Lean and Agile Project Management: For Large Programs and Projects

David F. Rico[1],

[1] Severn, Maryland, USA
dave1 (at) davidfrico.com

Abstract. This talk discusses how agile methods can be used for managing high-risk, time-sensitive R&D-oriented new product development (NPD) projects with demanding customers and fast-changing market conditions. It establishes the context, provides a definition, and describes the value-system for lean and agile project management. It provides a brief survey of popular lean and agile project management approaches and illustrates the mechanisms for scaling the lean and agile project management model up to large-scale, distributed projects. It also illustrate a few key agile project management case studies as well as basic, burnup/burndown, cost estimating, business value, earned value management, and advanced metrics for agile methods including real options. Finally, this talk addresses the critical differences between agile and traditional non-agile project management paradigms, as well as the debate surrounding the pros and cons of agile certification.

Keywords: Lean thinking; lean development; agile methods; agile project management; complex adaptive systems; systems thinking; flexibility, high-performance teams; adaptive, iterative, incremental, collaborative, participative, and rolling wave planning; real options; business value; return on investment, costs and benefits; earned value management; metrics; models; measurements.

1 APM Introduction

Agile Project Management (APM) is a new paradigm for managing high-risk, time-sensitive, research and development-oriented new product development projects [1]. APM seems to be the ideal model for modern, post-industrial information age knowledge workers. In reality, however, APM has a long and rich history and lineage. Tenets of APM can be traced back to the principles of experimentation used by Louis Pasteur in the 1800s and Thomas Edison in the early 1900s, organismic biology by Bertalanffy in the 1920s, cybernetics by Weiner in the 1940s, systems theory by Boulding in the 1950s, systems dynamics in the 1960s by Forrester, double-loop learning by Argyris in the 1970s, learning organizations by Senge in the 1980s, adaptive planning by Highsmith in the 1990s, and many others who are too numerous to mention here [2]. The fundamental notion or theory underlying APM is that modern systems are complex, not well-understood, subject to the forces of dynamic and unstable market conditions, technology intensive, and constantly changing [3].

Counter to the principles of complex adaptive systems are traditional methods based on scientific management principles pioneered by Adam Smith and Frederick Taylor in the British and American industrial revolutions of the 1800s and 1900s [4]. Key ideas emerging from this paradigm were division of labor, specialization, time and motion, Gantt charts, mass production, hierarchical organizations, and most other principles associated with 20th century manufacturing. The basic notion behind traditional methods is that all system requirements can and should be documented, work breakdown structures should be carefully constructed, all activities should be defined and scheduled, cost and effort estimated, and then meticulously detailed project plans should be carefully controlled using techniques such as earned value management to within a 5% or 10% level of precision [5]. After software-intensive systems reached crisis proportions in the 1960s, the term software engineering was coined, and many people began applying principles of traditional methods to software development as a means of controlling project scope, time, and cost.

While the proponents of Taylorism attempted to control chaos with scientific management principles, others began to rediscover the job-shop practices of highly creative and innovative individual artisans, mathematicians, and scientists used throughout the ages [6]. Part of this rediscovery included the formation of the human school of management in the 1930s and 1940s, autonomous work groups in the 1950s, computerized manufacturing in the 1960s, flexible manufacturing in the 1970s, new product development in the 1980s, and lean thinking in the 1990s [7]. Although the leading thinkers had already discovered that incremental planning was superior to long-term strategic planning in the 1970s, it wasn't until 2000 that traditional methods were officially declared obsolete [8]. The basic notion behind modern ideas is that inductive thinking is better than reductionism, chaos can't be controlled, planning should be done a little bit at a time, planning should be participative with the key stakeholders it affects, products should be built in smaller chunks, and projects should be frequently re-planned to dynamically adapt to changing market conditions [9].

