WESTMINSTER

INTERNATIONAL UNIVERSITY IN TASHKENT

An Accredited Institution of the University of Westminster (UK)

System Analysis and Design

Lecture 8

Agenda

1. Previous lecture recap
2. Extreme Programming (XP) introduction
3. XP values
4. XP principles
5. XP rules
6. Questions
7. Recommended reading

Recap

1. Main ideas from **Scrum** methodology.
2. What is the difference between **Scrum and Kanban**?

eXtreme Programming

Iterative & Incremental Models

Major Software Development Models

There a many Software Development Models with their strengths and weaknesses. But, they can be categorized into 3 major categories:

1. **Waterfall** – Structured, top/down approach.
   Examples: Waterfall, Structured System Analysis and Development Methodology (SSADM).

2. **Iterative & Incremental** – Iteratively go through important software development "steps" (analysis, design, build, etc.) and incrementally build the system.
   Examples: Spiral, RAD, Agile, etc.

3. **Prototyping** – Build prototypes, get feedback, refine. Once get the final version – build the software and deploy.
   Examples: RAD, Prototyping.

Taking things to their extremes…

That’s basically is the idea!

eXtreme Programming

- First introduced in 1996 by Kent Beck at Chrysler's accounting system.

- Formulated and published as a software development methodology in 1999. Later, in 2004, Kent Beck reviewed the methodology and published the second edition of "eXtreme Programming explained" book. (We have several copies in our LRC.)

What is meant by “taking things to their extremes”?

Taking “**good practices**” to extreme.

For example:

- Code review improves code quality → make code review continues → Pair programming!
- Tests reduce errors, improve code → test everything → Cover all your code with unit tests, integration tests, etc.
- Etc.

XP: Values

1. Communication – promote among the team members, also involve end users actively.
2. Simplicity – design the simplest solution. Code for today, not tomorrow. "You ain’t gonna need it" (YAGNI) approach.
3. Feedback – from unit tests, acceptance tests, customer.
4. Courage – to change your code or throw it away.
5. Respect – your peers. Don’t commit a change that breaks tests or compilation. Make sure everyone feels appreciated and heard.

XP: Practices

Fine-scale feedback

- Pair programming
- Planning game
- Test-driven development
- Whole team

Continuous process

- Continuous integration
- Refactoring or design improvement
- Small releases

Shared understanding

- Coding standards
- Collective code ownership
- Simple design
- System metaphor

Programmer welfare

- Sustainable pace

XP: Rules

Planning

- User stories are written.
- Release planning creates the release schedule.
- Make frequent small releases.
- The project is divided into iterations.
- Iteration planning starts each iteration.

Managing

- Give the team a dedicated open work space.
- Set a sustainable pace.
- A stand up meeting starts each day.
- The Project Velocity is measured.
- Move people around.
- Fix XP when it breaks.

Design

- Simplicity.
- Choose a system metaphor.
- Use CRC cards for design sessions.
- Create spike solutions to reduce risk.
- No functionality is added early.
- Refactor whenever and wherever possible.

