AGILE DEVELOPMENT

Rabindra Bista, PhD
Dept. of CSE
Kathmandu University

December 06, 2021

Contents

- Software Engineering
- What is an Agile Development?
- What is Agility?
- Agility and Cost of Change
- What is an Agile Process?
- Agility Principles
- Human Factors

Contents

- Extreme Programming (XP)
- Industrial Extreme Programming (IXP)
- Scrum

Software Engineering

- Software engineering defined as “ the establishment and use of sound engineering principles in order to obtain economically developed software that is reliable and worked efficiently on real machines”

What is an Agile Development?

- Philosophy + A set of Development Guidelines
- Philosophy encourages
  - Customer satisfaction and early delivery of s/w
  - Small highly motivated project teams
  - Informal methods
  - Minimal software engineering work products
  - Overall development simplicity
- Development Guidelines stresses
  - Delivery over analysis and design
  - Active and continuous communication between developers and customers

What is Agility?

- An effective response to change
  - s/w, team members, new technology etc.
- Encourages team structures and attitudes that make communication more simple
- Rapid delivery of operational s/w
  - Emphasizing incremental delivery strategy
- Adopts the customer as a part of the development
- Recognizes a project plan must be flexible
- Focuses on the most essential work products

Agility and the Cost of Change

- In the conventional way of s/w development
  - First, the cost of change increases nonlinearly as a project progress
    - Early requirement gathering and the cost is minimal
  - Then the cost escalate quickly when major changes require in the middle or final stage of project development phases

Agility and the Cost of Change

- In Agile development
  - A significant reduction in he cost of change can be achieved
    - Incremental delivery is coupled with other agile practices
      - Continuous unit testing, pair programming and so on

