Wharton
Aresty Institute of Executive Education
University *of* Pennsylvania
EXECUTIVE EDUCATION

Product Management and Strategy

Module 5: Agile+

Quick Reference Guide

Learning Outcomes

1. Discuss what problem Agile solves.
2. Define how Agile processes work.
3. Evaluate limits of Agile for integral architecture.
4. Evaluate the current trend of application of Agile outside of software.
5. Evaluate the use of Agile in any setting.

Introduction to Agile

Waterfall process

The default zero-to-one process used in product management is the waterfall process. This is simplified into four steps:

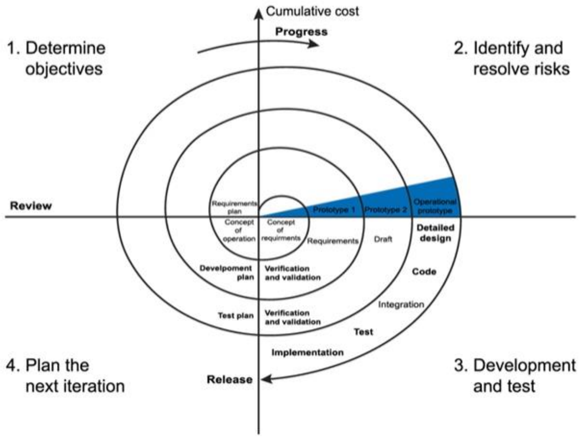
a. Product requirements (PRD): The PRD includes a list of customer needs and a set of target performance specifications.
b. System-level design: This phase develops a concept and an overall approach to the solution.
c. Detailed design: This phase resolves all designed details.
d. Testing: This phase comprises both internal and external testing.

Frederick Brooks' observation on software projects:

- They were delivered late
- They often missed customer requirements
- They required additional developer resources, which delayed projects even further

The solution to these observations was to create the smallest possible team composed of the highest-quality programmers to minimize coordination costs.

Boehm's spiral model of product development



In Boehm's model, the four basic stages in the waterfall model are followed but in rapid iterative cycles, in which increasingly more complexity is added to the product. For instance, in a first cycle, the product might have 10% of its eventual functionality. Then over the next nine cycles, the remaining 90% would be built. It's called a spiral because the process loops around through all phases of the waterfall process, but with scope increasing on each loop represented by increasing radius of the spiral.

Some Agile manifesto principles:

- Early delivery of the product: Satisfy the customer through early and continuous delivery of valuable software.
- Working software: This is the primary measure of progress.
- Frequent delivery: Deliver working software frequently with a preference for a shorter timescale.
- Adapt to change: Welcome changing requirements even late in development. Agile processes harness change for the customer’s competitive advantage.
- Business and developer cooperation: Businesspeople and developers must work together daily throughout the project.
- Maintain a constant pace: Agile processes promote sustainable development; the team should be able to maintain a constant pace indefinitely.
- Regular reflection and adjustment: At regular intervals, the team reflects on how to become more effective then tunes and adjusts its behavior accordingly.

The Agile Process

Scrum variant of Agile

In Agile, the overall solution for the job to be done is divided into small chunks which are called user stories. Given below are some Agile terminology:

