

WILL AGILE DEVELOPMENT CHANGE THE WAY WE MANAGE SOFTWARE PROJECTS? AGILE FROM A PMBOK® GUIDE PERSPECTIVE

**Nathalie Udo, President, Projectway, LLC
Sonja Koppensteiner, Founder, InterGlobe Consulting**

ABSTRACT

Although software has been developed for over 20 years, software companies are still unable to manage software projects with predictable outcomes like releasing on time, within budget and meeting requirements. The AgileAlliance has introduced a new promising way of software development called “Agile Software development”. How promising is it? Priorities of this methodology are to satisfy customers through early and continuous product delivery as well as allowing changes to requirements late in the development cycle. How does this affect industry accepted project management processes? As more companies implement this new methodology, will this be the start of a new era that will change the way software projects will be managed?

This paper will highlight the basics of Agile development methodologies and assess how they align with the project management processes outlined in A Guide to the Project Management Body of Knowledge (PMBOK® Guide): initiating, planning, executing, controlling and closing a project. We will investigate how the PMBOK® Guide knowledge areas compare to Agile software development. To conclude, we will discuss where we see the challenges and opportunities for project management of Agile developments.

INTRODUCTION

Software languages, operating systems, and computer architectures have evolved tremendously in the last two decades. Although new technologies play a role in the development of new project management models, the major driving forces are not technological. A key driving force lies in the evolution of the relationship between technology and business. In the late 60's all the power lay with the computer experts. They dictated the schedule and the business would have to wait until the product was ready. Over time the power base shifted gradually to the business, which would shop around for solutions and dictate the time-to-market. Both power extremes are not ideal and have resulted in today's frequent problematic relationships between IT and business.

The software industry has realized that there is a need for collaboration between the business and technical aspects of software development in order to improve time-to-market and customer satisfaction. IT and business professionals are becoming more aware of each other's needs and concerns, which is resulting in partnerships, that last throughout the project duration [Thomsett 2001]. On top of that, the increasing global competition is requiring an even faster turnaround of software products.

These two trends are driving the development of new project management paradigms, catalyzed by a programming community that sees the current comprehensive project management models often as burdens to software development. Some of their key complaints are that existing project management processes are often too bureaucratic and slow to respond to the dynamic nature of software development. The software industry is faced with users who only know what they want after they see an initial version of the software. This means that requirements are not fully understood at the start of the project and change during the software development phase. Based on this frustration and the need for faster product turnaround, software organizations have started to implement empirical software development processes such as the Agile methodologies.

AGILE METHODOLOGIES

A common cause of disaster in software development is that the end product is precisely what the customer originally ordered [Economist 2001]. In today's fast pacing world, customer's objectives are constantly being revised. As a result, software development needs to be able to hit a moving target, thus the birth of Agile development.

According to Highsmith [Highsmith 2002, p. xxiii], the definition of Agile is “the ability to both create and respond to changes in order to profit in a turbulent business environment”. There are three main focal points within Agile. The first one is that organizations are “chaordic” having characteristics of both chaos and order. The second point is that trust is the basis for all internal and external relationships with a focus on collaboration and not contract negotiations. The final focal point is that the focus lies on people and interactions first and on processes second.

Within the Agile development world there are several development approaches. The following gives a brief synopsis (based on [Highsmith 2002]) of the most important approaches:

Adaptive Software Development (ASD) – developed by Jim Highsmith, evolved out of Rapid Application Development (RAD) in 1992. ASD is designed to embrace change in complex, uncertain environments. ASD recognizes that failure is ok, and that teams must continually adapt to their specific project and situation. The ASD life cycle involves an iterative process of Speculate-Collaborate-Learn. Planning occurs in the Speculate phase. The term “speculate” is a more imprecise term than “plans”, and acknowledges uncertainty. Development occurs in the Collaboration phase. The Learning phase reflects on the project so that the team can adapt the project and their processes. The project manager’s role is to facilitate team collaboration.

