Software Engineering (CSE 415)

An Agile View of Process

Introduction

- An agile philosophy for software engineering stresses four key issues –
  - The importance of self-organizing teams that have control over the work they perform
  - Communication and collaboration between team members and between practitioners & their customers
  - A recognition that change represents an opportunity
  - An emphasis on rapid delivery of software that satisfies the customer
- Agile process models have been designed to address each of these issues.

The Manifesto for Agile Software Development

- "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."

-- Kent Beck et al.

What is “Agility”?

- Effective response to change
- Effective communication among all stakeholders
- Drawing the customer onto the team; eliminate the “us and them” attitude
- Organizing a team so that it is in control of the work performed
- Rapid and incremental delivery of software

Principles to achieve agility – by the Agile Alliance (1)

1. Highest priority ==> satisfy the customer
2. Welcome changing requirements
3. Deliver working software frequently
4. Business people and developers must work together
5. Build projects around motivated individuals
6. Emphasize face-to-face conversation

Principles to achieve agility – by the Agile Alliance (2)

7. Working software is the primary measure of progress
8. Agile processes promote sustainable development
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
11. The best designs emerge from self-organizing teams
12. The team tunes and adjusts its behavior to become more effective

Agile Software Process

- Agile Software Process – Characterized by three Key Assumptions:

1) Difficulty in predicting changes of requirements and customer priorities.
2) For many types of s/w, design and construction are interleaved.
3) Analysis, design, construction, and testing are not as predictable as we might like.

Agile Software Process

- An agile process must be adaptable
- It must adapt incrementally
- Requires customer feedback
- An effective catalyst for customer feedback is an operational prototype or a portion of an operational system
  - An *incremental development strategy* should be instituted.
- Software increments must be delivered in short time periods
  - Enables the customer to evaluate the software increment regularly
  - Provide necessary feedback to the software team

Agile Process Models

- Extreme Programming (XP)
- Adaptive Software Development (ASD)
- Dynamic Systems Development Method (DSDM)
- Scrum
- Crystal
- Feature Driven Development (FDD)
- Agile Modeling (AM)

Extreme Programming (XP)

- The most widely used agile process, originally proposed by Kent Beck.
- XP uses an object-oriented approach as its preferred development paradigm.
- XP suggests a number of innovative & powerful techniques that allow an agile team to create frequent software releases delivering features & functionality described & prioritized by the customer.

Extreme Programming (XP)

- **XP** defines four framework activities:
  1) Planning
  2) Design
  3) Coding
  4) Testing

XP – Planning

- Begins with the creation of a set of *stories* (also called *user stories*)
- Each story is written by the customer and is placed on an index card
- The customer assigns a value (i.e. a priority) to the story
- Agile team assesses each story and assigns a cost
- Stories are grouped together for a deliverable increment
- A commitment is made on delivery date, and the XP team orders the stories to be developed
- After the first software increment, the XP team computes "project velocity" which is used to help define subsequent delivery dates for other increments.

XP – Design

- Follows the KIS (keep it simple) principle
- Encourages the use of CRC (class-responsibility-collaborator) cards as an effective mechanism for thinking about the software in an object-oriented context.
- For difficult design problems, XP suggests the creation of “*spike solutions*”—a design prototype.
- Encourages “*refactoring*”—an iterative refinement of the internal program design.
- Design occurs both before and after coding commences.

XP – Coding

- Recommends the construction of a series of unit tests for each of the stories before coding commences
- Encourages "*pair programming*"
  - Mechanism for real-time problem solving and real-time quality assurance
  - Keeps the developers focused on the problem at hand
- Needs "continuous integration" with other portions (stories) of the s/w, which provides a "smoke testing" environment that helps to uncover errors early.

XP – Testing

- Unit tests should be implemented using a framework to make testing automated. This encourages a regression testing strategy.
- Integration and validation testing can occur on a daily basis.
- *Acceptance tests*, also called *customer tests*, are specified by the customer and focus on overall system features & functionality that are visible & reviewable by the customer.
- Acceptance tests are derived from user stories.

Adaptive Software Development (ASD)

- ASD stresses human collaboration and team self-organization.
- ASD uses an *iterative* process that incorporates adaptive cycle planning , relatively rigorous requirements gathering methods, and an iterative development cycle that incorporates customer focus groups & FTRs as real-time feedback mechanism.
- ASD philosophy has merit regardless of the process model used. ASD's overall emphasis on the dynamics of self-organizing teams, interpersonal collaboration, and individual & team learning yield software project teams that have a much higher likelihood of success.

Adaptive Software Development (ASD)

- ASD is organized as three framework activities –

1) Speculation ==> Project initiation, Adaptive Cycle Planning(Customer's mission statement, Project constraints, Basic requirements, Time-boxed release plan)
2) Collaboration ==> Requirements gathering , JAD, Mini-specs.
3) Learning ==> Components designed, implemented, tested.

   ASD teams learn in three ways –*Focus groups* for feedback, *FTRs*, *Postmortems*

Dynamic Systems Development Method (DSDM)

- An agile software development approach that provides a framework for building & maintaining systems which meet tight time constraints through the use of incremental prototyping in a controlled project environment.
- Similar in some respects the RAD process.
- Suggests a philosophy that is borrowed from a modified version of the *Pareto principle* (80 % of an application can be delivered in 20% of the time).
- Like XP & ASD, DSDM suggests an iterative software process.
- Advocates the use of time-box scheduling and suggests that only enough work is required for each software increment to facilitate movement to the next increment.

Dynamic Systems Development Method (DSDM)

- DSDM defines three different iterative cycles –
    1) Functional model iteration
    2) Design and build iteration
    3) Implementation
- These three cycles are preceded by two additional life cycle activities –
    a) Feasibility study
    b) Business study

Scrum

- **Scrum** emphasizes the use of a set of software process patterns that have proven effective for projects with tight timelines, changing requirements, and business criticality.
- **Framework activities** –Requirements, Analysis, Design, Evolution, Delivery. Within each framework activity, work tasks occur within a process pattern called a *sprint*.
- Each process pattern defines a set of development tasks and allows the Scrum team to construct a process that is adapted to the needs of the project.
  - Backlog
  - Sprint
  - Scrum meeting

Scrum

- **Scrum** is an iterative and incremental agile software development method.
- Scrum incorporates a set of process patterns that emphasize:
  - Project priorities
  - Compartmentalized work units
  - Communication
  - Frequent customer feedback

Agility vs. Software Engineering

- *So…..What you gonna choose?*
- *Agility, or Software Engineering?*

Agility vs. Software Engineering

- You don’t have to choose between agility and software engineering!
- Instead, define a software engineering approach that is agile.