2 Types of Major APM Models

As large, heavyweight traditional methods such as SW-CMM, CMMI, ISO 9001, ISO 12207, ISO 15288, PMBoK, SEBoK, and SWEBoK were in their golden age, agile methods finally emerged in the 1990s and 2000s [10]. Agile methods didn't emerge out of thin air, but were firmly based on autonomous work groups from the 1950s, end user involvement from the 1960s, iterative development from the 1970s, and rapid application development from the 1980s [11]. The major ones emerged in this order, Crystal Methods, Scrum, Dynamic Systems Development Methodology, Feature-Driven Development, and finally Extreme Programming (XP). XP emerged in 1998 and took the world by storm. In 2001, the creators of these methods formed what is known as the Agile Manifesto, which was a common set of operating principles. It was based on four broad values: (1) customer collaboration, (2) iterative development, (3) self-organizing teams, and (4) adaptability to change [12]. Shortly on their heels emerged the paradigm of APM, with models such as release planning, sprint planning, radical project management, extreme project management, and APM.

2.1 Sprint Planning

Scrum, one of the earliest forms of agile methods, was created by Jeff Sutherland at Easel circa 1993 [13]. Scrum is generally comprised of four broad stages, sprint planning, sprints, sprint review meetings, and sprint retrospective meetings. However, more emphasis has been placed on the project management components of Scrum. One view of Scrum divides its project management model into two broad phases, initial planning and the sprint cycle. The initial planning sub-phase consists of a discovery session when projects are initiated, scoped, and organized. It also consists of a release planning sub-phase when a project backlog is formed consisting of prioritized user needs and a general timeline for multiple development sprints. The sprint cycle phase consists of a sprint planning sub-phase, the development sprint itself, daily team meetings, sprint reviews, and retrospectives.

2.2 Release Planning

XP, one of the most popular agile methods, was created by Kent Beck at Chrysler circa 1998 [14]. Scrum influenced the creation of XP, although Scrum's project management model was refined based on XP. Originally, XP was comprised of 13 practices: planning game, small releases, metaphor, simple design, tests, refactoring, pair programming, continuous integration, collective ownership, on-site customer, 40-hour weeks, open workspace, and just rules. However, XP's project management model is comprised of two broad phases, release planning and iteration planning. The release planning phase consists of three sub-phases, exploration, commitment, and steering. During this phase, user needs are captured, prioritized, and a release plan is formed with a timeline for multiple iterations. During the sprint planning phase, technical tasks are formed, estimated, and executed to build out the product.

2.3 Extreme Project Management

Extreme Project Management (XPM) was created by Doug DeCarlo of the Cutter Consortium circa 2004 for all types of projects [15]. XPM's design was influenced by Rob Thomsett's Radical Project Management model, Jim Highsmith's APM model, and Kent Beck's XP model. Its motivation came from chaos theory and complex adaptive systems, and resembles a lightweight project management model for new product development. XPM consists of five broad phases: visionate, speculate, innovate, re-evaluate, and disseminate. A broad vision for the project and product is formed during the visionate phase. The output of the speculate phase is a project plan and the innovate phase is used to iteratively develop the solution. Finally, the project's and product's status are assessed during the re-evaluate phase and distributed to customers in the disseminate phase if it is successful.

2.4 Agile Project Management I

Another APM model was created by Sanjiv Augustine, then of CC Pace, circa 2004

[16]. Sanjiv's model was influenced by Jeff Sutherland's Scrum model, Kent Beck's XP model, and Jim Highsmith's APM model. Sanjiv's model focused on two broad areas, a leadership model to establish the organizational culture for agile methods and a broad framework for managing agile projects. There are three broad phases in Sanjiv's model: foster alignment and cooperation, encourage emergence and self organization, and learning/adaptation. The first phase consists of establishing organic teams and an overall project and product vision. The second phase consists of establishing simple rules, a climate of open information exchange, and light-touch for just the right balance of flexibility and discipline. The last phase focuses on learning and adaption at both the organizational and project levels.

2.5 Agile Project Management II

An influential model of APM was created by Jim Highsmith of the Cutter Consortium circa 1994 [17]. The design of Jim's model was influenced by Rob Thomsett's Radical Project Management model, Jeff Sutherland's Scrum model, and Kent Beck's XP model. Jim's model is based on four major ideas, establishing a project and product vision, planning for multiple releases, using agile practices for product development, and bringing administrative closure to a project. There are two broad phases in Jim's model, innovation lifecycle and iterative delivery. The first phase consists of envisioning a product, speculating or creating a release plan, exploring the product's development, launching a successful product, and closing it out administratively. The second phase consists of technical planning, product development, operational testing, adaptation, deployment, and a variety of other continuous activities.

