Agile Project Management

www.ccpace.com

Table of Contents

I. Introduction 3

II. The Problem: Project Manager as Uninspired Taskmaster 4

III. The Solution: Project Manager as Visionary Leader 6

IV. The Means: An Agile Project Management Framework 7

V. Conclusion 15

VI. References 16

I. Introduction

Today's Information Technology (IT) manager is under ever-increasing pressure to deliver results – in the form of applications that drive improvements to the bottom line – even while IT budgets are being significantly slashed. Meanwhile, despite the fall of the Internet economy business environments continue to change at a rapid pace leaving many IT shops struggling to keep up with the pace of change. These changes have led to an increased interest in agile software development methodologies with their promise of rapid delivery and flexibility while maintaining quality.

Agile methodologies such as eXtreme Programming (XP), SCRUM and Feature-Driven Development strive to reduce the cost of change throughout the software development process. For example, XP uses rapid iterative planning and development cycles in order to force trade-offs and deliver the highest value features as early as possible. In addition, the constant, systemic testing that is part of XP ensures high quality via early defect detection and resolution.

In spite of some early success with agile methodologies, a number of factors are preventing their widespread adoption. Agile methodology advocates often find it difficult to obtain management support for implementing what seem like dramatic changes in application development. These methodologies require developers, managers and users alike to change the way they work and think. For example, the XP practices of pair programming, test-first design, continuous integration, and an on-site customer can seem like daunting changes to implement. Furthermore, these methodologies tend to be developer-centric and seem to dismiss the role of management in ensuring success.

As managers of several successful XP projects, we have found that strong management is absolutely critical to the successful adoption and application of agile methodologies. But we have also discovered a lack of alignment between the methodologies and tools of traditional project management and those of newer agile methodologies. Furthermore, we believe this misalignment is symptomatic of a deeper problem – differences in fundamental assumptions about change, control, order, organizations, people and overall problem solving approach. Traditional management theory assumes that:

- Rigid procedures are needed to regulate change
- Hierarchical organizational structures are means of establishing order
- Increased control results in increased order
- Organizations must be rigid, static hierarchies
- Employees are interchangeable "parts" in the organizational "machine"
- Problems are solved primarily through reductionist task breakdown and allocation
- Projects and risks are adequately predictable to be managed through complex up-front planning

Within this context, it is small wonder that the new methodologies appear informal to the point of being chaotic, egalitarian to the point of actively fostering insubordination, and directionless in their approach to problem solving. We believe that the slow adoption of agile methodologies stems mainly from this misalignment between the fundamental assumptions of traditional management and those of the new agile development methodologies. As such, we believe there is a significant need for a change in assumptions and a new management framework when working with agile methodologies.

I. Introduction

In the search for a new framework, we have come to believe strongly in emerging management principles based on the "new science" of complexity that exploit an understanding of autonomous human behavior gained from the study of living systems in nature. Specifically, we have begun to build the notion of complex adaptive systems (CAS) into our management assumptions and practices.

Complexity scientists have studied the collective behavior of living systems in nature such as the flocking of birds, schooling of fish, marching of ants and the swarming of bees. They have discovered that, while the individual "agents" in these complex adaptive systems possess only local strategic rules and capacity, their collective behavior is characterized by an overlaying order, self-organization, and a collective intelligence that is greater than the sum of the parts. The theory of CAS has been applied successfully in several areas – economics, life sciences and more recently, to management.

The concepts of CAS led us to the inspiration that like the XP team, project managers also need a set of simple guiding practices that provide a framework within which to manage, rather than a set of rigid instructions. Following these practices, the manager becomes an adaptive leader – setting the direction, establishing the simple, generative rules of the system, and encouraging constant feedback, adaptation, and collaboration. This management framework, covered in detail in Section 4, provides teams implementing agile methodologies with:

- An intrinsic ability to deal with change
- A view of organizations as fluid, adaptive systems composed of intelligent living beings
- A recognition of the limits of external control in establishing order, and of the role of intelligent control that employs self-organization as a means of establishing order
- An overall problem solving approach that is humanistic in that:
    - It regards employees as skilled and valuable stakeholders in the management of a team.
    - It relies on the collective ability of autonomous teams as the basic problem solving mechanism.
    - It limits up-front planning to a minimum based on an assumption of unpredictability, and instead, lays stress on adaptability to changing conditions.