Agility and the Cost of Change

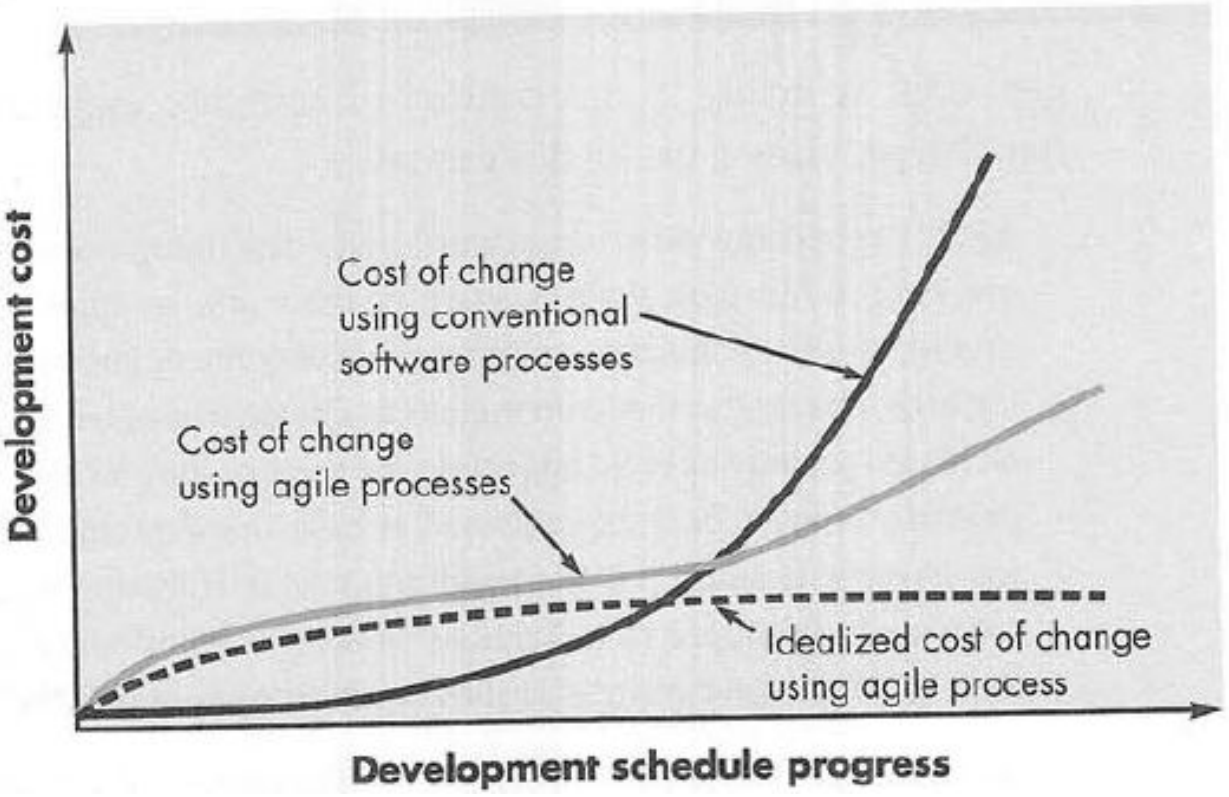

Fig. 1: Change cost as a function of time in development

What is an Agile Process?

- Unpredictability
  - Analysis
  - Design
  - Construction
  - Testing
- Requirements and their priorities
- How much design is necessary before construction?

What is an Agile Process?

- How do we create a process that can manage unpredictability?
  - Process adaptability
    - Incremental adaptation
      - Based on customers' feedbacks on operational prototype
    - Adaptation must keep pace with change
  - S/W increments must be delivered in short time periods

What is an Agile Process?

- An agile process can be defined as an iterative approach enabling the customer to evaluate the software increment regularly, provide necessary feedback to the software team, and influence the process adaptations that are made to accommodate the feedback

Agility Principles

The Agile Alliance (see [Agi03], [Fow01]) defines 12 agility principles for those who want to achieve agility:

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.
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.
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.
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.

Agility Principles

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 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.

The Politics of Agile Development

- Debate on benefits and applicability of Agile Software Development over Traditional Software Development
- What is the best way to achieve agility?
- Consider the best of both approaches

Human Factors

- Agile development focuses on the talents and skills of individuals, molding the process to specific people and teams
- Key traits of people on a agile team and the team itself:
  - Competence:
    - innate talent, specific s/w related skills and the overall knowledge of the process
  - Common Focus:
    - all should be focused on one goal-
      - to deliver a working s/w increment to the customers within the time promised
  - Collaboration:
    - team members must collaborate with one another and all other stakeholders

Human Factors

- Decision making ability:
  - team is given autonomy- decision making authority for both technical and project issues
  - allowed freedom to control its own destiny
- Fuzzy problem solving ability:
  - lesson learned from any problem solving activity may be of benefit to the team later in the project

Human Factors

- Mutual trust and respect:
  - jelled team (strongly knit team members)
  - Whole is greater than the sum of he parts
- Self-organization:
  - The agile team organizes
    - Itself for the work to be done
    - The process to best accommodate its local environment
    - The work schedule to best achieve delivery of the software increment

Extreme Programming (XP)

- The most widely used approach to agile S/W development
- **XP Values:** a set of five values
  - Communication, Simplicity, Feedback, Courage and Respect
    - Are used as a driver for specific XP activities, actions and tasks

Extreme Programming (XP) (Contd…)

- Communication:
  - XP emphasizes to establish effective metaphors (story or use case) for communication so that s/w engineers and other stakeholders know how the system works
- Simplicity:
  - Designers are encouraged to design a part of s/w (fraction) only for immediate needs not for he whole system at a time

Extreme Programming (XP) (Contd…)

- Feedback: three sources feedback
  - Implemented s/w itself, the customer and other s/w team members
- Effective testing strategy (unit testing), incremental development -> iterative planning
- The team provides the customer with rapid feedback in terms of cost and schedule impact

Extreme Programming (XP) (Contd…)

- Courage/ Discipline:
  - Design for today not for tomorrow because future requirement may change dramatically
- Respect:
  - Among team members, between other stakeholders and the team members, and software itself
    - Grows respect for the XP process

Extreme Programming (XP) (Contd…)

- **The XP Process:**
- Used an object-oriented approach
- Object Oriented = Objects + Classification (sub-class, super class) + Inheritance + Communication
- Four key framework activities:
  - Planning, Design, Coding and Testing

Extreme Programming (XP) (Contd…)

- Planning or Planning Game:
  - Listening to understand the business context for the s/w
  - Create stories (users' stories)
    - Describes required out put, major features and functionalities to be built
    - Customer assigns a value (for priority) to the story
  - The XP team members assign a cost (development week) to the story

