Agile Processes

CS 2720
Agile Overview

Agile software development methods were “introduced” in the Agile manifesto (2001) as a response to the high cost incurred by change in the plan-based approaches of the past.

It is worth reading the four values and twelve principles.
Agile Methods

There are several different software development processes (or “frameworks”) which follow the agile principles, including:

- Extreme programming (XP)
- Kanban
- Scrum
Extreme programming (XP) is one of the first, and one of the most “stereotypical”, agile processes. In *Extreme Programming Explained: Embrace Change* (1st edition) by Beck, twelve primary practices are given:

- Planning Game
- Small Releases
- Metaphor
- Simple Design
- Testing
- Refactoring
- Pair Programming
- Collective Ownership
- Continuous Integration
- 40-Hour Week
- On-Site Customer
- Coding Standards
Another agile process, one that focuses chiefly on management and not on development, is *scrum*. Scrum provides an easy-to-integrate starting point for agile methods. You should review Scrum in the textbook and on the Scrum Wikipedia page.

As far as use goes, Mike Cohn’s recommendation is to “...start with Scrum and then invent your own version of XP.” (Blog post from April 6, 2007).
Consider the scenario given below. Would you recommend a plan-driven approach like waterfall, or an agile approach like XP? Justify your response.

- CatCode is a custom software development firm with around 35 developers. They have been contracted by a local sports organization to build a system to help with recruiting and training. Specifically, they will build a program which analyzes videos of athletes performing various drills and assigns each athlete a set of scores representing how well they completed the drill. CatCode will also build the website which the sports organization will use to view the results of these drills, as well as track athlete progress and history. CatCode will assign all 35 developers to this project.
The Scrum team meets with the product owner to develop a set of epics and user stories for the product backlog. These epics and user stories can be placed on the storyboard.

The Scrum team and product owner hold a sprint planning event to select the user stories from the product backlog which will be implemented during this sprint. These user stories are moved to the sprint backlog.

Every day of the sprint, the Scrum team holds a brief stand-up (or daily Scrum) where team members share their progress, plans, and impediments.
4 Team members can monitor their sprint progress on the burn down chart, which is updated daily.

5 After the sprint is completed (2-4 weeks is typical), the Scrum team holds a sprint review with the product owner to demonstrate their progress.

6 After the sprint, the Scrum team also holds a sprint retrospective where they review what went well and what could be improved. They can also review things like their velocity.