Professional Issues II: Unit 8: Software Engineering *Extreme Programming (XP)*

Markus Roggenbach

April 2016

Finding a place for Extreme Programming (XP)

Background

Literature:

Kent Beck: Extreme Programming Explained.
Addison Wesley, 2nd Edition, 2005. (KB)

When & where:

invented by K Beck
at Chrysler in 1996

XP: Beauty and Beast

KB "codified XP to make life better for programmers".

XP is as much philosophy, consultancy as engineering.

Non-substantiated claims (in K Beck: XP explained)

- no reference projects
- no evidence for claims of
  - lower cost,
  - fewer defects, and
  - higher productivity.

Off place remarks / Platitudes

- Personal hygiene and health are important issues when pairing.
- Cover your mouth when you cough.
- Don't come to work when you are sick.
- Avoid strong colognes that might affect your partner.

(KB, p 43)

Consultancy phrases

- If you're have trouble succeeding, fail.
- Don't know which of three ways to implement a story? Try it all three ways. Even if they all fail, you'll certainly learn something valuable.

(KB, p 32)

SDM II: XP

The Whole XP Team

- Testers
- Interaction designers
- Architects
- Project managers
- Product managers
- Executives
- Technical writers
- Users
- Programmers

Replace “requirement” by “story”

Story: unit of customer visible functionality.

Examples:

1. Handle five times the traffic with the same response time.
2. Provide a two-click way for users to dial frequently used numbers.

Time management

- Weekly deployment:
  (a) Review progress.
  (b) Customer picks a week's worth of stories.
  (c) Break stories into tasks.
  (d) Team members sign up for tasks.
- Once a quarter reflect on
  - the team,
  - the project,
  - its progress, and
  - its alignment with larger goals.

Coding principles

- First write automated tests; then write the code.
- Ten-minute build: Automatically build the whole system and run all of the test cases in ten minutes.

(Lightweight) comparison with Waterfall model

- Fewer documents:
  - no specification – replaced by tests
  - no verification plan – replaced by 10 minute build
  - no module documentation – replaced by code commenting
- Higher flexibility:
  - system specification can change
  - user/market needs can be taken on board all the time

- Less thought through architecture:
  - functionality can change during development
  - architecture revision might turn out cumbersome

Pair programming

Pairs:

- Keep each other on task.
- Brainstorm refinements to the system.
- Clarify ideas.
- Take initiative when their partner is stuck, thus lowering frustration.
- Hold each other accountable to the team's practise.

XP in Lab 7 & 8

The task

Following the

XP paradigm,

build a computer game and test it with a software robot

Limitation of XP to Pair Programming

- There will be some infrastructure to start with.
- The lab-sheet will prescribe the stories.
- In total: 5 or 6 stories.
- You will form the Pair Programming teams.

The procedure for each story

1. Write test cases in JUnit
2. Code up the story
3. Test and debug the story
4. Test and debug the new product

Next lecture's topics

1. Understanding the Game
2. Understanding the programming principles of the Game
3. Understanding the setup of the software robot