Extreme Programming (XP) (Contd…)

- Story > 3 weeks -> customer splits the story into smaller stories -> reassign value and cost
- Commitment for a release : three options
  - All stories will be implemented immediately
  - Stories with the highest value, first
  - The riskiest stories, first
- Release first s/w increment
- Team members calculates project velocity
- The no. of customer stories implemented during the first release
  - Useful for further planning

Extreme Programming (XP) (Contd…)

- Design:
  - Data , architecture, components and interfaces
  - Follows KIS (keep it simple) principle
  - Provides implementation guidelines
  - Use of CRC (Class-Responsibility-Collaboration) cards
    - Are design work product
    - Identify and organize the classes (object oriented) that are relevant to system or product requirement

Extreme Programming (XP) (Contd…)

- Spike solution: the design prototype is implemented and evaluate for a story which occurs as a difficult design problem
- Refactoring: Cleaning up Code
  - Design occurs continuously as the system is constructed

- Coding:
  - First develop a series of unit tests for stories
    - Better able to focus on what must be implemented to pass the test

Extreme Programming (XP) (Contd…)

- Code for the each of the stories
- Unit tested immediately
- Key concept is pair programming
  - Two people work together at one computer to create code for a story
    - Real time problem solving
    - Real time quality assurance
- Continuous integration and smoke testing (for time critical projects)

Extreme Programming (XP) (Contd…)

- Testing:
  - Creation of unit tests before coding
  - Implement the unit tests
  - Encourages regression testing whenever code is modified:
    - re-execution of some sub-set of tests that have already been conducted to avoid unintended side effects due to change

Extreme Programming (XP) (Contd…)

- Universal testing suite:
  - individual unit tests are organized
- Integration and validation testing may occur on a day basis
- Acceptance tests (customer tests)
  - Derived from user stories

