Software Development Process

Lecture 7

XP

Extreme Programming

- **Extreme Programming** (**XP**) is a software development process focusing on excellent application of programming techniques, clear communication and team work [1]
  - Set of rules to follow that guarantee success
- XP
  - Takes extreme approach to iterative development
  - Emphasizes code improvements via refactoring
  - Lightweight: Addresses only constraints in software development. It does not address management, financial or project portfolio
  - Consists of 12 simple practices

XP Practices

- Planning Game
- Small Releases
- System Metaphor
- Simple Design
- Test-First Development
- Refactoring
- Pair Programming
- Collective Ownership
- Continuous Integration
- Sustainable Pace (40-Hour Work Week)
- On-Site Customer
- Coding Standards

1. Planning Game

- The game is a meeting held once per iteration
- Planning for next release
- The customers describe high-value requirements using "**user stories**"
- The dev team studies the user stories and produces a **task list**
  - The tasks are assigned to appropriate person
  - The time required for each task will be estimated

2. Small Releases

- Software is released frequently
- Each version has to be functional (albeit small increment in functionality)
- It helps the customer to gain confident in the project
- It also enables the customer to provide early feedback

3. System Metaphor

- A system metaphor is a set of terminologies that is used within a project, e.g.
  - Class names
  - Method names
- All stakeholders, customers, programmers and managers use the same set of words
  - Reducing confusion
  - Reducing technical jargons
  - Easier to explain the system

4. Simple Design

- It is encouraged to do things simple
- When you code, ask yourself:
  “Is this the simplest way to do? Is there any other simpler way?”
- If the code is complex, refactor
- Do not do any unnecessary thing

5. Test-First Development

- In XP, the automated unit tests are written ***before*** the coding
- The coder is said to ***finish*** the coding when all test conditions are passed
  - The coder would have think how to code such that the program does not fail the test
  - The test also gives the coder a clear goal
- When a new functionality is introduced, all the **new and old** tests are conducted

6. Refactoring

- **Refactoring**: A process of altering the code while preserving the same functionality
  - Changing the way (how) the system works, but does not change what it does
- Encourage the developers to keep improving the product

7. Pair Programming

- Two developers working in the same machine
  - One driver: Write codes
  - One observer: Review the codes
  - Roles are switched often
- Benefits:
  - Better design and shorter/cleaner programs
  - Alternative solutions/designs are considered more often
  - Instant code review
  - Knowledge sharing

8. Collective Ownership

- All programmers own the code
  - Everyone is responsible
  - Everyone can change the code
- Benefits
  - If an error occurs, anyone can fix it
  - If anyone leaves the project, other programmers still maintain knowledge
  - Everyone would have to take responsibility of the entire project, not just some parts of it

9. Continuous Integration

- Newly implemented code should be merged into the project immediately
  - If a developer has produced a new version locally, he has to pushed it to the repository as soon as possible (usually within few hours)
  - The developers should work on the latest version of the software
- This approach is intended to reduce delays caused by integration problems
- It also promotes small releases

10. Sustainable Pace

- The developer should not work more than 40 hours a week (8 hours a day)
- Too much working hours does not mean effective working hours
- As XP cycles are small and the software is frequently released, XP does not have huge deadlines and thus, requires no extra hours

11. On-Site Customer

- The customer should be on-site
- In XP, the customer is also the user, therefore he should be there to provide feedback or answers at all time
- He also ***steers*** the project
  - Guides the development to appropriate project outcomes
  - Ensures that the functionalities are prioritizes correctly

12. Coding Standards

- Coding standards are sets of coding rules upon which the developers agree
  - All codes should look the same
- The standard specifies coding styles, format and other coding conventions
- The code would be more understandable
  - Less inline comments
  - Clean and clear code
  - No need to reformat the code

XP Conclusion

- XP takes best practices iterated and incremental approach to the extreme
  - These best practices are reflected in XP practices
- It emphasizes
  - Simplicity
  - Team works and customer involvement
  - Small releases with systematic testing
  - Handle changes through refactoring
- It's problem:
  - Customer involvement might be difficult
  - May lead to inappropriate architectural design

Further Reading on XP

1. Beck, K., Andres, C., *Extreme Programming Explained: Embrace Change (2nd Ed.)*, Addison-Wesley Professional, 2004.
2. Sommerville, I., *Software Engineering (9th Ed.)*, Addison-Wesley, 2010.
3. Xprogramming.com
4. Extreme Programming: A gentle introduction, http://www.extremeprogramming.org/