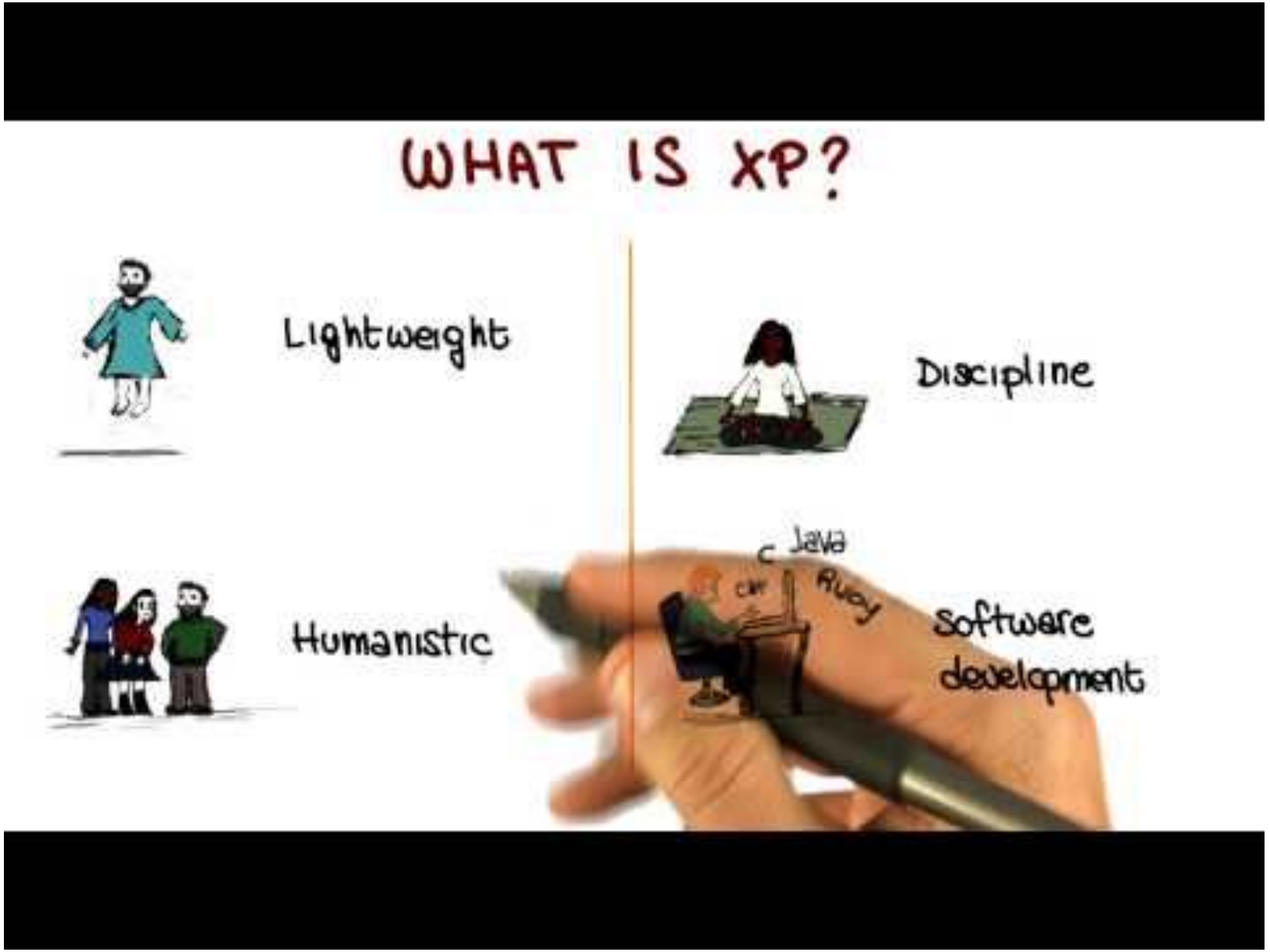
Coding

- The customer is always available.
- Code must be written to agreed standards.
- Code the unit test first.
- All production code is pair programmed.
- Only one pair integrates code at a time.
- Integrate often.
- Set up a dedicated integration computer.
- Use collective ownership.

Testing

- All code must have unit tests.
- All code must pass all unit tests before it can be released.
- When a bug is found tests are created.
- Acceptance tests are run often and the score is published.

eXtreme Programming (video)



Source: https://www.udacity.com/course/viewer#!/c-ud805/l-1771718546/m-439898982

XP: Discussion

- What are the advantages of adopting XP?
- What are the disadvantages of adopting XP?

Questions

Recommended Reading

- Beck, K. (2004). *Extreme programming eXplained*. 2nd ed. Reading, MA: Addison-Wesley.
- Extreme Programming: A gentle introduction, (2015). *The Rules of Extreme Programming*. [online] Available at: http://www.extremeprogramming.org/rules.html [Accessed 10 Nov. 2015].
- Georgia Tech (2015). *Software Development Processes*. [online] Udacity.com. Available at: https://www.udacity.com/course/software-development-process--ud805 [Accessed 10 Nov. 2015].
- Wikipedia, (2015). *Extreme programming*. [online] Available at: https://en.wikipedia.org/wiki/Extreme_programming [Accessed 10 Nov. 2015].

- *Google…*