Crystal Methods – developed by Alistair Cockburn, has a framework that matrixes communications needs (team size), project priorities, and system complexity. Although the Crystal framework acts as a process guide, processes are ultimately tailored to the team. Communication and people come first and processes come second. Control is limited. Crystal follows only two absolute rules: use incremental development cycles less than four months in duration, and use reflection workshops to adapt the methodology.

Dynamic Systems Development Model (DSDM) – originated in England in the mid 90’s, also grew out of RAD. DSDM is controlled by a consortium, and has extensive documentation, service and support available. Although benefits of the consortium are available only to members, it is relatively inexpensive to join. DSDM stands out as being the only Agile methodology that is ISO 9000 compliant. This could make it attractive for companies that require ISO 9000 compliance. DSDM has three main processes: Functional Modeling, Design & Development, and Implementation. Sub-processes exist for each main area, and all processes are iterative. Iterations are small and “time-boxed”.

Extreme Programming (XP) – is probably the most popular Agile methodology and generates the most interest. XP focuses on developer and customer interactions. XP also identifies the suitable environment for use: co-located teams of 10 or fewer developers with an onsite customer dedicated fully to the project. Like other Agile methodologies, development occurs over small iterations (three weeks or less).

Feature Driven Development (FDD) – Unlike other Agile methodologies, FDD claims to be based on “repeatable” processes. FDD places more emphasis on modeling than other Agile methodologies. Projects start with a skeletal model that is continually updated as the design matures. Features are defined as business functionality that can be delivered in two-week increments or less. Features drive the development life cycle in short two-week iterations (similar to all Agile methodologies). FDD also places a high emphasis on project management. Requirements are managed throughout the entire life cycle. Feature development variances of 10% or more go to management for scope decisions (reduce features or move the schedule).

Lean Development (LD) – emerged from the lean production movement in manufacturing in the 1980’s. LD is more strategically focused than other Agile models. The goal of LD is to simultaneously reduce the time, cost and defects of software development by one third. In order to accomplish these goals, incremental changes are not enough. Major improvements require radical thinking and support from the senior management. LD requires adoption at the highest levels of management in order to insure success.

Scrum – developed by Ken Schwaber & Jeff Sutherland, has been around for approximately ten years. Scrum has more of a project management emphasis than other Agile methodologies. Controlled chaos projects require different control techniques, and Scrum uses explicit monitoring processes and constant feedback mechanisms. Development occurs over 30-day iterations called sprints. Daily “scrum” meetings (30 minutes or less) are held with the entire team to identify roadblocks and provide status feedback. Requirements are held constant through the sprint to provide some stability in a quickly changing environment.

AGILE PROCESSES COMPARED TO PMBOK® GUIDE PROCESSES

The basis of the PMI PMBOK® Guide methodology is a set of process groups that define project management “best practices”. The philosophy is that projects are composed of processes that interact throughout the project and generally fall into two major categories: (1) *project management processes* describe, organize and complete the work of the project; (2) *product-oriented processes* specify and create the project’s product. The project management processes can be organized into five process groups [PMI 2000, p. 30]:

1. **Initiation**
2. **Planning**
3. **Executing**
4. **Controlling**
5. **Closing**

The question is how do the Agile practices align with the PMBOK® Guide process groups. Exhibit 1 shows a high level overview of the core aspects. The main characteristics of Agile development and the PMBOK® Guide are discussed by process group below.

	PMBOK® Guide	Agile
Initiating	Authorization of new project or next project phase	Focus on identifying the business needs for new product development
Planning	Rolling wave planning for the whole project (sequential)	Preliminary whole project planning followed by separate planning for each iteration
Executing	Execution of project plan	Per iteration development of identified feature set.
Controlling	Emphasizes on change control management to minimize changes	Open to scope changes throughout the process with limitations on changes during iterations.
Closing	Formal acceptance.	Customer acceptance per iteration

EXHIBIT 1: Agile compared per PMBOK® Guide process groups

Initiation

According to the PMBOK® Guide, initiation processes initiate a new project or the next project phase. The focus is to get formal approval to continue. The PMBOK® Guide emphasizes on the identification of stakeholders and of their needs, objectives and expectations for the project but does not go into the stakeholders expectations of success at this stage. Most of Agile methodologies start by describing how requirements get collected and prioritized as well as how the team is formed. Rapid Planning (RAP), one of the extreme planning methodologies in use, also describes a zero planning stage that focuses on establishing the project success expectations of each participant [E. Thomsett 2001].

