Software Development with Agile Approach

By Rohit Sinha, PMP

Agile is no longer a buzz word, and the immediate returns are what is making it popular. The basic philosophy of the agile approach is to accommodate changes. We know that ideas start pouring in when we actually see the running product. As many people say, one of the main pitfalls with the traditional waterfall model has been that managing changes (i.e., thinking everything through upfront) has always been a challenge. This article briefly talks about agile methodology and its characteristics and provides details about managing software projects with agile practices.

History

In a traditional waterfall model, the project flows from top to bottom just like a waterfall (i.e., the team need to follow a sequence). Requirements, design, implementation, testing, deployment, and maintenance are the steps to be followed one after another (Figure 1). The team starts with requirements, moves on to the design phase, and comes up with a detailed technical design, and so on. This model has many constraints and one of them is not to keep up with changing requirements. Freezing requirements before the start of implementation has been a continuous challenge. Intensive documentation is one of the key aspects of this model.

There has been a great deal of brainstorming on various alternative practical approaches to the current needs. The Agile Manifesto (drafted in February 2001) states: "*We are uncovering better ways of developing software by doing it [Agile] and helping others do it. Through this work we have come to value:*

- *Individuals and interactions over processes and tools*
- *Working software over comprehensive documentation*
- *Customer collaboration over contract negotiation*
- *Responding to change over following a plan*"

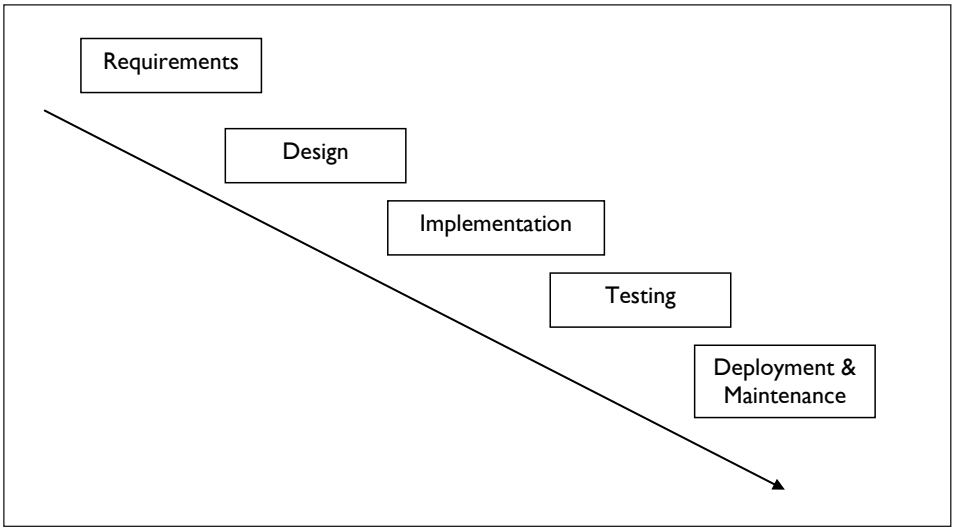

Figure 1: Traditional waterfall model.

Methodology

Agile is an adaptive approach that is based on the philosophy that changes are inevitable. It advocates a short delivery cycle, just-in-time analysis, close collaboration, and high visibility. Projects are divided into small time boxes known as iteration. Iteration is divided into scrums and sprints. A single iteration generally spans for two to four weeks and is usually completed with a delivery. Generally, first iteration is used for preliminary scoping, plan, and initial design. Hands-on development is done in the subsequent iterations. After completion of one development iteration, demos can be shown and feedback collected. Any changes needed in the working software are implemented in subsequent iterations. One of the benefits of this model is that needed modifications are incorporated in the software with no last-minute surprises.

> "It is not necessary to change. Survival is not mandatory."
>
> W. Edwards Deming

Figure 2 depicts a high-level view of overall agile process with its core ingredients.

The details of the processes shown in Figure 2 are discussed in subsequent sections of this article. This approach has following characteristics:

Scrums

Scrum is a software development process framework containing practices and predefined roles that enables creation of self-organizing teams. Scrum recognizes changes and focuses on dealing with emerging requirements. The main roles in scrum teams are scrum master, product owner, and team.