II. The Problem: Project Management as Uninspired Taskmaster

Traditional software lifecycle development methodologies grew out of a need to control ever-larger development projects, and the difficulties of estimating and managing these efforts to reliably deliver results. These methodologies drew heavily on the principles from engineering such as construction management. As a result, they stressed predictability (one has to plan every last detail of a bridge or building before it is built), and linear development cycles – requirements led to analysis which led to design which in turn led to development. Along with predictability, they inherited a deterministic, reductionist approach that relied on task breakdown, and was predicated on stability – stable requirements, analysis and stable design. This rigidity was also marked by a tendency towards slavish process "compliance" as a means of project control.

While these methodologies may have worked for some organizations in the past and may still work in some circumstances, for many companies these methodologies only added cost and complexity while providing a false sense of security that management was "doing something" by exhaustively planning, measuring, and controlling. Huge costs were sunk in premature planning, without the rapid iterative development and continuous feedback from customers that we have come to realize are prerequisites for success today.

The results are stark – repeated, public failures such as the London Ambulance System and the Denver Airport Baggage system earned the software industry a reputation for being "troublesome" with huge cost overruns and schedule slippages. Consider the results of the Standish Group's CHAOS surveys. In the first survey, it was estimated that only 18 percent of all software projects were considered successful, 31 percent were failures and 53 percent were challenged. Comparatively, the 1998 figures showed a marked improvement in which 26 percent were successful, 46 percent were challenged and 28 percent were failures. The study attributed the increase in success to scaling the size of projects back to manageable levels using smaller teams. This result is clearly in line with the principles of agile methodologies. Furthermore, we have found that many established project management practices still apply to agile development projects – with some adaptation and a strong dose of leadership.

While managers designed traditional methodologies in an effort to control projects, the technical community gave birth to agile methodologies in response to their frustrations with traditional management (or lack thereof) and the resulting impact on their products and morale. For example, the principles of XP are focused almost entirely on the development process. While the technical community has championed these principles, very little has been written about the management side of agile development projects. The implication is that there is little need for a project manager since XP teams develop and monitor their own tasks. No wonder that corporate management has been skeptical of agile methodologies and slow to embrace them. Managers conjure up an image of a room full of developers doing their own thing…. and the name "eXtreme" doesn't help matters either!

Regardless of the particular methodology, the traditional project manager is often seen as a "taskmaster" who develops and controls the master plan that documents (often in excruciating detail) the tasks, dependencies, and resources required to deliver the end product. The project manager then monitors the status of tasks and adjusts the plan as necessary. Underpinning this mechanistic approach is the assumption that equates individuals to interchangeable, controllable commodities.

So for many managers comfortable with traditional methodologies, the prospect of implementing agile methodologies on their development projects can be daunting. But it doesn't need to be. In fact, independent of agile methodologies, other trends in project management indicate a point to a convergence between the management community and the technical community.

III. The Solution: Project Manager as Visionary Leader

The best project managers aren't just organizers – they combine business vision, communication skills, soft management skills and technical savvy with the ability to plan, coordinate, and execute. In essence, they are not just managers – they are leaders. While this has always been the case, agile project management places a higher premium on the leadership skills than ever before.

For example, XP teams create and monitor their own iteration plans in collaboration with the customers. The customer creates stories (features) and prioritizes them based on business value. The developers divide up the tasks themselves as they work and measures progress for each iteration (time-boxed development cycle), adjusting plans with the customer as necessary. So, if the project no longer needs a detailed master project plan, why does it need a project manager?

Because every project needs a leader. Agile methodologies free the project manager from the drudgery of being a taskmaster thereby enabling the project manager to focus on being a leader – someone who keeps the spotlight on the vision, who inspires the team, who promotes teamwork and collaboration, who champions the project and removes obstacles to progress. Rather than being an operational controller, the project manager can become an adaptive leader – if she can relinquish her reliance on old style management.

