04834580 Software Engineering (Honor Track) 2025-26

Agile Methods

Sergey Mechtaev
mechtaev@pku.edu.cn
School of Computer Science, Peking University

The Agile Manifesto [1]

Formulated philosophy behind agile methods:

- value **individuals and interactions** over processes and tools;
- value **working software** over comprehensive documentation;
- value **customer collaboration** over contract negotiation;
- value **responding to change** over following a plan.

Among best-known agile methods are **extreme programming (XP)** and **Scrum**:

XP introduces practices for reducing the costs of ongoing changes [2];

Scrum enables development teams to operate adaptively within a complex environment using imprecise processes [3].

How Extreme Programming Addresses Risks? [2]

Schedule slips — sets short release cycles, implements the highest priority features first.

Project canceled — asks the customer to choose the smallest release that makes the most business sense.

System degrades over time — maintains a regression test suite.

Defect rate — developers create tests function-by- function, customers write tests program-feature-by-program-feature.

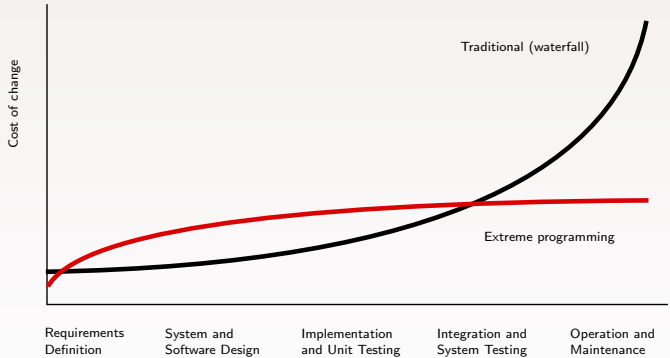
Business misunderstood — the customer is part of the team.

Business changes — shortens the release cycle.

Features irrelevant to customers — addresses tasks based on priority.

Staff turnover — lets programmers estimate and complete their own work, encourages human contact.

A key goal of extreme programming is to **flatten the change cost curve**.



> *Neither business considerations nor technical considerations should be paramount. Software development is always an evolving dialog between the possible and the desirable. [2]*

Business people decide **scope**, **priority**, **composition of releases**, **dates of releases**.

Technical people decide **estimates**, **consequences**, **process**, **detailed scheduling**.

Definition (by Mike Cohn [4])

A user story describes functionality that will be valuable to either a user or purchaser of a system or software. User stories are composed of three aspects:

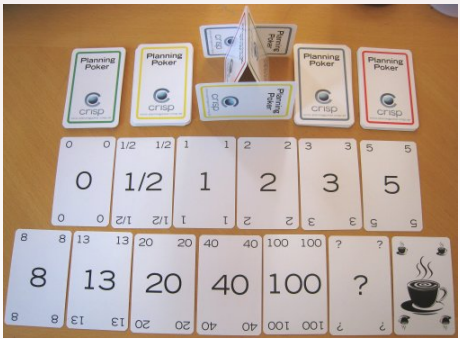
- a written description of the story used for planning and as a reminder
- conversations about the story that serve to flesh out the details of the story
- tests that convey and document details and that can be used to determine when a story is complete

Example user story:

> *As a user, I can indicate folders not to back up so that my backup drive is not filled up with things I do not need to be saved.*

Rationale: avoid the influence of the other participants.

1. Overview of a user story to be estimated is given.
2. Each individual chooses a card with estimate.
3. Everyone turns their cards simultaneously.
4. Discussion and justification.
5. Repeat until a consensus is reached.

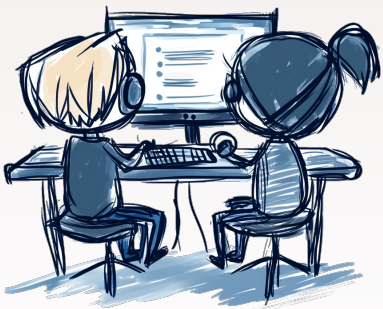


Code is written with two people looking at one machine, with one keyboard and one mouse.

Driver writes code;

Navigator reviews each line of code as it is typed in.

The two programmers switch roles frequently.

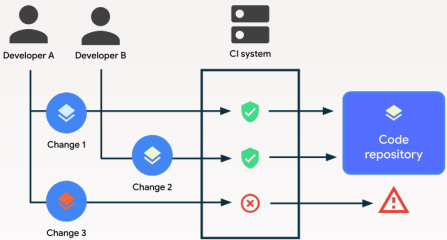


Research shows that pair programming is beneficial for achieving correctness on highly complex programming tasks [6].

The term is proposed by Grady Booch in 1991 [7].

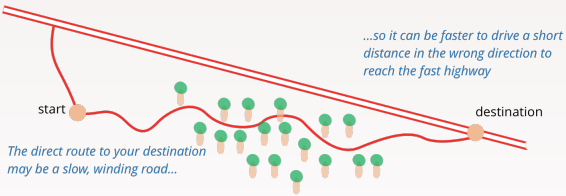
Code is integrated and tested after a few hours — a day of development at most.