Planning

The biggest differences between PMBOK® Guide and Agile are visible in the planning processes. PMI puts a lot of importance on the planning processes. The number of planning processes demonstrates this: there are 11 core and 10 facilitating planning processes out of all 39 processes described in the PMBOK® Guide. The planning processes are subject to frequent iterations prior to completing the project plan. The final product of the planning processes is a project plan including a schedule.

The general essence of Agile is small teams working in short iterative cycles comprised of planning, development, testing, and acceptance [Schwaber 2001]. This means that Agile planning processes are very simple. A preliminary estimate is made of each requested feature. These estimates are added up to the total amount of days it will take to complete the full project (see exhibit 2a). Every iteration is a fixed amount of days and starts with a joint meeting between the customer(s), who can be external or internal, and the development team. The customer(s) prioritizes the requested feature list and the development team chooses the feature set it thinks it can develop within that iteration. The tasks for the iteration are planned and the workload is tracked on a regular basis against the remaining work and time left in the iteration. After the iteration, time to complete the full project is recalculated based on the results of

the previous iteration and any changes the customer(s) has made, including priority changes or adding or deleting features.

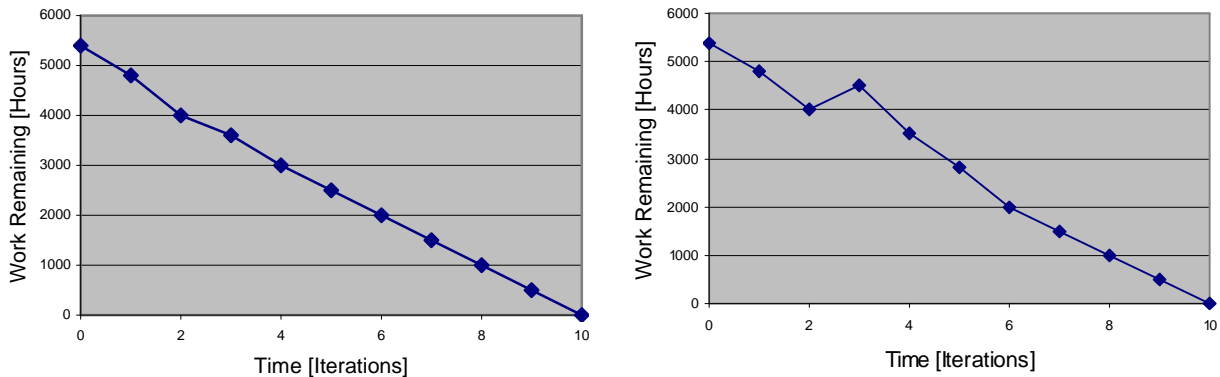


EXHIBIT 2a and 2b: Remaining Work over Time (based on [Schwaber 2001]).

Execution

The PMBOK® Guide Executing processes focus on carrying out the project plan by performing the activities in it. Agile methodologies focus on developing the feature set promised to be delivered in the current iteration as opposed to adhering to a project plan. During the iteration the team is empowered to make all relevant decisions regarding coding and testing the software. The result of an iteration is a product increment.

Controlling

The PMBOK® Guide controlling processes ensure that project objectives are met through frequent monitoring and measurement of progress to identify variances from plan so that corrective action can be taken when necessary [PMI 2000, p. 30]. The scope is locked in at the start of the project and a rigid change management control process is used to evaluate requirements changes. PMBOK® Guide focuses on reducing normal operating costs, in particular the cost of changes. Therefore it recommends change control systems for scope, schedule, cost, and contract changes.

