Agile Project Management

CS413 - Software Engineering Project Management

Department of Computer Engineering, Bilkent University

Dr. Mustafa Değerli

Bilkent University

Definable Work vs. High-Uncertainty Work

- Two ends of project work
- Definable work projects are characterized by clear procedures that have proved successful on similar projects in the past
- New design, problem solving, and not-done-before work is exploratory
- High-uncertainty projects have high rates of change, complexity, and risk

Principles Behind the Agile Manifesto

- Thought leaders in the software industry formalized the agile movement in 2001 with the publication of the Manifesto for Agile Software Development

Values of the Agile Manifesto

We are uncovering better ways of developing software by doing it 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

That is, while there is value in the items on the right, we value the items on the left more.

Principles Behind the Agile Manifesto

- 1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- 2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Principles Behind the Agile Manifesto

- 3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- 4. Business people and developers must work together daily throughout the project.

Principles Behind the Agile Manifesto

- 5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- 6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Principles Behind the Agile Manifesto

- 7. Working software is the primary measure of progress.
- 8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

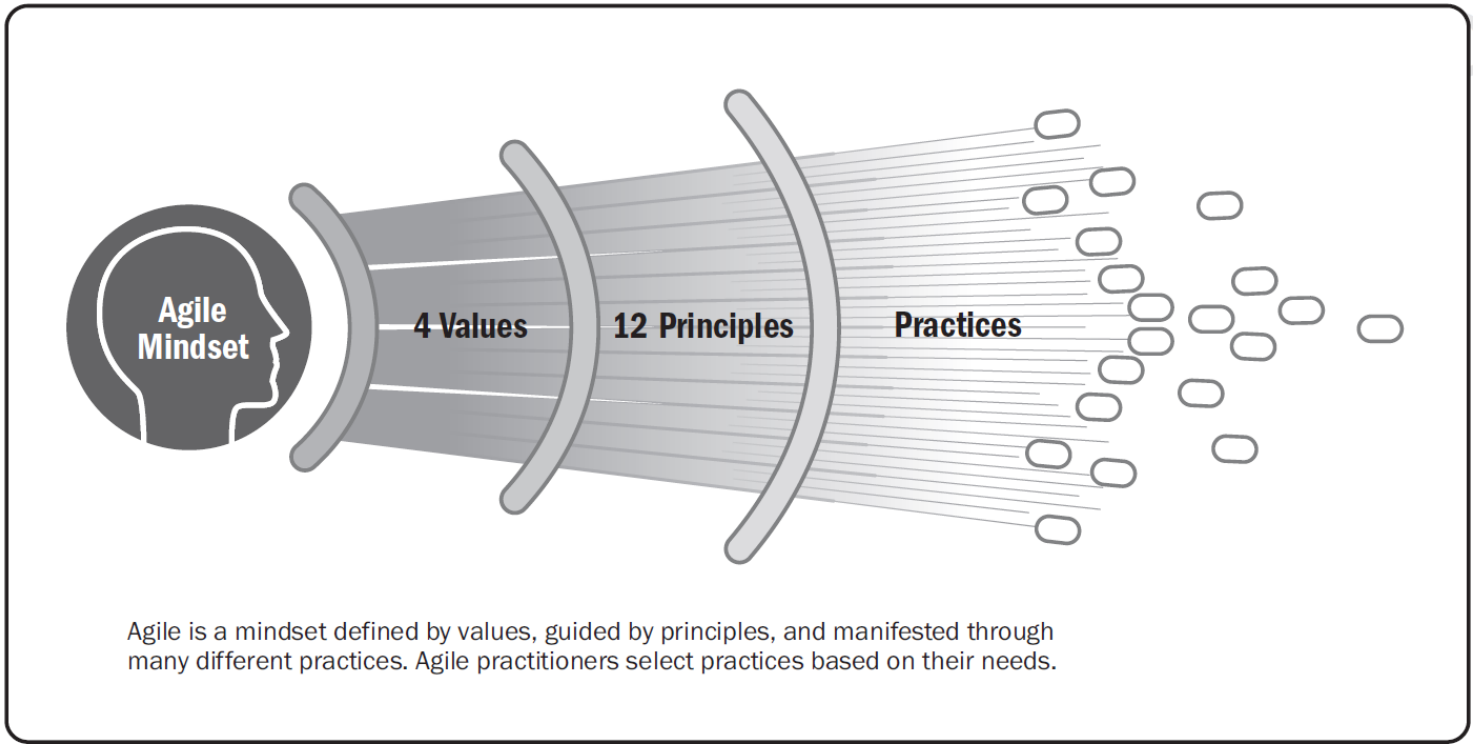
Principles Behind the Agile Manifesto

- 9. Continuous attention to technical excellence and good design enhances agility.
- 10. Simplicity—the art of maximizing the amount of work not done—is essential.

Principles Behind the Agile Manifesto

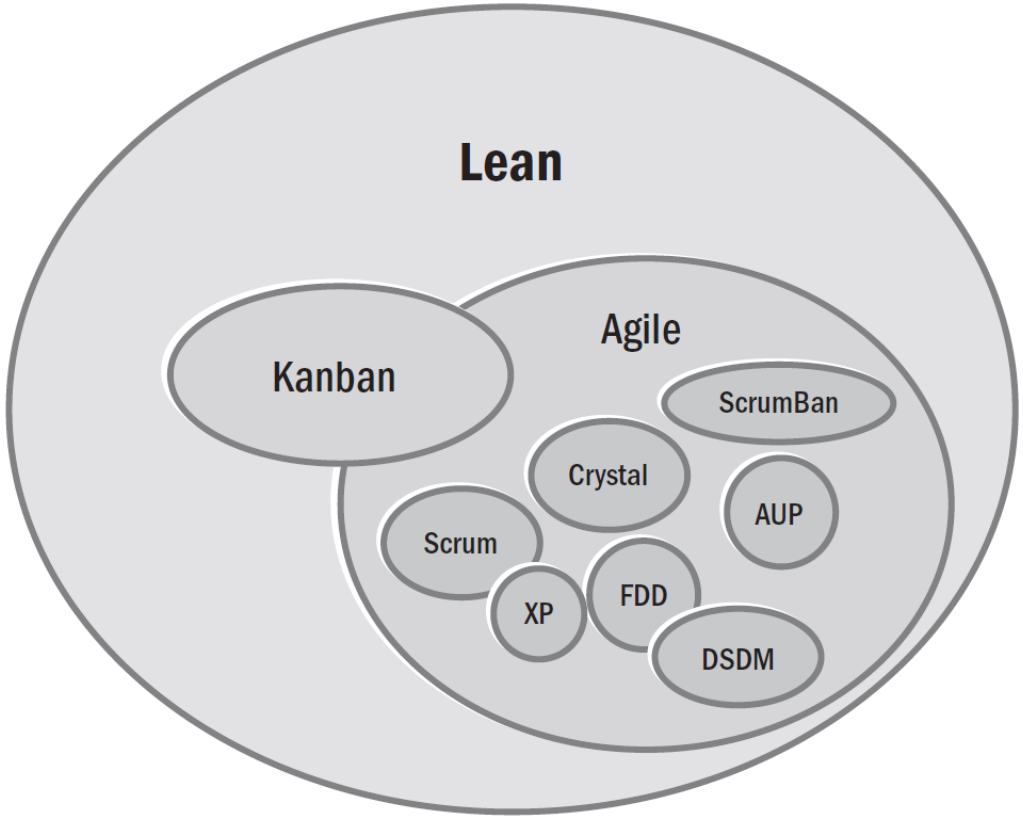
- 11. The best architectures, requirements, and designs emerge from self-organizing teams.
- 12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Agile Approach