- *Scrum master* is the role held by a project manager. This person is a coordinator and maintains the processes. He or she is the one who facilitates scrum processes and coordinates with the product owner and the development team.
- *Product owner* is a key stakeholder representing the end user and serves as a proxy customer to the team. He or she is the one who prioritizes the requirements. The product owner answers team questions and provides directions to the team. In my view, some valuable traits for the product owner to have are good communication skills, willingness to go deeper into understanding the product and its market value, good UI (user interface) skills, and some technical background (helpful in communicating with development team).

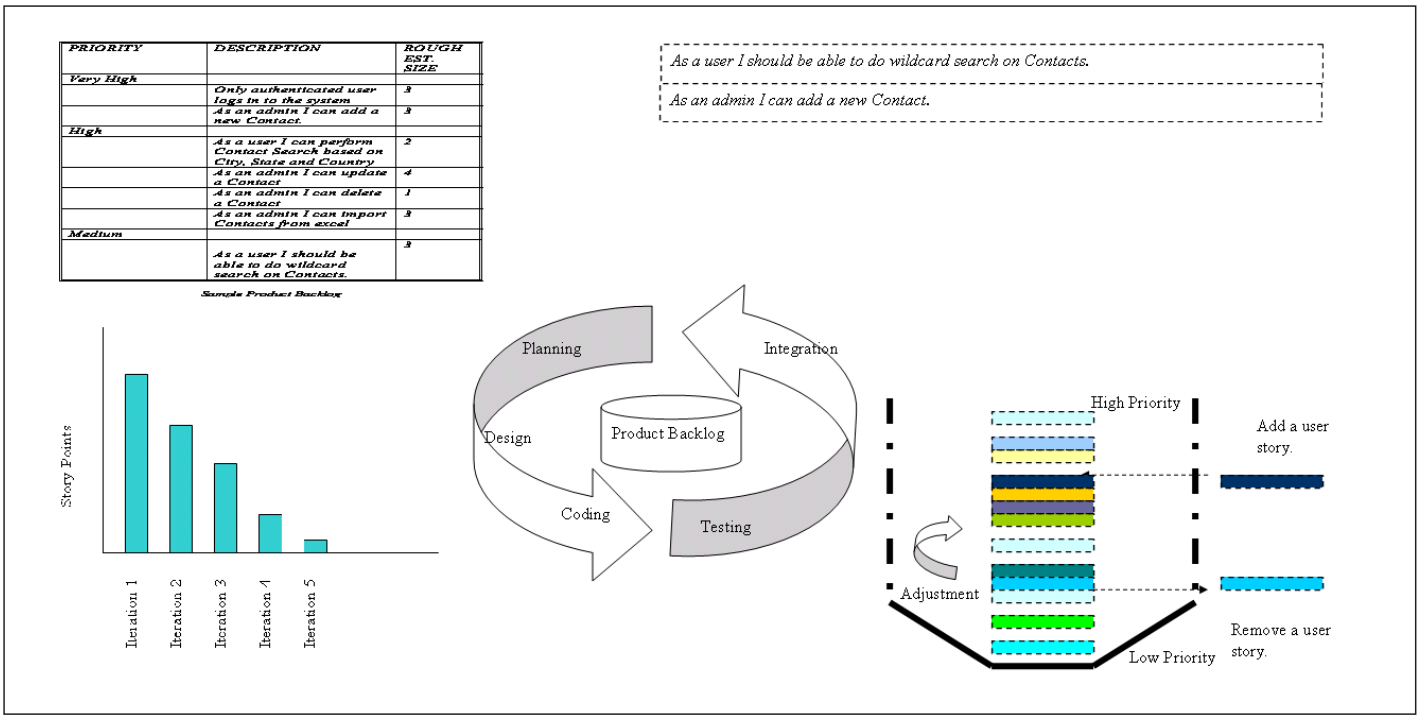

Figure 2: Typical agile development in-progress.

- *Team* is a cross-functional group of 5 to 9 members doing analysis, design, implementation, and testing.