The Agile way is to prioritize the requirements and then define the scope per iteration. After each iteration, features can be added or removed (note: within the overall architectural abilities). Iterations are used as control devices to help make it easier to change course during the project, as well as provide some stability (changes are not made during iterations). This characteristic of Agile is based on the Agile philosophy that response to change is more important than following a plan. In case the team identifies that it takes more effort to develop the requested feature set (see time point 3 in exhibit 2b), the customer(s) can change priorities, add resources or remove functionality to address this change in workload. Agile methodologies focus on reducing the cost of change rather than reducing the amount of change. There is no need for change control processes since the customer usually sets priorities when changes occur.

Closing

In PMBOK® Guide the closing process focuses on completion and settlement of contracts as well as generating information to formalize phase or project completion. The product of the project is signed off and the project and product information are archived.

In comparison, Agile development places customer collaboration over contract negotiation. In many cases the customer is co-located with the development team and formal contracts play a less important role. The customer has the opportunity to review the product throughout the development process. Formal reviews take place after each iteration is completed. The customer formally accepts the product functionality developed and at the same time can correct future requirements or priorities for the next iteration. Documentation is seen as a requirement just like any other feature. As such it will need to be prioritized in the work queue. There is no standard formal closure process at the end of the project.

In conclusion, the biggest differences between PMBOK® Guide and Agile are in the planning and controlling phase. Even though there is overlap in the PMBOK® Guide process groups (see exhibit 3) the methodology is based on first defining scope, second getting customer signoff on the scope definition and subsequently managing tasks against the scope definition including a strict change control system. Agile on the other hand starts with a preliminary planning and continues planning after each iteration. Scope changes are possible before every iteration, but usually not during an iteration. Control is not based on scope but on remaining work hours. If the work needed is more than what is remaining, the customer is requested to drop features or add resources. Exhibit 3 and 4 show the two different approached per methodology.

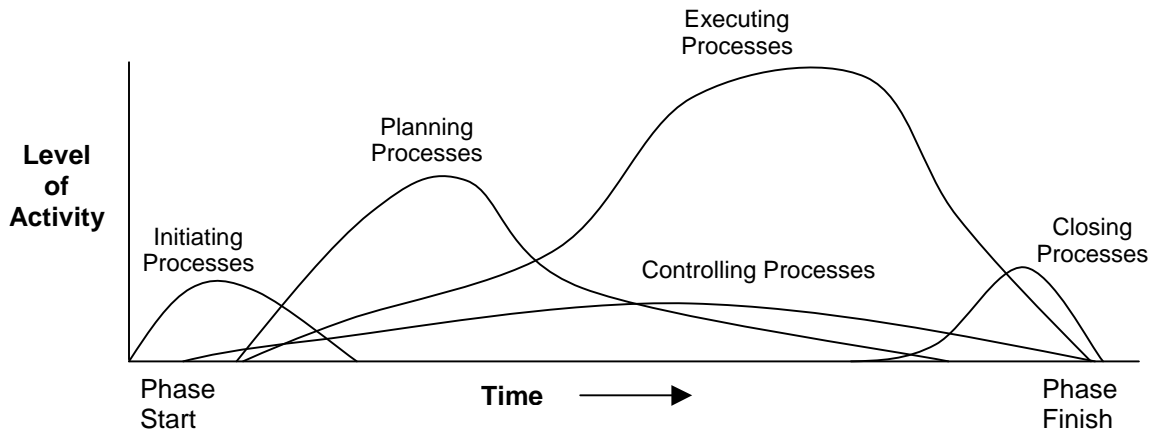


EXHIBIT 3: PMBOK® Guide process groups in a phase (based on [4])