3 Scaling APM to Large Programs and Projects

As use of agile methods spread, traditional methodologists felt they were only for very small projects, although they were never designed with this limitation in-mind [18]. Literature emerged that exhibited the applicability and scalability of agile methods to large programs and projects. Some of the major techniques for doing so included multi-level teams, plans, backlogs, coordination, and governance [19]. Multi-level teams are comprised of product management teams who primarily interface to the customer, release management teams who plan agile projects, and feature teams who are responsible for managing day to day development. Multi-level plans consist of product roadmaps, release plans for multiple iterations, and iteration plans for day to day activity. Multi-level backlogs consist of capabilities or epics, feature sets or themes, and user stories or system-level requirements. Multi-level coordination consists of capability teams, feature-set teams, and feature teams (also known as a Scrum of Scrums). Multi-level governance also consists of governing, functional, and feature teams for establishing program and project policies, standards, processes, tools, and non-functional requirements. Numerous other scaling techniques are emerging from the literature on distributed teams.

4 Metrics and Models for APM

Many seek to identify the right blend of metrics and models for APM [20]. For some, the goal is to map traditional metrics to those of agile methods. For instance, basic metrics for size, effort, productivity, complexity, quality, testing, and reliability apply to agile projects as well as traditional ones. However, size and productivity may be measured in terms of story points, which is similar to function points. Productivity or velocity refers to story points per sprint or iteration and are tracked using burndown or burnup charts. This gives a basic measure of work completed within a two to four week period. Basic effort and cost models are starting to emerge based on lines of code, function points, and user stories per hour. Business value is measured in terms of costs, benefits, breakeven point, benefit to cost ratio, return on investment, net present value and real options [21]. Some are willing to adopt the use of agile methods, so long as they can apply earned value management, which led to the emergence of AgileEVM [22]. However, agile project plans have a much shorter time horizon than traditional ones, and change frequently. While traditional projects are designed for small changes, agile projects are designed for larger size, cost, and scope changes, as long as it results in greater business value.

5 APM Case Studies

Thousands of projects are now using agile methods on a world-wide basis. As a result, hundreds of documented APM case studies have emerged over the last 20 years. While it is not the purpose to analyze all of them, five agile case studies will be examined here, by Google, Primavera, FDA, FBI, and the U.S. DoD, in order to illustrate the range of industries applying APM. As an illustration of electronic commerce, Google used Scrum on one of its largest projects, Ad words, in order to improve project planning, estimation, and quality [23]. As an example of the shrink-wrapped software industry, Primavera used Scrum on a 100 person team to achieve dramatic quality improvements and cycle time reductions [24]. As an example of the highly-regulated healthcare market for safety-critical systems, Abbott used Extreme Programming to achieve significant cost, schedule, staff-size, and quality improvements [25]. As an example of a large, traditional civilian law enforcement government agency, High-Performance Technologies used Extreme Programming to achieve dramatic productivity and quality improvements [26]. Finally, as an example of a large, traditional U.S. DoD government agency, FGM used Extreme Programming to improve teamwork, productivity, and quality [27].

5 APM Summary

APM is a fundamentally new paradigm, and is not simply a lighter weight traditional project management approach. At its core, APM is also based upon the four major values of agile methods: (1) customer collaboration, (2) iterative development, (3)

self-organizing teams, and (4) adaptability to change. Therefore, new metrics should be used to reflect these four values, rather than simply applying traditional measures. On average, APM results in 50% improvements in cost, schedule, quality, and personnel resources [28]. Agile certification is a topic of debate, although it helps form a common understanding of processes and terminology, create a more disciplined workforce, show a commitment to its values, and result in recognition [29]. Common agile myths are slowly being disproven: they are only for small co-located software teams, they don't scale up to large projects, and they are undisciplined. A frequently asked question is "When is it appropriate to use traditional versus agile methods?" Early theories asserted that traditional methods were better for large projects, while agile methods were for small ones [30]. However, this is misguided as agile methods were designed the uncertainty and instability found in large projects, while traditional methods were not (according to historical project failure rates associated with using them).