Sprint

Sprint, which has a duration of two to four weeks, contains in-progress shippable product containing a list of user stories taken from the product log. Each day in a sprint, a daily standup occurs, also known as daily scrum. This helps in knowing the current status and in resolving any roadblocks in the project. During the meeting, each team member answers three questions:

- What have you done since yesterday?
- What are you planning to do today?
- Do you have any problems stopping you from finishing your goal?

Scrum of Scrums

The term *scrum of scrums* normally applies to a project consisting of multiple teams, as it helps in resolving many cross-team issues and dependencies. It involves representatives of each team, and normally the scrum master represents his or her team in this meeting. Generally this happens each day after daily scrum.

User Stories

In agile world, requirements are expressed as a user story. This is a quick way of representing requirements without going into detail. It is written in one or two sentences in a business language. Each feature of the application is represented by a user story. Shown below are two examples:

As a user I should be able to do wildcard search on Contacts.

As an admin I can add a new Contact.

In my personal experience, this is a great way of maintaining a product log. The details are looked into as you come closer to delivering a story. I also found that a high-level illustration of requirements gives the development team needed flexibility to think more, and sometimes they surprise you with their innovation.

Story Points

User stories need to be estimated, and estimation is done in story points. A story point is a number that tells how easy or difficult it is to implement a story. It represents the size of a user story relative to other stories. For example, a story is given two points if you think it is twice as big as that of a one-point story and about half as big as a four-point story. This method of estimation is more practical, and therefore the total estimated effort is appropriate.

Product Backlog

The product backlog, which is the heart of scrum, is a prioritized list of user stories. The product owner normally collaborates with a customer and comes up with backlog of items. Its characteristics are as follows:

- It is dynamic and continuously changed based on changes in the requirements.
- Every item in the backlog is attached with a priority (Table 1).
- Typically it doesn't contain the detailed requirement. Usually details are looked into at the time of implementation.

PRIORITY	Description	Rough Est. Size
Very High		
	Only authenticated user logs in to the system	3
	As an admin I can add a new Contact.	3
High		
	As a user I can perform Contact Search based on City, State, and Country	2
	As an admin I can update a Contact	4
	As an admin I can delete a Contact	1
	As an admin I can import Contacts from Excel	3
Medium		
	As a user I should be able to do wildcard search on Contacts.	3

Table 1: Sample product backlog.

- Items higher in the list usually are more detailed compared with lower items, for which requirements are generally vague.

Managing Agile Projects

As mentioned in the Agile Manifesto, agile projects recommend face-to-face communication over communication via documents, especially if the team is co-located. In the case of teams with members who are at different locations (geographically dispersed teams), video conferencing, skype, wiki, and e-mails may substitute for in-person interactions. In my experience, communication in this manner in geographically dispersed teams works well. Also, with agile projects, stakeholders are involved from the beginning of the process; I have found this to be very helpful, as it allows issues to be discovered early on and helps build a customer-friendly product. Examples are used to describe the application features or unit of code. Working software is the measure of progress. Software practices like test-driven development, continuous integration, code refactoring, and small releases improve the overall efficiency of the project.

Iteration Duration

A typical agile iteration lasts for two weeks, as one week is too short and one month can be too long. Two weeks are sufficient to get some meaningful work done as there are some overheads in iteration planning and closure. However, sometimes for bigger teams (8 to 9 development members), it makes sense for iteration to have a longer duration for showing progress. In my opinion, a general rule of thumb is two weeks for iteration, but in case a project or team size is bigger, you may consider a four-week cycle.

Iteration Planning

The primary objective of planning is to define the scope of the iteration. On the basis of the current project and business situation, features are added or removed from the backlog. Iteration planning should not be more than two to four hours. This is done just before the start of iteration. The highest-priority features are selected; there may also be a few lower-priority items if they suit the overall goal of the iteration. These features are then broken down into tasks. Each task normally takes approximately four hours to two days. If a task needs more than two days, it further breaks down into subtasks. During the iteration plan, if it is discovered that the features will be completed ahead of time, the product owner adjusts it by adding more features. Similarly, if iteration time is not sufficient for all of the features, the product owner adjusts the iteration by taking away some features and assigning it to different iterations.