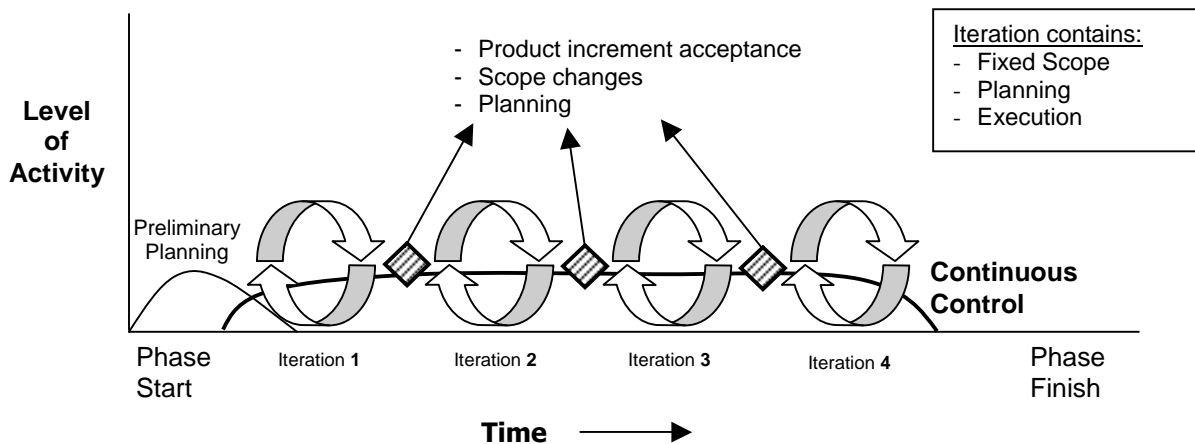


EXHIBIT 4: Agile process flow in a phase

PMBOK® GUIDE RELEVANCY TO AGILE DEVELOPMENT

There is a fundamental difference in the point of perspective between PMBOK® Guide and Agile. PMI focuses on optimizing the plan and reducing the normal operation costs, in particular the costs of making frequent changes. The Agile movement focuses on exploration. It requires reducing the cost of change rather than to reduce the amount of change itself. Also the perspective on how to view a project is different. The philosophy behind PMI's project management is that you can utilize best practices¹ to manage a project, using similar procedures and milestones. The philosophy behind the Agile methodologies is that the execution of each project is unique. It depends on the size of the team and how critical the application is. Developing a web site, for instance, is vastly different from writing a control system for an airplane. Although some processes can help guide a project, repeatability is thought of as

¹ Principles are applicable to other disciplines or industries. Practices are specific guidelines on what to do and by their nature not transferable. Practice according to the Concise Oxford Dictionary (1995) is a “habitual action or performance” were as a principle is “a fundamental truth as the basis of reasoning or action”.

ultimately futile. Development strategies and practices need to reflect the uniqueness of projects. Agile focuses on translating principles rather than utilizing best practices.

Agile software development practices seek to eliminate waste. Major areas in software development that are waste and do not add value to the customer are partially done work, extra processes, documentation, and features not requested by the customer [Poppendieck 2003]. The practice of eliminating waste also eliminates much of the relevancy of the PMBOK® Guide knowledge areas to Agile development (see exhibit 5).

Agile development also seeks to leverage cooperation over negotiation. For example, today’s majority of contracts are either fixed price or time-and-material. These contract types contain hidden costs, like transaction and control costs, and lead to defensive behavior either from the customer or from the vendor. Agile’s approach put trusts over legal language. Contract types that fit with this philosophy are multistage contracts or target-cost contracts. Multi-stage contracts are simply stated fixed-price contracts per short cycle. In target-cost contracts the responsibility is with both the customer and the vendor. If the target cost is exceeded, both parties will pay more, and if it is under target cost, both parties will share in the benefits. This encourages cooperation.