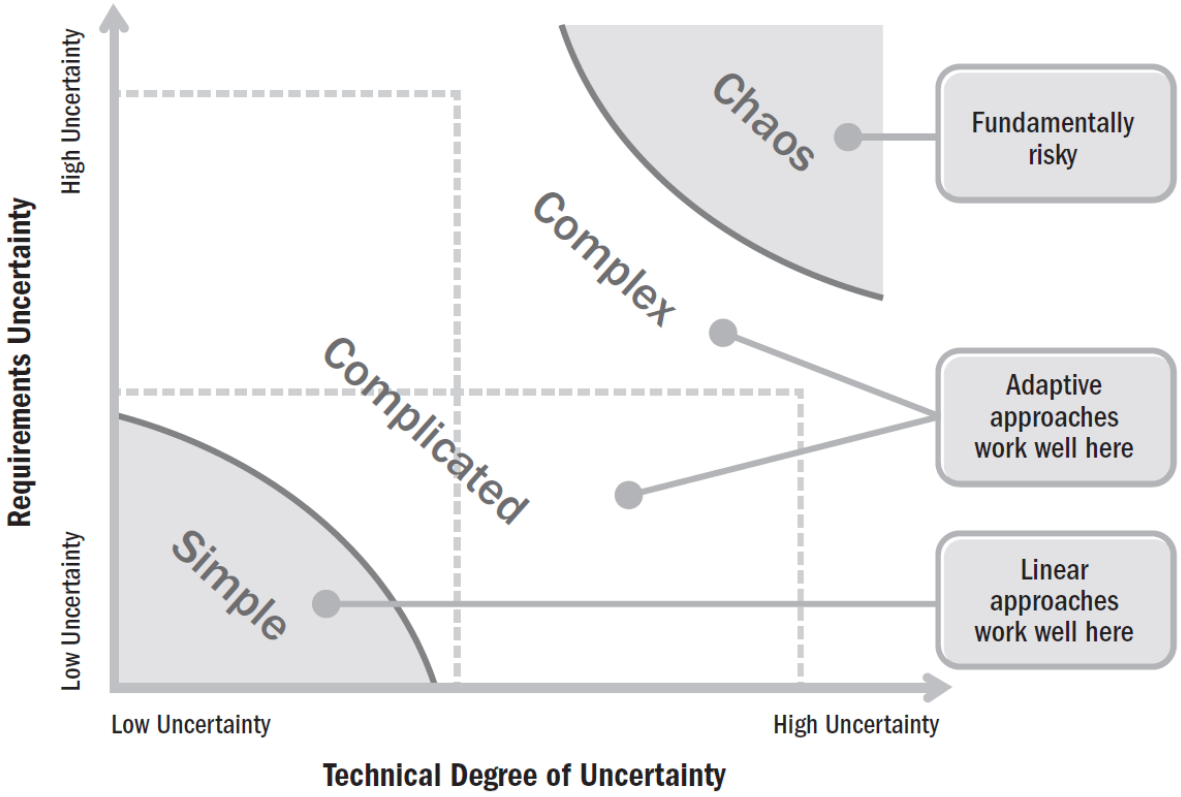


Agile is a mindset defined by values, guided by principles, and manifested through many different practices. Agile practitioners select practices based on their needs.

Agile Approaches



Uncertainty and Complexity Model



Life Cycles

- **Predictive life cycle.** A more traditional approach, with the bulk of planning occurring upfront, then executing in a single pass; a sequential process
- **Iterative life cycle.** An approach that allows feedback for unfinished work to improve and modify that work

Life Cycles

- **Incremental life cycle**. An approach that provides finished deliverables that the customer may be able to use immediately
- **Agile life cycle**. An approach that is both iterative and incremental to refine work items and deliver

Agile Teams

- Focus on rapid product development so they can obtain feedback
- The most effective agile teams tend to range in size from 3 to 9 members
- Ideally, agile teams are colocated in a team space
- Team members are 100% dedicated to the teams

