Extreme Programming Explained: Embrace Change

Kasra Madadipouya

Table of content

Introduction to XP

- XP stands for e**X**treme **P**rogramming, is
  - Created by *Kent Beck* during his work on the Chrysler Comprehensive Compensation System (C3) payroll project (1996-1999).
  - A set of principles and practices for rapid development of software.
  - An Agile Development Methodology with the focus on collaborative & iterative development.
- Why is it called “Extreme Programming”?
  - Named because it takes 12 well-known software “best practices” to their logical extremes.

XP core values

1. ***Communication --*** Team members work face-to-face and each member is involved in all aspects of development. Developers, Managers, Customers should communicate.
2. ***Simplicity --*** Better do a simple thing today and pay a little more to change it tomorrow. Forget about “You’re gonna need it later” mindset.
3. ***Feedback --*** Software is demonstrated early (Programmers immediately estimate time, customers & testers write functional tests) with frequent iteration (~ every 2 weeks).
4. ***Courage --*** Throw parts of the code not required, start again, refactor, improve the code.
5. ***Respect --*** Team members respect each other as developers, and everyone works to contribute value. Developers respect the expertise of the customer, and management respects the developers’ knowledge and responsibility over their work.

13 best practices about XP (1-4)

1. ***Planning Game --*** Customers decide on scope and timing of releases according to programmers estimates. Programmers ***Only*** implements demanded functionalities.
2. ***Small Releases --*** New product should be put into production in a few months, before completion of the program. New releases are made often—anywhere from ***daily*** to ***monthly***.
3. ***Metaphor --*** The shape of the system is defined by a metaphor or set of metaphors shared between the customer and programmers.
4. ***Simple Design --*** At every moment, the design runs all the tests, communicates everything the programmers want to communicate, contains no duplicate code, and has the fewest possible classes and methods. This rule can be summarized as, “Say everything once and only once.”

13 best practices about XP (5-8)

5. ***Tests*** **--** Programmers write unit tests minute by minute, customers write functional test. These tests are collected and they must all run correctly.
6. ***Refactoring*** **--** The design of the system is evolved through transformations of the existing design that keep all the tests running.
7. ***Pair Programming*** **--** ***two*** people at ***one*** screen/keyboard/mouse.
8. ***Continuous Integration*** **--** New code is integrated with the current system after no more than a few hours to built the system from ***scratch***.

13 best practices about XP (9-13)

9. ***Collective Ownership --*** Every programmer improves any code anywhere in the system at any time if they see the opportunity (***The Boy Scout Rule***).
10. ***On-site Customer --*** A customer sits with the team full-time.
11. ***40-hour Weeks --*** No one can work a second consecutive week of overtime.
12. ***Open Workspace --*** The team works in a large room with small cubicles around the periphery.
13. **** Just Rules --*** They're just the rules. The team can change the rules at any time as long as they agree on how they will assess the effects of the change.

Benefits & limitations

- ***Benefit* --** XP improves quality of the code, productivity and morale compare to traditional software development methods.
- ***Limitation*** --XP has scalability issue in terms of team size. It is efficient with teams of less than 25 or so.

Q & A

Thank you

You can find me at kasra@madadipouya.com