Knowledge Areas	PMBOK® Guide	Agile
Integration	Ensures that various elements of the project are properly coordinated.	Need for formal coordination is limited due to reduced use of processes in various elements.
Scope	Ensures that project includes only the work required to complete the project successfully. Focuses on defining and controlling what is or is not in the project Uses rigid change management processes.	Scope is only fixed when iterations are in progress. No formal scope control is needed.
Time	Focuses on defining activities for project schedule and schedule control to ensure timely project completion.	Time per iteration is fixed. Focus is on delivering value (features) as fast as possible. Overall schedule is based on features instead of activities.
Cost	Determines budget based on resources needed for project and ensures that the project is completed within the approved budget.	Determines budget based on functionality requested. Resources, functionality and timing are balanced throughout. Cost of delay is quantified.
Quality	Ensures that the project will satisfy the need for which it was undertaken. Focus is on conformance to requirements.	Project success criteria are set by the customers, which also deliver feedback after iterations. Focus is realization of purpose or fitness for use.
Human Resource	Processes to make most effective use of the people involved with the project.	Focus on team and not individual. Incentives are based on group productivity: e.g. jointly recognize tester and developers for a low defect ratio.
Communications	Ensures timely and appropriate generation, collection, dissemination, storage, and ultimate disposition of project information.	Focus on eliminating waste. No unnecessary features, paperwork or documentation. Open access to information for all involved.
Risk	Focuses on identifying, analyzing and responding to project risk.	The Agile methodologies treat risk management differently. There is not one common Agile approach.
Procurement	Focuses on acquiring goods and services to attain project scope from outside the performing organization. Based on using contracts, best practices, processes & procedures.	Follows best principles for acquiring goods and services by putting collaborations over contracts.

EXHIBIT 5: Agile compared per PMBOK® Guide knowledge area

CONCLUSION – TO BE AGILE OR NOT TO BE AGILE

According to the Agile movement, software development is not intended to produce repeatable results. It is not a “get it right the first time” effort. There are a too many unknowns at the start of software projects to be able to fix the scope and plan the project to completion. New business processes, development languages or architectures are used that have never been used before. Most projects are attempting something that has never been done before, using the latest tools and architectural approaches. The effects of implementation are unclear due to integrations with old customized legacy systems. In this situation, rapid development, test and fix cycles will produce a better product than focusing on perfecting the design and then performing the work. Many short development cycles maximize the learning potential for the organization and the development team. In Agile an important integration point is that customers have many feedback opportunities.

The core of Agile is to embrace change unlike the PMBOK® Guide methodology. Many companies view change as a necessary evil rather than as a competitive opportunity. To succeed companies need to change their core business principles and potentially they way they are conducting business. This means that Agile methodologies could change how software businesses are conducting business. Bigger corporations might choose to apply Agile just for prototype projects only and fund each iteration separately. It could provide them the capability to cancel investments made in R&D prototype projects if they don’t show any progress before investing in a multiple year endeavor.

The authors believe that the Agile principles will change the way we manage software projects. In a world where 74 percent of the IT projects fail or are challenged [Standish Group 1994], Agile software development is a promising solution. One of the main reasons of failure is lack of customer involvement and Agile’s project collaboration approach combined with the short development and feedback cycles addresses the issue.

Don’t throw the PMBOK® Guide out of the window, but combine the knowledge and experience with the new Agile approaches. Companies need to let go of the rigid control systems. The project manager’s role needs to change from “police officer” controlling the “law” to managing relationships and empowering the team. He or she needs to create an environment for open communication, with faster responses to market changes and new demands.

REFERENCES

- Economist (2001, Sep 20), Agility Counts, *The Economist*, retrieved on Aug 4, 2003 from http://www.economist.com/displaystory.cfm?story_id=S%26%28%280%25QQ3%2A%0A&CFID=12472588&CFTOKEN=48a17b8-38dd774f-9804-4fde-8944-e3ed8ca36fd7.
- Highsmith, J. (2002), *Agile Software Development Ecosystems*. Boston, MA: Addison Wesley.
- Poppendieck, M. & Poppendieck, T. (2003), *Lean Software Development: An Agile Toolkit for Software Development Managers*. Boston, MA: Addison Wesley.
- Project Management Institute. (2000) *A guide to the project management body of knowledge (PMBOK® Guide)* (2000 e.). Newtown Square, PA: Project Management Institute.
- Schwaber, K., Beedle, M. & Martin, R.C. (2001), *Agile Software Development with SCRUM*. Upper Saddle River, NJ: Prentice Hall.
- Standish Group (1994), The CHAOS Report, 1-2.
- Thomsett, E. (2001). Extreme Project Management. *E-Project Management Advisory Service Executive Report*, 2 (2), 2-6.