Agile Teams

- Agile encourages self-managing teams, where team members decide who will perform the work within the next period's defined scope
- Agile teams thrive with servant leadership. The leaders support the teams' approach to their work

Agile Roles

- Three common roles:
  - Cross-functional team members
  - Product owner
  - Team facilitator

Agile Roles

Cross-functional team member	Cross-functional teams consist of team members with all the skills necessary to produce a working product. In software development, cross-functional teams are typically comprised of designers, developers, testers, and any other required roles. The cross-functional development teams consist of professionals who deliver potentially releasable product on a regular cadence. Cross-functional teams are critical because they can deliver finished work in the shortest possible time, with higher quality, without external dependencies.

Agile Roles

Product owner	The product owner is responsible for guiding the direction of the product. Product owners rank the work based on its business value. Product owners work with their teams daily by providing product feedback and setting direction on the next piece of functionality to be developed/delivered. That means the work is small, often small enough to be described on one index card. The product owner works with stakeholders, customers, and the teams to define the product direction. Typically, product owners have a business background and bring deep subject matter expertise to the decisions. Sometimes, the product owner requests help from people with deep domain expertise, such as architects, or deep customer expertise, such as product managers. Product owners need training on how to organize and manage the flow of work through the team. In agile, the product owners create the backlog for and with the team. The backlog helps the teams see how to deliver the highest value without creating waste. A critical success factor for agile teams is strong product ownership. Without attention to the highest value for the customer, the agile team may create features that are not appreciated, or otherwise insufficiently valuable, therefore wasting effort.

Agile Roles

Team facilitator	The third role typically seen on agile teams is of a team facilitator, a servant leader. This role may be called a project manager, scrum master, project team lead, team coach, or team facilitator. All agile teams need servant leadership on the team. People need time to build their servant leadership skills of facilitation, coaching, and impediment removal. Initially, many organizations invite external agile coaches to help them when their internal coaching capability is not yet fully developed. External coaches have the advantage of experience, but the disadvantage of weak relationships in the client organization. Internal coaches, on the other hand, have strong relationships in their organization, but may lack the breadth of experience that would make them highly effective.

Where is PM in agile?

- The role of the project manager in an agile project is somewhat of an unknown, because many agile frameworks and approaches do not address the role of the project manager
- Some agile practitioners think a project manager is not needed, due to self-organizing teams taking on the former responsibilities of the project manager

Where is PM in agile?

- Pragmatic agile practitioners and organizations realize that project managers can add significant value in many situations
- The value of project managers is not in their position, but in their ability to make everyone else better

Scrum

- A single-team process framework used to manage product development
- Consists of Scrum roles, events, artifacts, and rules, and uses an iterative approach to deliver working product
- Run on timeboxes of 1 month or less with consistent durations called sprints where a potential a potentially releasable increment of product is produced

Scrum

- Scrum roles
  - The product owner
    - Responsible for maximizing the value of the product

Scrum

- The development team
  - Cross-functional, self-organizing team consisting of team members who have everything they need within the team to deliver working product without depending on others outside of the team