The basic phases of an agile development project are really no different from those of any other project. You still must define and initiate the project, plan for the project, execute the plan, and monitor and control the results. But, the manner in which these steps are accomplished are different and require the project manager to retrofit what they know about traditional management to a new way of thinking – the thinking of complex adaptive systems. The practices outlined below provide a framework for project managers working in this new world.

IV. The Means: An Agile Project Management Framework

The authors have applied XP successfully on several projects over the past years, and evolved the use of XP practices as an integral part of a CAS inspired framework for agile project management, as described in Section 4.2. Section 4.1 provides a guiding philosophy of the team as a complex adaptive system.

4.1 A Guiding Philosophy: The Team as a Complex Adaptive System

As the literature will attest, traditional command-and-control management is largely derived from the principles of Frederick Taylor's "scientific management." Taylor's scientific management approach was based in turn on the seventeenth century science of Newton that saw the world as a vast and magnificently ordered "clockwork universe" governed by the classical laws of nature. Scientific management is recognized as the prime mover in lifting the "working masses" in developed countries to new levels of affluence in the 20th century.

In today's world, however, we have trouble imposing command-and-control management on teams because "working masses" have been replaced by knowledge workers. In the computer software industry for example, we have situations where skilled software developers are often worth as much or more to their employers than their managers. In Taylor's world, it was the manager who had the specialized problem solving knowledge. In ours, this key problem solving knowledge resides with the knowledge workers, and not the manager. So, how do we adapt project management techniques to deal with this key reality?

The scientific world has changed. For nearly two centuries after Newton, his ideas held sway, and found widespread adoption in many other disciplines. Subsequent advances in the sciences – from Einstein's relativity thinking to quantum physics – have since replaced the Newtonian world-view in many disciplines. In particular, a more recent revolution in the scientific community looks set to finally change traditional management – the new science of *complexity.*

Over the past two or three decades, scientists have explored living systems in many fields – as diverse and biology and economics – to search for common properties that explain complex phenomena such as Darwinian natural selection and increasing returns on the stock market. They have uncovered that many natural systems (brains, immune systems, ecologies, societies) and many artificial systems (parallel and distributed computing systems, artificial intelligence systems, artificial neural networks, evolutionary programs) are characterized by complex behaviors that emerge as a result of interactions among their component systems at different levels of organization.

These results have been used to unravel the mysteries of the collective behavior of living systems in nature such as the flocking of birds, schooling of fish, marching of ants and swarming of bees for strategic purposes. While the individual "agents" in these groups possess only local strategic rules and capacity, their collective behavior is characterized by an overlaying order, self-organization, and a collective intelligence that is greater than the sum of the parts. In addition, these living systems regularly display a remarkable ability to adapt to a complex and dynamic environment.

In a nutshell, complexity holds forth some fundamental ideas about living systems gleaned from the facts of nature:

- Living systems are *complex,* in that they consist of a great many agents interacting with each other in a great many ways.
- The interaction of individual agents is governed *by simple, localized rules.*

- The richness of the interactions of the agents allows the system as a whole to undergo *spontaneous self-organization,* whereby complex order, known as emergent order, arises from the system itself, rather than from an external dominating force.
- These complex, self-organizing systems are *adaptive* in that they react differently under different circumstances.
- Holistic patterns *emerge* that overlay the individual behavior of the agents.
- These systems co-evolve with their environment (changes in the environment cause changes in their behavior, which in turn cause changes in the environment) to a point where a dynamic equilibrium is reached. This point where continuous learning and adaptation are in balance with continuous change has been called the *edge of chaos.*

If we view our organizations and teams as complex adaptive systems, then knowledge of CAS learned elsewhere can be applied to drive a new philosophy of management. In particular, the rules of traditional project management can be retrofitted to a new CAS model. The authors have applied XP successfully on several projects over the pastyears, and evolved the use of XP practices as an integral part of a CAS inspired framework for agile project management, as described in Section 4.2.