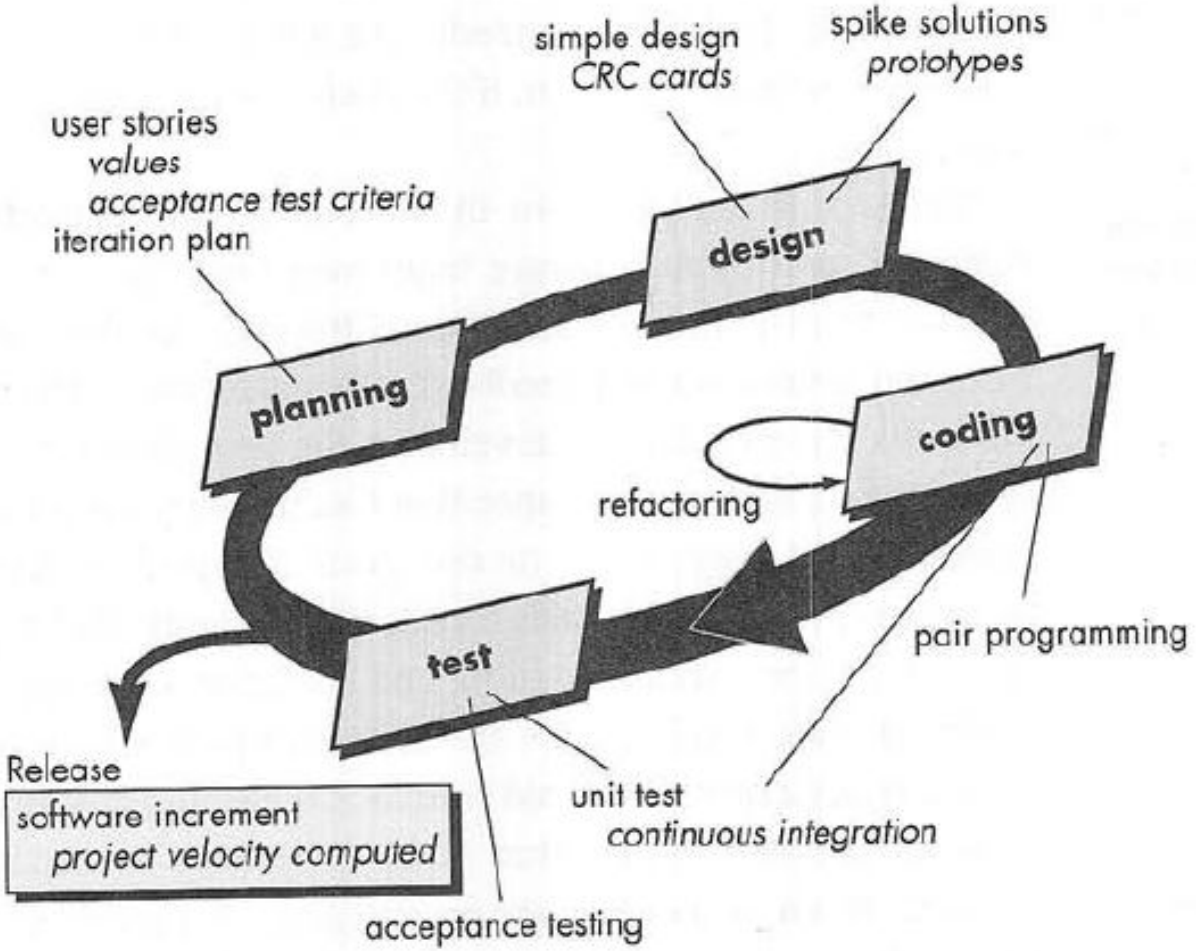

Fig 2:The Extreme Programming process

Industrial Extreme Programming (IXP)

- Organic evolution of XP for large organizations
- Imbued with XP's minimalist, customer-centric, test-driven spirit
- Greater inclusion of management, its expanded role for customers and its upgraded technical practices

Industrial Extreme Programming (IXP)

- **IXP incorporates six new practices**
- Readiness assessment:
  - Existence of an appropriate development environment
  - Team with proper set of stakeholders
  - Distinct quality program and continuous improvement
  - Support new values of an agile team by organizational culture
  - populate broader project community

Industrial Extreme Programming (IXP)

- Project Community:
  - Morphing of a team into community
    - Technologist, customers, other stakeholders
    - The role of the community members must be explicitly defined
- Project Chartering:
  - IXP team examines
    - Business justification for the project
    - Over goals and objectives
    - Context of the project

Industrial Extreme Programming (IXP)

- Test-driven Management:
  - Establishes a series of measurable destinations
  - Defines mechanism for determining whether or not these destinations have been reached
- Retrospectives:
  - Conducts a specialized technical review after a s/w increment is delivered
  - *Examines issues, events and lesson-learned*
- Continuous Learning:
  - Encourage to learn new methods and techniques

Industrial Extreme Programming (IXP)

- **Other practices in IXP:**
  - Story-driven development (SDD): Stories for acceptance test be written
  - Domain-driven Design (DDD): Creation of domain model to represent how domain experts think about their subject
  - Pairing: include mangers and other stakeholders
  - Iterative usability: usability design evolves with software increments

The XP Debate

- Codependent nature of XP practices are both its strength and weakness
- Issues:
  - Requirements volatility
  - Conflicting customer needs
  - Requirement are expressed informally
  - Lack of formal design

Other Agile Process Models

- Adaptive Software Development (ASD)
- Scrum
- Dynamic Systems Development Method (DSDM)
- Crystal
- Feature Driven Development (FDD)
- Lean Software Development (LSD)
- Agile Modeling (AM)
- Agile Unified Process (AUP)

Scrum

- For project with tight timelines, changing requirements, and business criticality
- **Five Framework activities:**
  - Requirement, Analysis, Design, Evolution, and Delivery
  - By a team called Scrum Team
  - Within each framework activity, work tasks occur within a process pattern called a Sprint
    - A *basic unit* of development in scrum
  - Higher the product complexity and size higher the number of sprints for each framework activity

Scrum

- **Development actions:**
  - Backlog:
    - A prioritized list of high level project requirements or features which is changeable
    - Product manager assesses the backlog and updates priorities
  - Sprints:
    - Work units to achieve a requirement defined in the backlog
    - Must fit into a predefined time-box (30 days)
    - Allows work in short-term with stable environment
      - There is no change in the sprint level
      - At the end, new functionality is demonstrated

