. Systems engineering pullback is exactly the reverse of what should be happening. On large projects in particular, there are still some very important sequential activities that must happen. For example, systems engineering must do a high-level first pass of requirements and allocate them to major incremental releases before the developers get going. This is by no means the end of systems engineering. Today, this point is too often being missed.

Lesson 6: The program manager should not assume the agile team knows how to collaborate.

Lesson 7: The program manager's role on an agile project is affected by how he/she interacts with the agile team, particularly with respect to requirements and task lists.

Some program managers have asked, "Does agile affect my job?" If you are the program manager, I recommend that you encourage your team to resolve ambiguous requirements and add missing tasks to the appropriate list. This will ultimately provide more accurate visibility of the real status back to you. Program managers should also let their team know they expect to hear about more issues early and that they will not shoot the messenger.

Lesson 9: Pilot a proven Agile approach (e.g., Disciplined Agile Delivery) first before customizing it(don't start by changing an approach that works)

Lesson 10: Understand the motivation behind a specific practice before (adding / dropping)

Lesson 11: Start with a small Pilot, prove success, learn from the pilot, and then scale Agile adoption (avoid too much change, too fast)

Lesson 12: Impacted Teams must see value so avoid the "Checklist Agile" adoption

Lesson 13: Avoid WAGILE (Waterfall Agile – e.g, BUFR Big Up Front Requirements)

Lesson 14: Use a wave of change Adoption Roll out strategy

Lesson 15: Develop new incentives that recognize team delivery instead of individual delivery

Conclusion

The lack of guidance for project managers of agile development projects has been a gaping hole in the software development community over the past several years. The contrast between the world of agile software development and traditional project management has left many managers wondering what their role should be. By viewing the agile development team as a complex adaptive system and the manager as an integral part of that system, we have begun to develop a framework for managers. This framework of practices is meant to overlay the practices of existing agile methodologies such as XP, and provide clear guidelines for the visionary leadership of projects that use them.

These six practices of agile project management do not provide a sure-fire recipe for success. Building and nurturing a successful team is much more like cooking chili than baking a cake – it requires creativity, flexibility, and attentiveness to the unique qualities and interactions of the ingredients. However, we believe that by following these basic practices and adapting them to your own style over time, managers will not only find that they add tremendous value to projects but also that they will enjoy not only the achievement of success but the journey along the way.

From my experiences, successful enterprise agile transformations require the proper mixture of top-down and bottom up activities. Agile transformation efforts are much smoother when buy in and continued support is obtained from sponsors high up in the organizations. Results are more tangible when the impacted teams help define the values, practices and tools to be adopted. With both approaches in play, I firmly believe that any organization can succeed.

Where does Agile work best?

- Smaller projects
- Lower value projects
- Novel activities
- Where knowledge is limited
- Where scope is changing
- Non-critical systems
- Requirements are not clearly defined

References

1. See http://agilemanifesto.org.
2. See "Benefits of a Top-Down Agile Adoption Strategy", Venkatesh Krishnamurthy, Agile Journal, May 2007

3. See Patterns of Agile Practice Adoption, posted July 13th, 2010 by John Turner

4. See Agile Transformation vs. Agile Adoption,
5. See "Working in Priority Order: Agile Change Management",www.agilemodeling.com/essays/agileRequirements.htm
6. See Agile Advice – Working with Agile Methods (Scrum, OpenAgile),http://www.agileadvice.com/page/3/ (accessed February 16, 2011).