References

1. Thomsett, R. (2002). Radical project management. Upper Saddle River, NJ: Prentice-Hall.
2. Rico, D. F. (2007). Effects of agile methods on website quality for electronic commerce. Retrieved on July 27, 2010, from http://davidfrico.com/rico07q.pdf
3. Highsmith, J. A. (2000). Adaptive software development: A collaborative approachch to managing complex systems. New York, NY: Dorsett-House.
4. Pine, B. J. (1992). Mass customization: The new frontier in business competition. Boston, MA: Harvard Business School Press.
5. Project Management Institute. (2008). A guide to the project management body of knowledge (PMBOK Guide) (4th ed.). Newtown Square, PA: Author.
6. Thomke, S. (2003). Experimentation matters: Unlocking the potential of new technologies for innovation. Boston, MA: Harvard Business School Publishing.
7. Shafritz, J. M., & Ott, J. S. (2001). Classics of organization theory. New York, NY: Wadsworth Publishing.
8. Mintzberg, H. (1994). The rise and fall of strategic planning. New York, NY: Free Press.
9. Chin, G. (2004). Agile project management: How to succeed in the face of changing project requirements. Broadway, NY: Amacom.
10. Highsmith, J. A. (2002). Agile software development ecosystems. Boston, MA: Addison-Wesley.
11. Rico, D. F., Sayani, H. H., & Field, R. F. (2008). History of computers, electronic commerce, and agile methods. In M. V. Zelkowitz (Ed.), Advances in computers: Emerging technologies, Vol. 73. San Diego, CA: Elsevier.
12. Agile Manifesto. (2001). Manifesto for agile software development. Retrieved July 27, 2010, from http://www.agilemanifesto.org
13. Schwaber, K. (2004). Agile project management with scrum. Redmond, WA: Microsoft Press.
14. Beck, K., & Fowler, M. (2001). Planning extreme programming. Upper Saddle River, NJ: Addison-Wesley.
15. DeCarlo, D. (2004). Extreme project management: Using leadership, principles, and tools to deliver value in the face of volatility. San Francisco, CA: Jossey-Bass.
16. Augustine, S. (2005). Managing agile projects. Upper Saddle River, NJ: Pearson Education.
17. Highsmith, J. A. (2009). Agile project management: Creating innovative products. Boston, MA: Pearson Education.

18. Boehm, B. W. (2002). Get ready for agile methods with care. IEEE Computer, 35(1), 64-69.
19. Leffingwell, D. (2007). Scaling software agility: Best practices for large enterprises. Boston, MA: Pearson Education.
20. Rico, D. F., Sayani, H. H., & Sone, S. (2009). The business value of agile software methods: Maximizing ROI with just-in-time processes and documentation. Ft. Lauderdale, FL: J. Ross Publishing.
21. Rico, D. F. (2008). What is the ROI of agile vs. traditional methods? TickIT International, 10(4), 9-18.
22. Sulaiman, T., Barton, B., & Blackburn, T. (2006). Agile EVM: Earned value management in scrum projects. Proceedings of the Agile 2006 Conference (Agile 2006), Minneapolis, Minnesota, USA, 7-16.
23. Striebeck, M. (2006). Ssh: We are adding a process. Proceedings of the Agile 2006 Conference (Agile 2006), Minneapolis, Minnesota, USA, 193-201.
24. Schatz, B., & Abdelshafi, I. (2005). Primavera gets agile: A successful transition to agile development. IEEE Software, 22(3), 36-42.
25. Rasmussen, R., Hughes, T., Jenks, J. R., & Skach, J. (2009). Adopting agile in an FDA regulated environment. Proceedings of the Agile 2009 Conference (Agile 2009), Chicago, Illinois, USA, 151-155.
26. Babuscio, J. (2009). How the FBI learned to catch bad guys one iteration at a time. Proceedings of the Agile 2009 Conference (Agile 2009), Chicago, Illinois, USA, 96-100.
27. Fruhling, A., McDonald, P, & Dunbar, C. (2008). A case study: Introducing extreme programming in a U.S. government system development project. Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008), Waikaloa, Big Island, Hawaii, USA, 464-473.
28. Mah, M. (2008). Measuring agile in the enterprise: Proceedings of the Agile 2008 Conference, Toronto, Canada.
29. Ambler, S. W. (2007). Coming soon: Agile certificatino. Dr. Dobb's Journal, 32(7), 67-69.
30. Boehm, B., & Turner, R. (2004). Balancing agility and discipline: A guide for the perplexed. Boston, MA: Addison-Wesley.