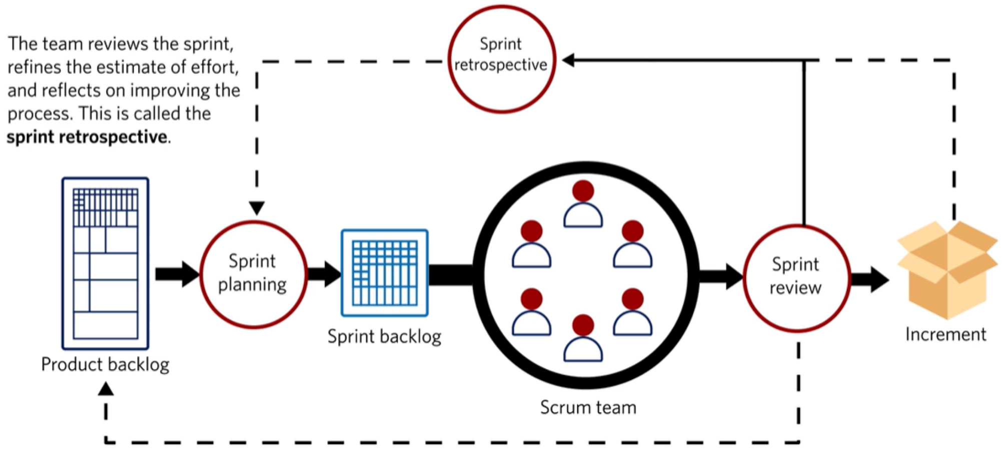
- Epic: The overall job to be done as an epic
- User stories: The sub-tasks within the overall job.
  - The product manager is responsible for the prioritization of user stories, they typically consult with other stakeholders in forming a judgment.

- Backlog: A list of user stories. The development team then estimates the work required to complete each story.
- Story point:
  - It is a unit of work
  - There is no predefined conversion factor to estimate person-hours
  - It is roughly a person-hour effort

Planning poker/scrum poker

Team members hold up numbered cards to indicate their individual on the work required.

Scrum framework


The team reviews the sprint, refines the estimate of effort, and reflects on improving the process. This is called the **sprint retrospective**.
Sprint retrospective
Product backlog
Sprint planning
Sprint backlog
Scrum team
Sprint review
Increment

The scrum framework involves the following steps:

1. Product backlog: This is a prioritized list of stories with an estimate of effort for each story.
2. Sprint planning: The product manager and engineering manager agree on which stories will be taken from the product backlog at this stage.
3. Sprint backlog: These stories become the focus of the next calendar period of work, this is called a sprint. Sprints are usually two weeks long.
4. Scrum team: The development team works only on those selected user stories and attempts to complete them within the sprint interval.
5. Sprint review: The team reviews the sprint.
6. Increment: At this stage, the team refines their estimate of how many story points can be completed in a sprint interval.
7. Sprint retrospective: The team reflects on how the entire process might be improved.

Using the Agile methodology, the team takes on finite tasks for finite periods of calendar time with the expectation that the work assigned to the sprint interval will be completed.

What is an epic?

In Agile methodology, larger tasks need to be broken into smaller pieces such that each piece can be completed within a sprint interval. If a story is too large, it's called an epic, and it is divided into smaller stories. A new version of the product is released for testing at the end of each sprint.

Benefits of Agile

- Predictable:
  - The organization is guaranteed to have a completed working solution at the end of each sprint interval.
  - That solution grows incrementally in complexity and completeness with each subsequent sprint interval.
- Flexible: It provides a flexible solution scope irrespective of the schedule (unlike the waterfall methodology)
- Collaborative: The method forces constant reconciliation of capacity to do work with the priorities of the product manager. Every two weeks, the entire team faces their actual performance relative to their estimates and forecast.
- Software-oriented:
  - It makes use of a productivity software which allows the team to track the backlog, update priorities and time estimates, and monitor productivity.
  - It's also the means of communication for assignments to individual team members.

The Kanban Version of Agile

The word “Kanban” comes from:

- The Toyota production system
- The just-in-time (JIT) approach to materials management

The actual word “Kanban” refers to a card or board that is used to represent inventory levels. The Kanban system in the Toyota production system ensures:

- The inventory level in a system never exceeds a prespecified level.
- New material is only pulled into the system when existing material is used.

Kanban system can be used in different approaches such as inventory management and coding.

Kanban system in inventory management

- For each size and thickness of material, a plastic card is placed in the stack of finished goods.
- This card indicates the item and amount of item to be produced.
- When sufficient material is used to expose the card, it is taken and dropped in a box where production happens.
- The production team treats the card as an order to produce the required item.
- Once the production is completed, the team places them on the finished goods stack and resets the card. In this way, production exactly tracks the consumption of finished goods.