4.2 A CAS-Based Project Management Framework: Six Practices for Managing Agile Development Project

We have established a CAS-based project management framework with six Agile Project Management (PM) practices for managing agile development projects – *Guiding Vision, Teamwork and Collaboration, Simple Rules, Open Information, Light Touch and Agile Vigilance.* Together these practices help us to manage our teams as complex adaptive systems while allowing us the freedom to overlay our own personal leadership styles. The six practices build on the fundamentals of CAS, as shown in Table 1.

These practices are explained in further detail in Sections 4.2.1 through 4.2.6.

Table 1.
CAS Principals and Corresponding Agile Project Management Practices

CAS Principle	Corresponding Agile Project Management Practice
Non-material fields exert force on material objects.	*Guiding Vision.* Recognizing vision as a non-material field rather than an elusive destination results in vision continuously guiding and influencing behavior in positive ways.
Autonomous, intelligent agents form the basis of CAS. *Interactions* between these agents result in *self-organization* and other *emergent* phenomena.	*Teamwork and Collaboration.* Recognizing individual team members as intelligent, skilled professional agents and placing a value on their autonomy is fundamental to all other practices. Teamwork and Collaboration form the basis for rich interactions and cooperation between team members.
Local, strategic rules support complex, overlaying behavior in a team environment.	*Simple Rules.* Simple Rules such as XP Practices support complex, overlaying team behavior.
Information is energy that serves as an agent of change and adaptation.	*Open Information.* Open information is an organizing force that allows teams to adapt and react to changing conditions in the environment.
Emergent order is a bottom-up manifestation of order, while imposed order is a top-down manifestation.	*Light Touch.* Intelligent control of teams requires a delicate mix of imposed and emergent order.
Non-linear dynamical systems are continuously adapting when they reach a state of *dynamic equilibrium* termed the *edge of chaos.*	*Agile Vigilance.* Visionary leadership implies continuously monitoring, learning and adapting to the environment.

4.2.1 Practice #1: Guiding Vision – Establish a guiding vision for the project and continuously reinforce it through words and actions.

CAS theory informs us of *non-material fields* that exert real force on material objects in the universe. For example, Gravity – a field familiar to us – is a force of attraction exerted by a celestial body, such as the earth, upon objects near or upon its surface that draws them closer to its center. These fields are thus understood to be forces with both magnitude and direction that permeate and influence the space and objects around them.

As articulated by Margaret Wheatley [1], when a project vision is translated into a statement of the greater purpose and dreams of the organization, and communicated to all members of the team, it serves as a field that has a powerful effect on their behavior. It can permeate the project environment and influence team behavior in extremely positive ways, much more so than a simple task can. The vision needs to become a guiding force that helps the team make consistent choices, rather than embody an elusive end state on a piece of paper.

A real example of this principle is the use of the "commander's intent" in the U.S. Army. The Army knows that its leaders cannot be everywhere in the field of combat controlling all the decisions. Therefore, Army leaders clearly establish the "commander's intent" to serve as a guide on which soldiers can base their own initiatives, actions and decisions. Thus, even if the mission falls on the shoulders of the lowest ranking person, she must be able to understand and carry out the mission.

Likewise, you, the agile manager, can guide the team and continuously influence team behavior by defining, disseminating and sustaining a guiding vision. At the outset of the project, work closely with the customer to understand the vision for the project, how it is expected to support business goals, and how it will be used. To promote team ownership of the vision, facilitate a group discussion with the team to build a joint project vision. A strong grasp of the vision will help the team through difficult decisions about business value and priority and keep them focused on and inspired by the ultimate goal.

The traditional process of reducing project tasks into ever-smaller components for assignment and tracking often causes degeneration into "fractal" tasks, tasks at ever repeated smaller scales. The traditional tool for guidance – a project plan with fractal tasks – often has tasks at too small a level to be really meaningful. Instead, maintain a focus on the forest over the trees and promote a planning process that keeps tasks at a level that sets intent and desired outcome, while preserving flexibility for the team innovation and autonomy.

Throughout the project, gently guide the team to maintain focus on the vision. Everyday decisions and interactions are opportunities to reinforce the vision and create positive energy. Beware of actions that are not consistent with the vision and your message, this kind of dissonance creates the negative energy that deflates teams and inspires many Dilbert strips. For example, in planning sessions, ask questions to provoke thinking about whether stories and the assigned business value are in line with the vision.