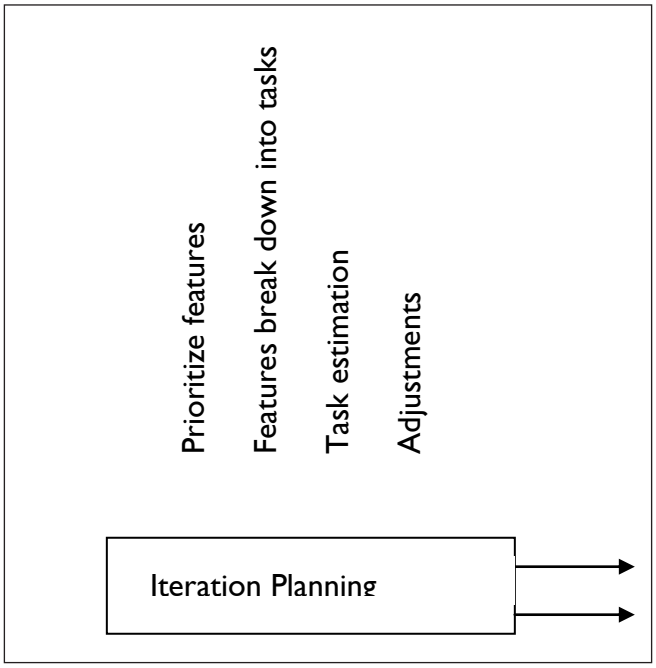

Figure 3: Iteration planning.

Development

Most agile implementations have daily face-to-face communication among team members. This also includes the product owner as customer representative and some stakeholders as observers. Here, working software is the primary measure of progress. When a team works in different locations, they maintain daily contact through videoconferencing, voice, e-mail, etc. Automated build generation and automated testing are generally part of development and help in maximizing the efficiency of the team. I have found that it is worth spending some time initially to put these processes in place. It makes everybody's life smooth and the overall development efficient. Projects are quality focused, which also includes code quality. Code refactoring and testing are attributes of quality. The practice of code refactoring is where you make simple changes to the code that improve quality without changing the semantics. The team prefers to test often and early, and the more disciplined ones even take a test-driven approach where they write a single test and just enough code to fulfill that test, and then reiterate. Figure 4 shows a typical iteration cycle:

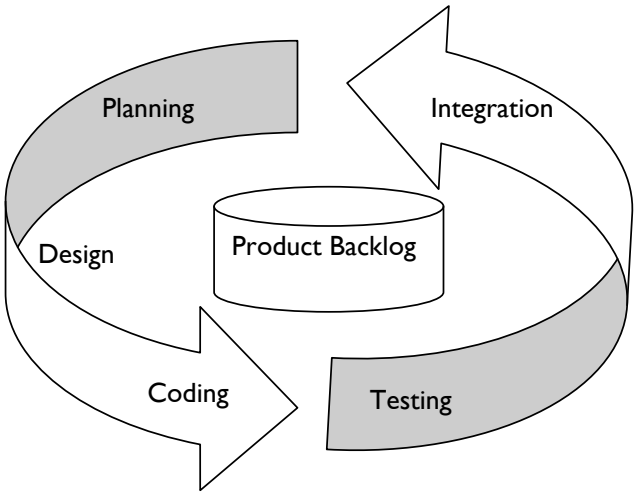

Figure 4: Agile development cycle revolves around product backlog.

Tracking and Adjustment

The project progress is tracked in iteration, and adjustments are made (Figure 5). As a project progresses, stakeholders get a better understanding by seeing the working software, and their needs can be changed. In addition, changes in the business situation or changes in priority at the organization level are also common factors behind these changes. The product log gets updated based on current needs, and subsequent iterations are replanned.

