Agile Methodologies

Software Process

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Tao Xie, Professor
Computer Science @ Illinois
Learning Objectives

By the end of this video, you will be able to
• Explain what the basic principles of agile software development are.
• Explain what the core practices of XP are and why they are effective.
• Explain what pair programming is about and its benefits.
• Explain the steps in the Scrum process.
Manifesto for Agile Software Development

- Individuals and interactions
- Working software
- Customer collaboration
- Responding to change
- Processes and tools
- Comprehensive documentation
- Contract negotiation
- Following a plan

(Agile Manifesto authors, 2001)
eXtreme Programming (XP)

• Radically different from waterfall

• Big ideas
  • Don’t write much documentation
    • Working code is the main written product
  • Implement features one by one
  • Release code frequently
  • Work closely with the customer
  • Communicate a lot with team members
XP: Some Key Practices

• **Planning game** for requirements

• **Test-driven** development for design and testing

• **Refactoring** for design

• **Pair** programming for development

• **Continuous** integration for integration
XP: An Iterative Process

• Organize each iteration as a **two-week cycle**

• **Plan** each iteration in an iteration meeting held at the start

• Iteration is going to implement a set of **user stories**

• Divide work into **tasks** small enough to finish **in a day**

• Each day, programmers work in **pairs** to finish tasks
Working Software

• All software has automated (unit) tests

• All tests pass, all the time
  • Never check in broken code

• How to work on a task
  • Get latest version of the code – all tests pass
  • Write test first – it fails
  • Write code to make test pass – now all tests pass
  • Refactor (clean up)
  • Check in your code
Pair Programming

Two programmers work side-by-side at one computer, continuously collaborating on the same design, algorithm, code, and test

➔ Two people producing a higher quality of code than that produced by summing their solitary efforts

**Driver:** types or writes

**Navigator:** observer (looking for tactical & strategic defects)

- Periodically switch roles of driver and navigator
  - Every 30 minutes or less
- Pair coding, design, debugging, testing, etc.

(Williams, 2013)
Benefits of Pair Programming

(Williams et al., 2000)

• Anecdotal evidence:
  • “We can produce near defect-free code in less than half the time.”

• Empirical evidence
  • Pairs produced **higher quality code**
    • 15% fewer defects
  • Pairs completed their tasks **in about half the time**
    • 58% of elapsed time
  • Pair programmers are **happier programmers**
    • Pairs enjoy their work more (92%)
    • Pairs feel more confident in their work products (96%)
More Benefits of Pair Programming

• Pair Rotation
• Pair Learning
• Pair Pressure
• Pair Negotiation
• Pair Courage
• Pair Reviews
• Pair Debugging
• ...

[Image of two people working on computers]
Scrum Process

Sprint Backlog

- Backlog tasks expanded by team

Daily Scrum Meeting

24 hours

Product Backlog

- As prioritized by Product Owner

30 days

Potentially Shippable Product Increment

(Marekventur, 2011)
References


The End