4.2.2 Practice #2: Teamwork & Collaboration – Facilitate collaboration and teamwork through relationships and community.

Self-organization and emergent order are due in part to rich interactions between agents in a CAS. These phenomena are explained by expressing the sum of the interactions of a CAS as a gestalt *connectivity* with each agent working in alignment with other agents. It is this connectivity that we believe can be manifested through teamwork and collaboration.

We have all seen that when people work together leveraging complementary individual strengths the results can be exceptional. But getting people to work this way can be a challenge and it cannot happen by mandate. The project manager's role is to actively facilitate collaboration and establish the conditions for good relationships.

Good relationships among team members starts with the project manager's relationship with the team members. You set the standard and are the role model for the others. You need to take steps to get to know each team member as a person – know what makes each of them tick outside of work and what motivates each of them at work. In addition, by treating each person with respect you establish the model for working relationships on the team.

In addition to getting to know the team members yourself, you should help team members get to know each other also by creating opportunities and the right conditions. Opportunities can be created from planning games, everyday interaction, and special events. To set the right conditions, you must establish an environment in which team members treat each other with respect. You may even need to intervene to stop disrespectful behavior.

We recognize many managers may not be able to pick and choose their team but, if at all possible, the first practical step in building a collaborative team is selecting team members with the right attitude and complementary skills. Particularly, if the organization has not worked with XP before, the team members should be people who are adaptable and willing to try new ways of working, although having a few non-believers can have its advantages. In theory, XP teams have no experts – all developers work on all aspects. In reality, sometimes experts are needed when the team is learning some new tools or a specific component requires technology with which the organization has no experience. You must ensure that the role of experts and learning goals are clearly defined in order to achieve positive collaboration.

This initial stage of the project also provides the project manager with opportunities to get to know the team and help them get to know each other. The time-honored kick-off group lunch can be combined with techniques often using in training sessions such as sharing personal and professional information with a colleague who then makes the group introduction. In addition, the project manager should ensure that the physical workspace is arranged in a way that facilitates collaborative activities such as pair programming and team problem solving. Ideally, the team should be located in an open space with both individual and common areas.

Keep in mind that such open but close quarters have the potential to both encourage and inhibit collaboration. Some people may not be comfortable bringing their technical problems to the group. You should be finding ways to gradually get developers used to this mode of working such as beginning with pair programming and smaller groups and demonstrating that bringing a problem to the group is not a sign of weakness. Some developers want to ask for help but aren't good at coming out with it. Start to learn individual team members' signals. For example, on one project a developer would signal interest in starting a dialogue by taking his earphones of and "coughing".

Planning sessions are fertile ground for developing a common understanding and respect between the developers and the Customer – something that is often sadly lacking in many application development projects. With the right kind of leadership, as the project progresses these sessions can become highly collaborative and creative resulting in improved morale and a better product. Basic facilitation techniques such as making sure all parties have an opportunity to speak, summarizing and confirming, and drawing out concerns can help to build the team.

There are many situations that can impede collaboration such as disrespectful treatment, egotism, and non-performing team members. The project manager must monitor the team dynamics and decide when to intervene.

As the project progresses, continue to look for special opportunities to get to know people better and to help the team know each other. For example,

- Establish a regular day for group order-in or potluck lunches
- Giving team members fun (positive!) nicknames
- Celebrating successes and milestones with nominal gifts that reflect knowledge of staff interests (e.g., music, gift certificates, special foods).

The team that laughs and plays together works together better.

4.2.3 Practice #3: Simple Rules – Establish and support the team's set of guiding practices.

In a CAS, agents follow simple rules, but their interactions result in complex behavior emerging from the bottom-up over time. For example, birds in a flock follow simple rules such as avoiding objects, keeping pace and staying close to other birds. By following these simple rules, flocks of birds exhibit complex, collective behavior by flying in formation for long distances and adapting to changing conditions along the way. The gestalt order that emerges is a result of following these simple rules.