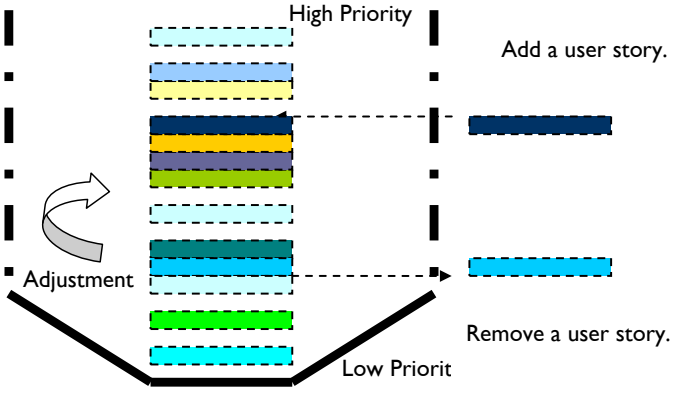

Figure 5: A product backlog jar depicts change management.

One of the tools that help in tracking the progress is the burn down chart. It is a graphical representation of work versus time. Generally vertical axis represents the amount of work left and horizontal axis represents time. It is one of the preferred ways of showing progress in a sprint. In Figure 6, a release burn down chart is depicted where work is shown as story points and time is shown in iterations:

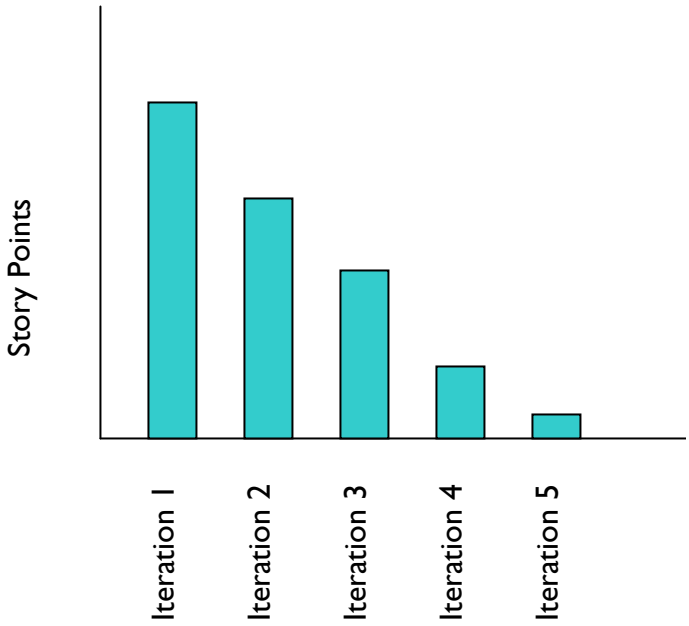

Figure 6: Sample burn down chart.

Concerns

Some people have doubts about this approach. Some of the common questions and concerns are: "Is it necessary to adopt?" "It doesn't look organized and or easy to sell this process to the clients," "What will happen to QA?" and "It sounds too risky." These are all valid points to look into before this approach is adopted. Also, agile is not necessarily always be a good fit for a project. Other development approaches are worth considering if the following are true of a particular project:

1. Team size is big.
2. A short iteration cycle is not possible.
3. Requirements are clearly stated and detailed out.
4. Requirements do not change.
5. Stakeholders are not available during development.
6. Intensive documentation is needed and key to success.

In real-world situations, the above points do not apply most of the time, and therefore agile is a good solution.

Conclusion

Although agile practices still have some concerns, its benefits—such as accommodating changing customer

requirements, immediate returns, close team coordination, and flexibility—facilitates its rapid acceptability to those involved in software development. The philosophy of "people over process" is making it more successful with each passing day. It is a well-thought-out approach of successfully executing projects. In today's fast-paced world, many companies are quickly adopting agile practices and becoming successful.

About the Author

Rohit Sinha, PMP, a product manager with 11 years of experience in IT projects, is an innovative professional with an outstanding background in leadership and a proven ability to identify, analyze, and solve problems to increase customer satisfaction and controls costs. He is responsible for both the execution and management of multiple projects, and focuses on building the trust and relationship that are needed to keep projects moving. He has domain expertise in insurance, accounting, and document management. He is a project management professional (PMP)® credential holder and holds master's degree in computer science. He can be contacted at rohit_s25@hotmail.com.