- automated build;
- automated test;
- automated code analysis;
- automated deployment;



Definition (by Martin Fowler [8])

A change made to the internal structure of software to make it easier to understand and cheaper to modify without changing its observable behavior.



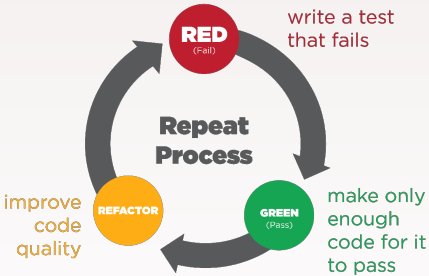
A metaphor by Jessica Kerr: "It's like I want to go 100 miles east but instead of just traipsing through the woods, I'm going to drive 20 miles north to the highway and then I'm going to go 100 miles east at three times the speed I could have if I just went straight there." [9]

Write test cases before you write code.

Red — write a test that fails;

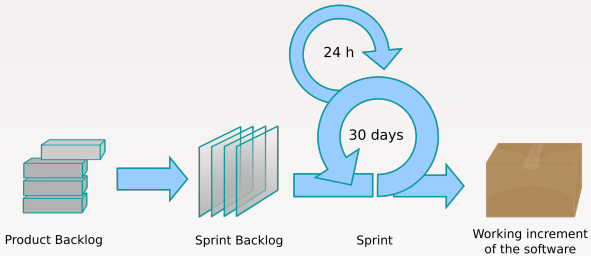
Green — write code that makes the test pass;

Refactor — improve code structure.



The discomfort of standing for long periods helps to keep the meetings short. [11]

Introduced by Hirotaka Takeuchi and Ikujiro Nonaka in 1986 [12].

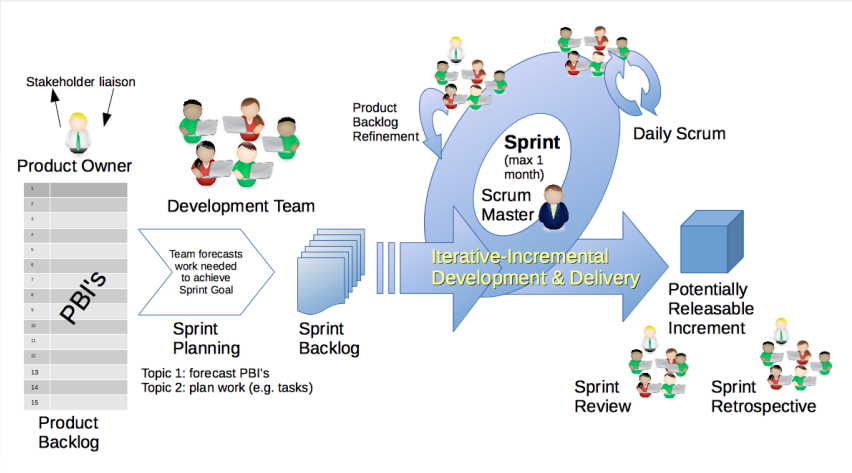


Roles in a **scrum team**:

Product owner — focuses on the business side of product, liaising with stakeholders.

Scrum master — coaching, objective setting, problem solving, oversight, planning, backlog management, and communication facilitation.

Developers — development and support of the product.



Note: PBI in the figure refers to product backlog item.

References I

[1] Kent Beck, Mike Beedle, Arie Van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, et al.
The agile manifesto.
http://agilemanifesto.org/, 2001.
[Online; accessed 27-Jan-2025].

[2] Kent Beck.
Extreme programming explained: embrace change.
addison-wesley professional, 2000.

[3] Ken Schwaber.
Scrum development process.
In *Business object design and implementation: OOPSLA'95 workshop proceedings 16 October 1995, Austin, Texas*, pages 117–134. Springer, 1997.

References II

[4] Mike Cohn.
User stories applied: For agile software development.
Addison-Wesley Professional, 2004.

[5] Mike Cohn.
Agile estimating and planning.
Pearson Education, 2005.

[6] Jo E Hannay, Tore Dybå, Erik Arisholm, and Dag IK Sjøberg.
The effectiveness of pair programming: A meta-analysis.
Information and software technology, 51(7):1110–1122, 2009.

[7] Grady Booch.
Object oriented design with applications.
Benjamin-Cummings Publishing Co., Inc., 1990.

[8] Martin Fowler.
Refactoring: improving the design of existing code.
Addison-Wesley Professional, 2018.

[9] Martin Fowler.
An example of preparatory refactoring.
https://martinfowler.com/articles/preparatory-refactoring-example.html, 2015.
[Online; accessed 27-Jan-2025].

[10] Kent Beck.
Test driven development: By example.
Addison-Wesley Professional, 2022.

[11] Wikipedia authors.
Stand-up meeting.
https://en.wikipedia.org/wiki/Stand-up_meeting, 2025.
[Online; accessed 27-Jan-2025].

[12] Hirotaka Takeuchi and Ikujiro Nonaka.
The new new product development game.
Harvard business review, 64(1):137–146, 1986.