Software Process Model

UNIVERSITY of INFORMATION TECHNOLOGY

Learning Objectives

Lesson Objectives

- To understand the principles of agile processes
- To know the concept of development agility and the Agile Manifesto
- To review each of the major agile development methods underscoring their strengths and weaknesses

4.1 Agile Method

❖ agile software process addresses a number of key assumption

1) It is difficult to predict in advance which software requirements will persist and which will change. It is equally difficult to predict how customer priorities will change as the project proceeds.
2) It is difficult to predict how much design is necessary before construction is used to prove the design
3) Analysis, design, construction, and testing are not as predictable

4.1

Agile Method

- Emphasis on flexibility in producing software quickly and capably

Agile manifesto

- Value individuals and interactions over process and tools
- Prefer to invest time in producing working software rather than in producing comprehensive documentation
- Focus on customer collaboration rather than contract negotiation
- Concentrate on responding to change rather than on creating a plan and then following it

Agile Method

UNIVERSITY of INFORMATION TECHNOLOGY

Examples of Agile Process

1) Extreme programming (XP)
2) Crystal: a collection of approaches based on the notion that every project needs a unique set of policies and conventions
3) Scrum: 30-day iterations; multiple self-organizing teams; daily “scrum” coordination
4) Adaptive software development (ASD)

4.1

Agile Method

Agile Methods: Extreme Programming

- Emphasis on four characteristics of agility
  - *Communication*: continual interchange between customers and developers
  - *Simplicity*: select the simplest design or implementation
  - *Courage*: commitment to delivering functionality early and often
  - *Feedback*: loops built into the various activities during the development process

4.2 Agile Method: Twelve facts of Extreme Programming

UNIVERSITY of INFORMATION TECHNOLOGY

1. **The planning game *(customer defines value)***
2. **Small release**
3. **Metaphor** *(common vision, common names)*
4. **Simple design**
5. **Writing tests first**
6. **Refactoring**

4.2 Agile Method: Twelve facts of Extreme Programming

7 **Pair programming**

8 **Collective ownership**

9 **Continuous integration** *(small increments)*

10 **Sustainable pace** *(40 hours/week)*

11 **On-site customer**

12 **Coding standard**

4.2 Agile Method: Twelve facts of Extreme Programming

Planning game:

Begins with listening : The requirements gathering activity that understand the business context for the software and required output and major features and functionality.

Small releases:

The functionality can be delivered as soon as possible. Functions are decomposed into small parts. It requires a phased–development approach.

4.2 Agile Method: Twelve facts of Extreme Programming

Metaphor :

The development team agrees on a common vision of how the system will operate. To support its vision, the team chooses common names and agrees on a common way of addressing key issues.

Simple design :

If a particular portion of a system is very complex, the team may build a spike- a quick and narrow implementation- to help it decide how to proceed.

4.2

Agile Method: Twelve facts of Extreme Programming

UNIVERSITY of INFORMATION TECHNOLOGY

Writing test first :

Two kinds of tests:

(1) Functional tests

- -specified by the customer and executed by both developers and users
- -Considered to be part of the system specification

(2) Unit Tests

- -are written and run by developers
- -Verify each modular portion of the implementation works as designed.

4.2 Agile Method: Twelve facts of Extreme Programming

UNIVERSITY of INFORMATION TECHNOLOGY

Refactoring :

Revisiting the requirement and design, reformulating them to match new and existing needs.

Pair Programming :

Attempts to address the artistic side of software development, Using one keyboard, two paired programmers develop a system from the specifications and design

4.2 Agile Method: Twelve facts of Extreme Programming

Collective ownership :

Any developer can make a change to any part of system as it is being developed.

Continuous Integration :

Emphasis is on small increments or improvements

4.2 Agile Method: Twelve facts of Extreme Programming

UNIVERSITY of INFORMATION TECHNOLOGY

Sustainable pace :

Suggest a goal of 40hrs for each work week. Developers can devote much time to meeting deadlines when the deadlines are unreasonable or insufficient resources

On-site customer :

A customer should be present on-site, working with developers to determine requirements and providing feedbacks

4.2

Agile Method: Twelve facts of Extreme Programming

UNIVERSITY of INFORMATION TECHNOLOGY

Coding standards :

advocates clear definition of coding standards, to encourage teams to be able to understand and change each other’s work.

4.2 Agile Method: Twelve facts of Extreme Programming

When Extreme is Too Extreme?

- Extreme programming's practices are interdependent
  - ✓ A vulnerability if one of them is modified
  - ✓ Uncomfortable with pair programming -> require more coordination and documentation

4.2 Agile Method: Twelve facts of Extreme Programming

UNIVERSITY of INFORMATION TECHNOLOGY

When Extreme is Too Extreme?

- Requirements expressed as a set of test cases must be passed by the software
  - ✓ System passes the tests but is not what the customer is paying for
- Refactoring issue
  - ✓ Difficult to rework a system without degrading its architecture

Quiz

1

Agile Software Development is based on

a) Incremental Development

b) Iterative Development

c) Linear Development

[✓ d) Both Incremental and Iterative Development]

UNIVERSITY of INFORMATION TECHNOLOGY

Quiz

2

Select the option that suits the Manifesto for Agile Software Development:

a) Working software

b) Individuals and interactions

c) Customer collaboration

[✓ d) All of the above]

UNIVERSITY of INFORMATION TECHNOLOGY

Quiz

3

Which of the following framework activities are found in the Extreme Programming(XP) ?

a) Planning, Analysis, Design, Coding

b) Analysis, Design, Coding, Testing

[✓ c) Planning, Design, Coding, Testing]

d) None of the above