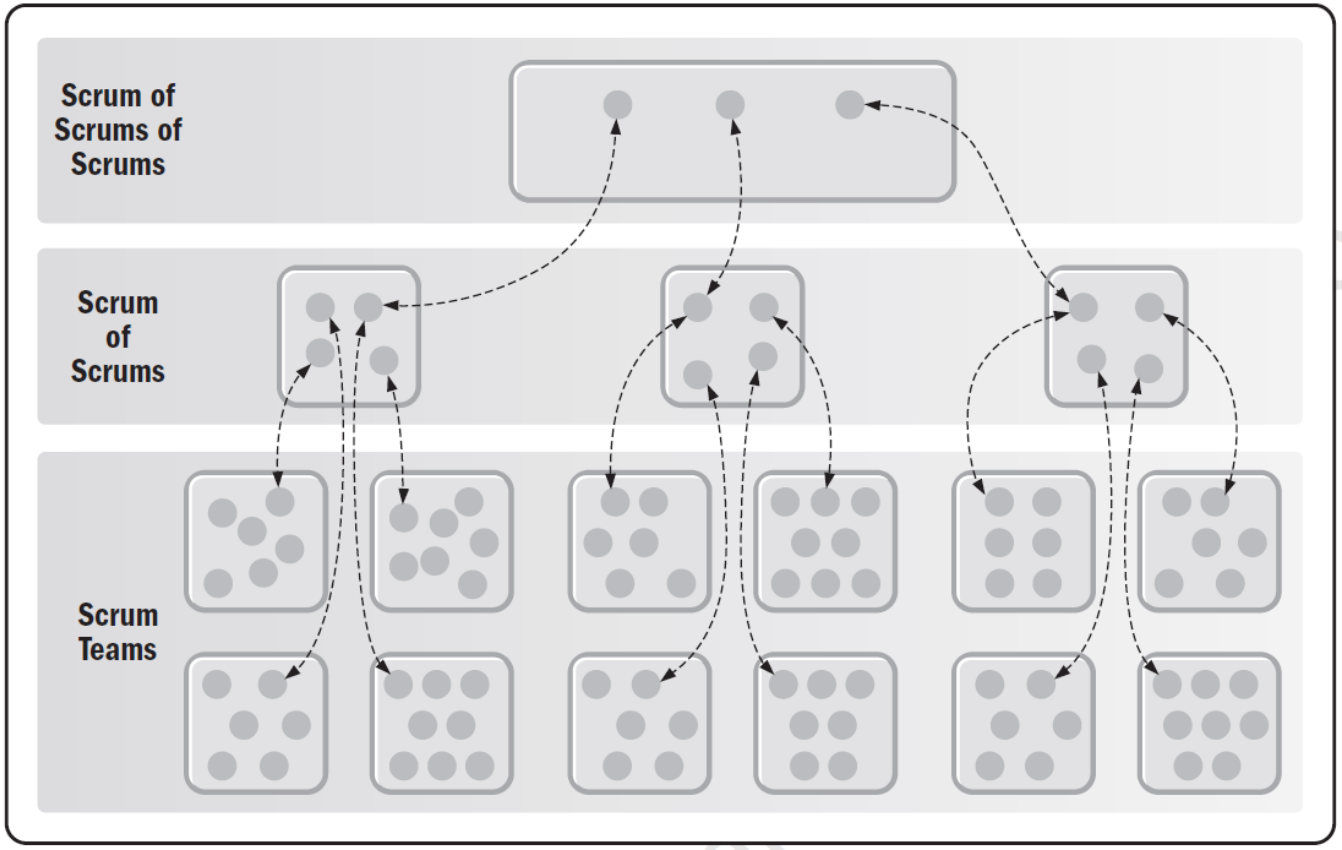
Scrum

- The scrum master
  - Responsible for ensuring the Scrum process is upheld and works to ensure the Scrum team adheres to the practices and rules as well as coaches the team on removing impediments

Scrum

Events	Artifacts
Sprint	Product backlog
Sprint planning	Sprint backlog
Daily scrum	Increments
Sprint review	
Sprint retrospective	

Scrum of Scrums (SoS)



Attributes of Successful Agile Teams

Attribute	Goal
Dedicated people	• Increased focus and productivity • Small team, fewer than ten people
Cross-functional team members	• Develop and deliver often • Deliver finished value as an independent team • Integrate all the work activities to deliver finished work • Provide feedback from inside the team and from others, such as the product owner
Colocation or ability to manage any location challenges	• Better communication • Improved team dynamics • Knowledge sharing • Reduced cost of learning • Able to commit to working with each other
Mixed team of generalists and specialists	• Specialists provide dedicated expertise and generalists provide flexibility of who does what • Team brings their specialist capabilities and often become generalizing specialists, with a focus specialty plus breadth of experience across multiple skills
Stable work environment	• Depend on each other to deliver • Agreed-upon approach to the work • Simplified team cost calculations (run rate) • Preservation and expansion of intellectual capital