We have used the twelve standard practices of XP as a set of simple rules for our software development projects. The XP practices provide the team with a flexible structure within which to work. To use the XP practices as simple rules, they must be explicitly stated and agreed to by all members of the team at the outset, although the team should have the ability to modify practices that are not working or add new practices. If the developers and the customer have not used XP before, provide the team with training on the full set of XP practices. Often, a one-day seminar on the practices including some XP exercises to simulate the planning game and short development iterations is sufficient. Based on this knowledge the team can discuss how best to apply the practices on the particular project at hand.

Take a leading role in encouraging the team to try certain practices about which team members may be doubtful. For example, on one of our XP projects a developer doubted the effectiveness of the *test-first design* practice but was able to quickly see the value after being encouraged to try it.

In applying the XP practices, you set up simple generative rules that are just enough to provide clear boundaries, but no so much as to restrict the autonomy and creativity of the team. Throughout the project, appropriately point out when practices are not being followed and seek to understand why, looking for opportunities to adjust and improve on the practices or their practical use.

4.2.4 Practice #4: Open Information – Provide open access to information.

In a CAS, information is the lifeblood of change and adaptation. Interactions between agents involve the exchange of information. The richness of the interactions between agents therefore, depends in large part on the openness of the information.

For an agile team to be able to adapt, information must be open and free flowing. Traditional managers have long prevented this openness and freedom because of a fear that it will result in chaos. Because of this fear, traditional managers have controlled information and meted it out on a "need to know" basis. On traditionally managed projects, teams often feel like they don't know what is going on – only the project manager has the "master plan" and only the project manager interacts with project sponsor.

In the agile world, information is freed to leverage its power. XP practices, for example, promote open access to information– story cards are public property, as is visible documentation of all status information such as the tracking data. *Collective code ownership* encourages everyone to contribute to the project. Customer and developer are placed in close proximity via *on-site customer* to promote an open exchange of information.

To promote open information, try a variety of techniques:

- Place team members within close proximity of each other whenever possible.
- Make use of information radiators [2] such as whiteboards, charts, etc to disseminate information
- Rather than have status meetings with the project sponsor(s) in an office or conference room, bring him/her to the project room for public status reports and hands-on demos.
- Use a team wiki [3] (free form web site written by users) to share information.
- Establish daily status meetings to promote the flow and exchange of information.
- Sustain open information exchange between business domain experts and the development team.

4.2.5 Practice #5: Light Touch – Apply just enough control to foster emergent order.

In traditional management, everything is seen through the prism of control: change control, risk control and most importantly – people control. Elaborate methodologies, tools and practices have been evolved to try and "manage" an out-of-control world. But tools fail when neat linear task breakdowns cannot easily accommodate cyclical processes, and neat schedules require frequent updating to reflect the reality of changing dates and circumstances. Complex start-to-finish plans laid out in advance of a project carry a certain naïve optimism that the future won't stray too far from what has been laid out.

In the zealousness of imposing more and more control, managers seem to have forgotten the original purpose of control – to create order. As traditional managers, we had come to believe that more control would give us more order. Unfortunately, this conventional view doesn't really help us in the uncertain real world because life is characterized by probabilities, not certainties. As experience teaches, unforeseen events can lay the best of plans to naught in an instant. Skilled professionals do not take well to micromanagement. Tools and techniques reach their limitations quickly when used inappropriately.

4.2.4 Practice #4: Open Information – Provide open access to information.

In a CAS, information is the lifeblood of change and adaptation. Interactions between agents involve the exchange of information. The richness of the interactions between agents therefore, depends in large part on the openness of the information.

For an agile team to be able to adapt, information must be open and free flowing. Traditional managers have long prevented this openness and freedom because of a fear that it will result in chaos. Because of this fear, traditional managers have controlled information and meted it out on a "need to know" basis. On traditionally managed projects, teams often feel like they don't know what is going on – only the project manager has the "master plan" and only the project manager interacts with project sponsor.

In the agile world, information is freed to leverage its power. XP practices, for example, promote open access to information– story cards are public property, as is visible documentation of all status information such as the tracking data. *Collective code ownership* encourages everyone to contribute to the project. Customer and developer are placed in close proximity via *on-site customer* to promote an open exchange of information.