Kanban system in coding

- Each different type of development resource is assigned its own queue of tasks. These might correspond to a single individual such as a UI/UX specialist, or to a small team such as database programming.
- For each resource, there's a maximum number of tasks in the queue. This means that, for example, the UI/UX specialist has a small and specific number of tasks to focus on.
- Each task is categorized according to the type of resource that it requires for completion. The product manager prioritizes the items on the master backlog.
- Once a resource completes a task on its queue, a slot opens, and the next item is taken off the backlog and put into the queue for the resource.

The analogy to inventory management is that the queue of work for each resource is like the inventory level. When the queue or inventory level drops below a pre-specified level, then a signal like the Kanban card is sent to the product manager to release a new chunk of work into the queue.

Kanban differs from scrum in two ways:

1. Kanban allows the development team to be divided into non-fungible types of resources to handle different types of tasks
2. Kanban allows the team to work continuously rather than in a predefined sprint interval. Whenever a slot opens, a new story is released into the development queue for that resource.

Kanban with software tools

In the Kanban version of agile, the team typically benefits from the use of a software tool for implementation, particularly if the team is working remotely where more literal visual system would be in impractical. Jira is one such tool that includes the Kanban functionality, as part of the core suite of features for managing Agile projects.

The Limits of Agile

The two limitations of Agile relates to the following:

1. The difference between the initial system-level design and subsequent development and improvement of that system.
2. The difference between development of a modular system and the development of an integral system.

Limitations of Agile

1. Initial system-level design vs. subsequent development and improvement of the system
   - The development team informally creates an early prototype. Here, an Agile mindset can be considered by:
     - Prioritizing some features in the first version
     - Incrementally improving that version
   - The team then uses the prototype to get feedback from the market and to demonstrate it to other stakeholders

- When the initial prototype is validated, the technical team leader makes design decisions such as architectural choices of data structures, platforms, and the use of preexisting building blocks

Conclusion:

- The initial architectural design is not very amenable to an Agile process
- Preferably, it is carried out by a small technical team
- It consumes less time; therefore, it is rare for development teams

2. Development of a modular system vs. an integral system

   Given below is the difference between hardware products and software products:

   Hardware products:

   - They are highly integral.
   - This allows designers to reduce size and weight, improve performance and to lower costs. For software, there are no such benefits to integrality. And so most software systems are highly modular, separate chunks of code control each distinct function of the system.

   Software products

   - They are highly modular
   - Separate chunks of code control each distinct function of the system

   Development of complex hardware using Agile

   - Complex hardware projects are divided into discrete prototype phases such as proof of concept prototype, alpha prototype, beta prototype, and pre-production prototype.
   - These discrete prototypes serve as integrating milestones for all the separate development efforts.
   - Although this type of process is necessary for most hardware, they result in highly uncertain schedules.

   Agile methodologies work very well with modular systems.

Agile for X

Agile process for an individual

- Create a backlog of tasks. For example: to-do list, a personal productivity app, etc.
- Estimate the effort required, such as in terms of number of hours.
- Establish a priority order for the tasks.
- Maintain a sprint interval. For example, a day or a week.
- Plan the sprint interval. Here, you identify a set of the highest priority tasks whose total effort does not exceed the available time for the week.

In the scrum approach, all tasks are assigned to teams and thus, the resources of the team are fungible across tasks.

In the Kanban approach, different tasks are assigned to specialized team members as they complete their tasks.

Agile methodology: Strengths

Factors that influence managers to apply Agile approach for nontechnical tasks:

- Timely and predictable completion of tasks
- Prioritization
- Realistic reconciliation of supply and demand

Agile methodology: Weaknesses

- Disregards unexpected demands
- Discounts multitasking