Common Agile Practices

- **Retrospective**
  - Allows the team to learn about, improve, and adapt its process
  - Helps the team learn from its previous work on the product and its process
  - The principles behind: “At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly”

Common Agile Practices

- **Backlog Preparation**
  - The ordered list of all the work, presented in story form, for a team
  - No need to create all of the stories for the entire project before the work starts
  - Product owners might produce a product roadmap to show the anticipated sequence of deliverables over time

Common Agile Practices

- **Backlog Refinement**
  - The product owner often works with the team to prepare some stories for the upcoming iteration during one or more sessions in the middle of the iteration
  - To refine enough stories so the team understands what the stories are and how large the stories are in relation to each other
  - Not more than 1 hour per week

Common Agile Practices

- **Daily Standups**
  - To microcommit to each other, uncover problems, and ensure the work flows smoothly through the team
  - No longer than 15 minutes
  - The team “walks” the Kanban or task board in some way, and anyone from the team can facilitate the standup

Common Agile Practices

- **Daily Standups**
  - Answers
    - What did I complete since the last standup?
    - What am I planning to complete between now and the next standup?
    - What are my impediments (or risks or problems)?
  - Standups are for realizing there are problems—not for solving them
  - Teams run their own standups

Common Agile Practices

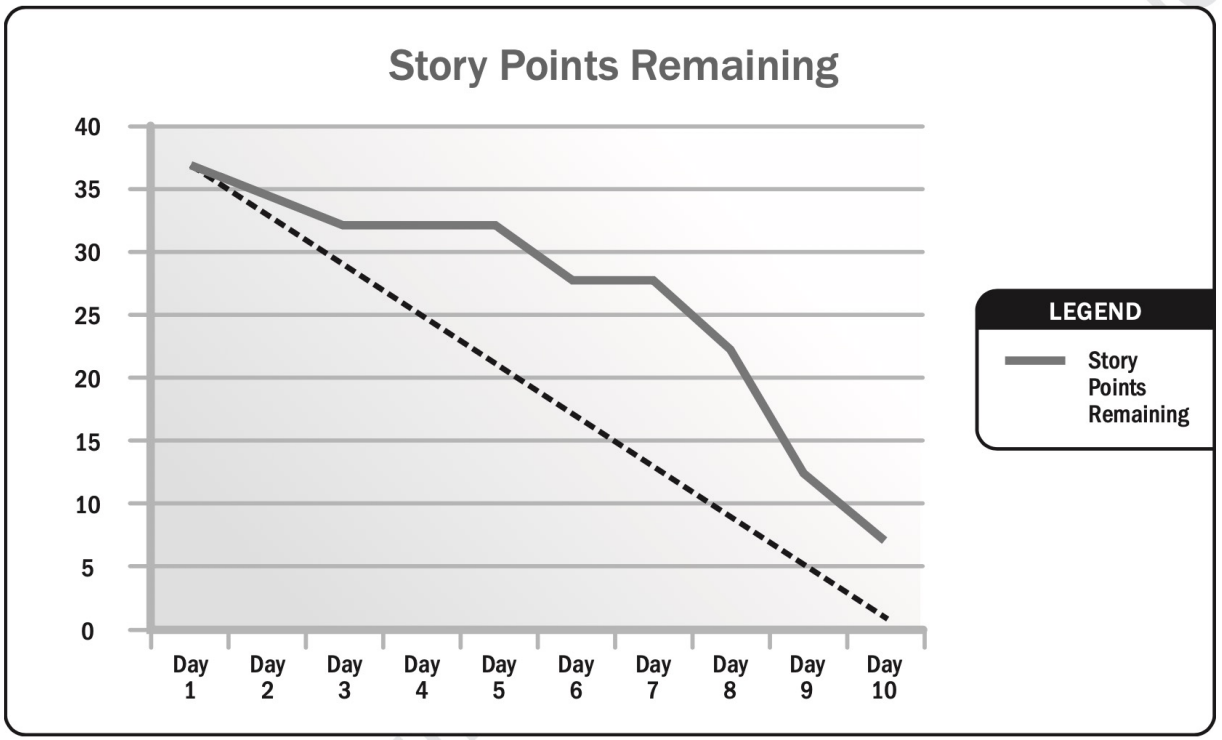
- **Demos/Reviews**
  - As the team completes the features, the team periodically demonstrates the working product
  - The product owner accepts or declines
  - Demonstrate whatever the team has as a working product
  - A fundamental part of what makes a project agile is the frequent delivery of a working product