To promote open information, try a variety of techniques:

- Place team members within close proximity of each other whenever possible.
- Make use of information radiators [2] such as whiteboards, charts, etc to disseminate information
- Rather than have status meetings with the project sponsor(s) in an office or conference room, bring him/her to the project room for public status reports and hands-on demos.
- Use a team wiki [3] (free form web site written by users) to share information.
- Establish daily status meetings to promote the flow and exchange of information.
- Sustain open information exchange between business domain experts and the development team.

4.2.5 Practice #5: Light Touch – Apply just enough control to foster emergent order.

In traditional management, everything is seen through the prism of control: change control, risk control and most importantly – people control. Elaborate methodologies, tools and practices have been evolved to try and "manage" an out-of-control world. But tools fail when neat linear task breakdowns cannot easily accommodate cyclical processes, and neat schedules require frequent updating to reflect the reality of changing dates and circumstances. Complex start-to-finish plans laid out in advance of a project carry a certain naïve optimism that the future won't stray too far from what has been laid out.

In the zealousness of imposing more and more control, managers seem to have forgotten the original purpose of control – to create order. As traditional managers, we had come to believe that more control would give us more order. Unfortunately, this conventional view doesn't really help us in the uncertain real world because life is characterized by probabilities, not certainties. As experience teaches, unforeseen events can lay the best of plans to naught in an instant. Skilled professionals do not take well to micromanagement. Tools and techniques reach their limitations quickly when used inappropriately.

4.2.6 Practice #6: Agile Vigilance – Constantly monitor and adjust.

The common thread throughout all the practices is this final practice – Agile Vigilance. In computer simulations, artificial systems that operate within a framework of simple rules can sometimes display amazingly lifelike behavior such as reproduction. Their most interesting behavior occurs at the border between order and chaos – unpredictable enough to be interesting and ordered enough to avoid falling into chaos. It is our contention that the most creative and agile work of a team occurs at this hypothetical edge of chaos. However, just as in dynamical non-linear systems, we believe that operating on this edge requires continuous learning and adaptation to changing environmental conditions.

Of course, all good things come with a price. To paraphrase Thomas Jefferson, the price of agility on the edge of chaos is eternal vigilance. In leading a team by establishing a guiding vision, fostering teamwork and cooperation, setting simple rules, championing open information, and managing with a light touch, the job of the agile manager has been likened to herding cats – each person has his or her own ideas, and is likely to behave in accordance with those ideas.

You, the agile manager, therefore must be continually vigilant to merit the mantle of leadership: monitoring progress, and keeping a finger on the pulse of the development team. This does not mean hovering and controlling everything – remember, you have established simple rules and must trust in your people and the process. Instead, it means being observant, continuously seeking feedback and monitoring success or failure, and adapting by making changes as situations warrant:

- Reinforce the guiding vision at every opportunity – examine project decisions to see whether they line up with the vision.
- Continually encourage teamwork and collaboration. Talk to your team members one-on-one as often as possible to keep a pulse on the heartbeat of the project. Watch for signs of stress – rising tempers, fatigue, etc, and deal with them quickly. Keep abreast of technology so that you can interpret the “tribal language” of your software developers.
- Establish simple rules, but take every opportunity to conduct process reflections: regularly examine what works and what needs improvement. Act with courage to make changes when you feel they are necessary.
- Work relentlessly to break down the barriers to information sharing. Keep apprised of cultural sensitivities, egos, and other such factors that may impinge upon its success. Operate with a light touch. Intervene quickly, but wisely to solve personnel issues. Motivate and reward initiative, but manage expectations. Recognize and encourage self-organization, but disallow cliques.

V. 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.

VI. References

[1] Wheatley, Margaret. *Leadership and the New Science: Discovering Order in a Chaotic World*. Berrett-Koehler Publishers, 1999.
[2] Cockburn, Alistair. *Agile Software Development*. Addison Wesley Longman, 2001.
[3] The original wiki web site is http://c2.com/cgi-bin/wiki.

For Additional Information Contact:

CC Pace
4100 Monument Corner Drive
Suite 400
Fairfax, VA 22030
703/631.6600
www.ccpace.com
info@ccpace.com