AGILE METHODOLOGIES: XP

Lecture:
Extreme programming

AGENDA

1. Extreme Programming (XP) introduction
2. XP values
3. XP principles
4. XP rules

XP: ITERATIVE AND INCREMENTAL

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.

XP: AGILE

Agile Manifesto

"*Individuals and interactions over processes and tools*
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on
the right, we value the items on the left more"

XP: HISTORY

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.

XP: IDEA

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

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

For example:

- Code review improves code quality → make code review continuous → 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
- Sustainable pace

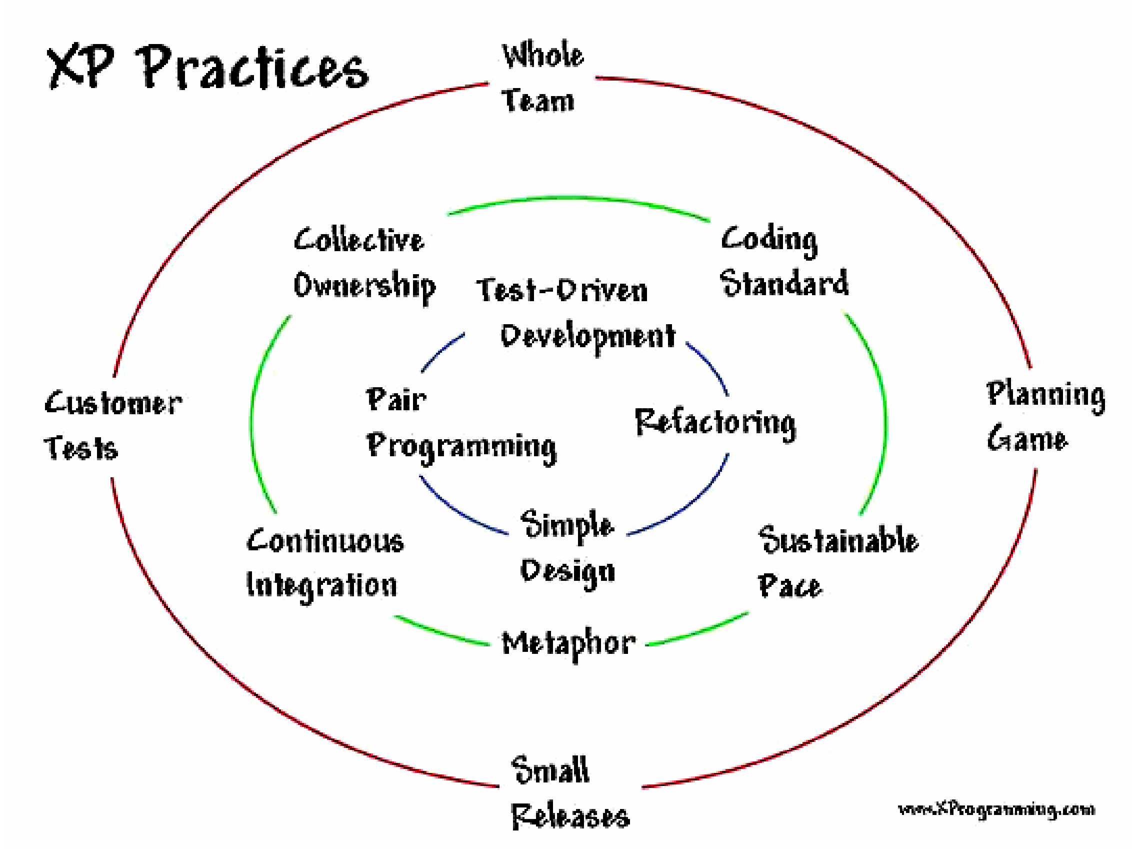
Shared understanding

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

Programmer welfare

- Sustainable pace

XP: PRACTICES



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.

XP: RULES

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 (to guard against it coming back).
- Acceptance tests are run often and the score is published.

XP: ROLES

Tracker

Customer

Programmer

Tester

Coach

XP: ROLES

Tracker – detects problems with product development and traces progress.

Optional member of the team. Normally one of the developers takes on the role of the tracker. Trackers use different metrics to make sure the project is viable, sustainable, meets KPIs etc.

XP: ROLES

Customer – sets priorities, describes business processes, provides scenarios (stories)

Customer must be actively engaged in decision making and development of the product. Customer role can be shared by more than one person.

XP: ROLES

Programmer – codes the product

Programmers or developers are responsible for realising stories into the computer programme. Programmers are very important part of the team and in fact many other roles share some of the programmers tasks.

XP: ROLES

Tester – person responsible for the quality of the final product through testing its components

XP: ROLES

Coach – monitors team work, motivates to follow effective practices, controls the process

A person with previous XP experience. The main role of the coach is to make sure the principles of XP are followed by the team members in a most efficient way so that the project genuinely follows the XP methodology.

XP: USE OF XP SURVEY

- Almost all of the projects were rated successful.
- 100% of the asked developers would reuse XP in the next project, when appropriate.
- The frequent absence of the customer was identified as high project risk.
- Problems with XP often come from “barriers in the mind”: management was skeptic, company philosophy didn’t allow on-site customer, developers refused pair programming.
- Most useful XP elements were common code ownership, testing and continuous integration. Most critical metaphor and on-site customer.
- As most important success factors have been mentioned: testing, pair programming and the focus of XP on the right goals.

(https://arxiv.org/ftp/arxiv/papers/1409/1409.6599.pdf)

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…