Scrum

- Scrum Meetings:
  - Team leader or the Scrum Master leads the meetings
  - Daily 15 minutes meeting
  - Uncover potential problems as soon as possible
  - Includes three key questions
    - What did you do since the last team meeting?
    - What obstacles are you encountering?
    - What do you plan to accomplish by the next team meeting?
- Demos:
  - Deliver the s/w increment (functionality) to the customer for evaluation

Scrum

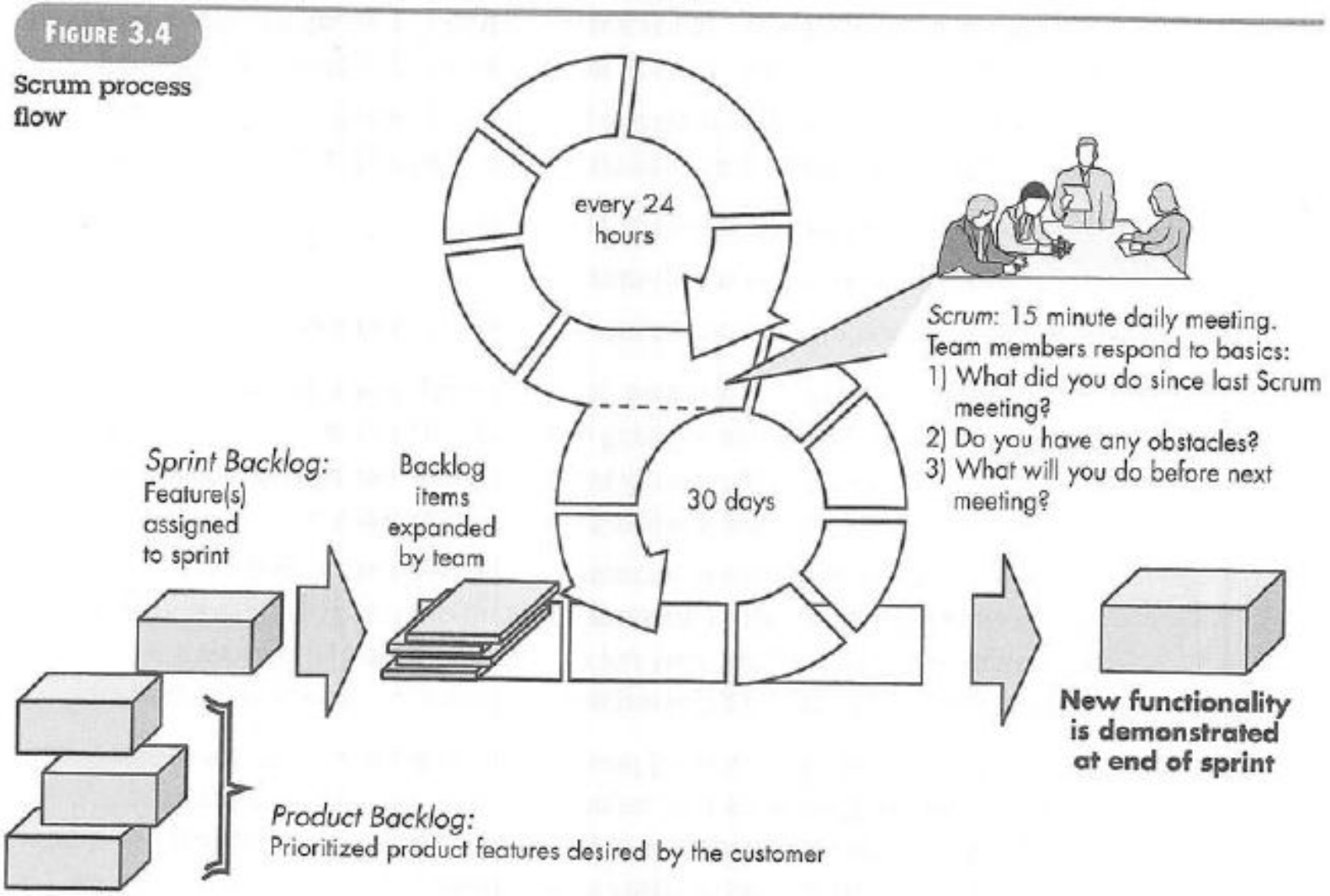

FIGURE 3.4
Scrum process flow