Common Agile Practices

- **Planning Iterations**
  - Teams estimate what they can complete
  - When product owners make the stories smaller and teams see progress in the form of a finished product, teams learn what they are able to do for the future
  - Agile teams do not plan just once in one single chunk. Instead, agile teams plan a little, deliver, learn, and then replan a little more in an ongoing cycle

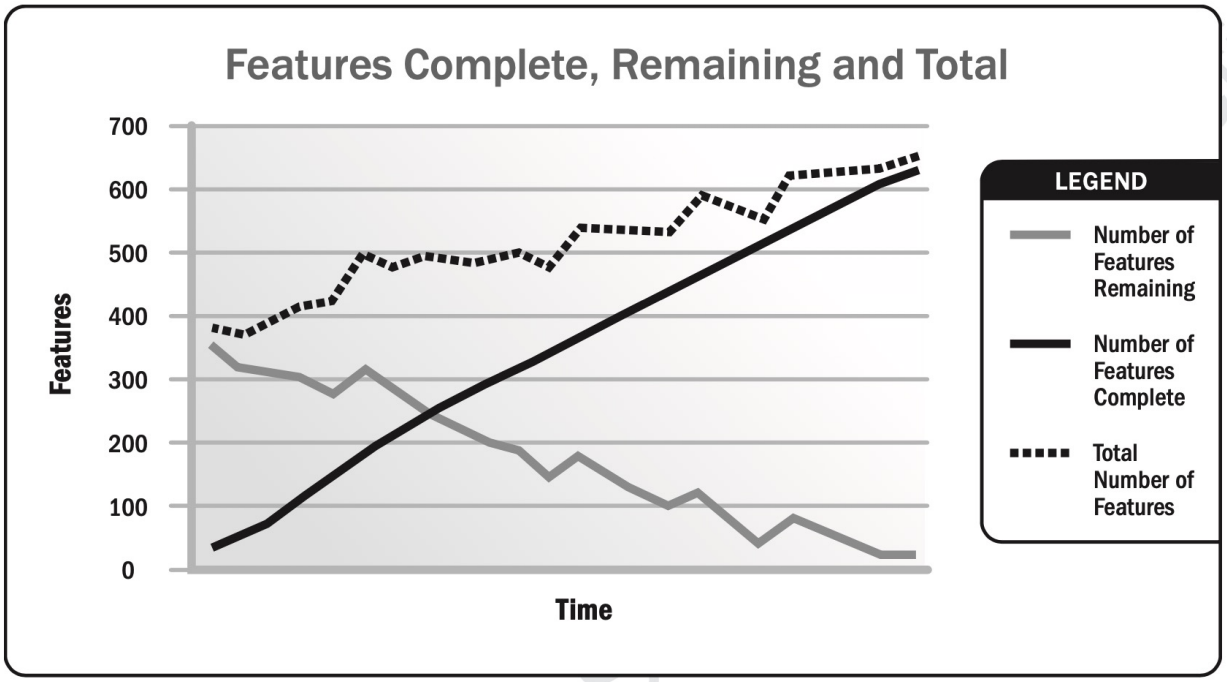
Common Agile Practices

- **Burndown Chart**



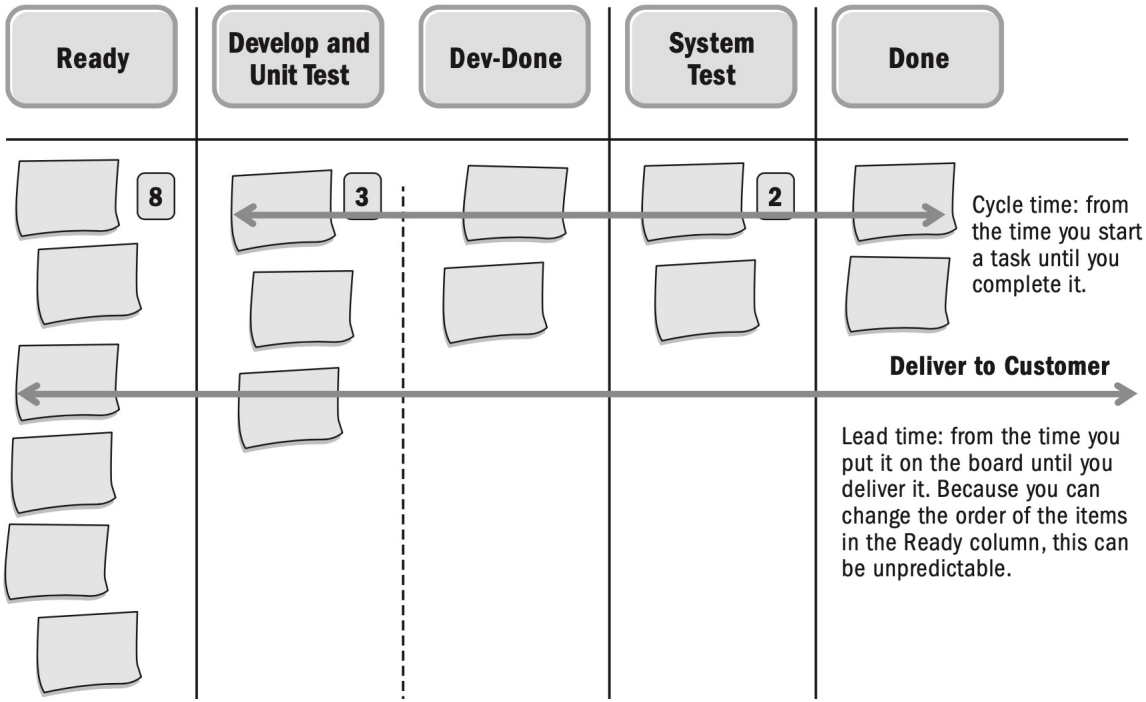
Common Agile Practices

- **Feature Chart**



Common Agile Practices

- **Kanban Board**



Common Agile Practices

- **eXtreme Programming**

XP Practice Area	Primary	Secondary
Organizational	• Sit together • Whole team • Informative workspace	• Real customer involvement • Team continuity • Sustainable pace
Technical	• Pair programming • Test-first programming • Incremental design	• Shared code/collective ownership • Documentation from code and tests • Refactoring
Planning	• User stories • Weekly cycle • Quarterly cycle • Slack	• Root cause analysis • Shrinking teams • Pay per use • Negotiated scope contract • Daily standups
Integration	• 10-minute build • Continuous integration • Test-first	• Single code base • Incremental deployment • Daily deployment

References

- Agile Practice Guide, 2017, Project Management Institute

Agile Project Management

CS413 - Software Engineering Project Management

Department of Computer Engineering, Bilkent University

Dr. Mustafa Değerli

Bilkent University