Extreme Programming

Extreme Programming (XP) was founded by Kent Beck in the late nineties and is an agile methodology best-suited for small/medium sized teams. XP aims to reduce costs that arise as a result of changes in requirements by implementing short cycles. In XP, features are only implemented when needed. XP has three elements: pair programming, test-driven development and extensive code review. Test-Driven Development (TDD) relies on unit testing. Test cases are developed for each requirement, prior to the coding. Automated testing may be implemented to help aid the rapid development. Examples of automated testing include Selenium, which is used for web tests. XP has five values:

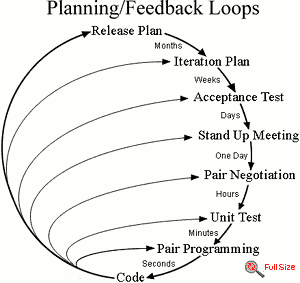
Simplicity: During the requirements phase of the project, use cases are developed from the requirements gathered to identify the features needed. These features are them simplified into manageable iterations. Features are only developed when they are required; this prevents unnecessary costs.

Communication: Pair Programming is encouraged by XP and enforces communication among team members. According to a study carried out at the University of Utah, two programmers working on a single application produces the same amount of work as two working independently, but produce software of a higher quality. In pair programming, both programmers are equal, despite any differences in level of expertise or experience. One programmer codes, whilst the other reviews the code, before switching roles, after a few hours.

Feedback: Through iterative delivery, XP allows customers to express any changes to requirements or any additional requirements. This makes the final product less likely to be rejected by users, as any mistakes are corrected in the early stages.

Respect: Since XP relies on a collaborative effort, respect amongst the managers, development team and customers are vital for on-going collaboration.

Courage: The development team is honest about estimates and progress made; any failures are not feared, as members never work alone on anything.

XP Lifecycle3

During the release plan, requirements are gathered by interviewing stakeholders (or customers) and writing user stories. Iteration planning follows the release plan where user stories are broken down into manageable iterations. The features undergo acceptance testing, which examines how well the implementation has met user requirements. Stand-up meetings are held regularly to update all members on issues and possible solutions may be discussed. Pair negotiation is encouraged in XP to enforce team effort. Unit tests are used to test sections of a program. Pair programming enhances the quality of the software and code reviews are carried out, aiming to make the code more efficient, by refactoring, for example.

The Roles in XP

There are four major roles in XP: the developer, the customer, the manager (tracker), and coach. The developers hold several responsibilities, including estimating the time required to complete tasks, breaking down user stories into iterations (or tasks), write unit tests, write code that will pass the tests, refactoring and continuous integration. Developers are programmers, architects/designers, UI/UX designers, database designers, testers and network designers.

Customers influence what the final product. Customers may write detailed user stories and often write functional tests. Customers are also responsible for prioritising user stories. It is possible that the customer is a team of people, e.g. product managers, business analyst, salesman, and the end user.

The manager introduces rules of the planning game. The manager monitors the planning game and manages anything that does not conform to the rules. The manager may update rules, with regards to any changes in the project. The manager conducts the release planning and schedules meetings. The manager will also engage in estimating how long it will take to complete a particular task; the manager supports this by tracking and analysing previous estimates and actual time taken to complete those tasks, to help developers make a more accurate estimate.

The role of the coach is particularly crucial, when the team has no or little experience in extreme programming. The coach will identify XP practices that would benefit the project, when problems arise and advise the team on the practices.

5 Principles of XP3

* Rapid Feedback: mistakes or misunderstandings can be identified early. This is important as earlier mistakes tend to have a domino effect on future development.
* Assume Simplicity: Simplifying features into iterations allows the development team to obtain feedback faster on early developments.
* Incremental Change: Each increment will build on previous increments. Decisions are to be made during the course of the project; an incremental approach will allow for live feedback, which would aid decision-making.
* Embracing Change: Rapid Feedback, assuming simplicity, and incremental change allows XP to recognise and respond to change.
* Quality Work: Tests are written, prior to coding; this allows developers to test sections of their code. In XP, every line of code is tested, using Junit, for example. Code is easily maintained by having no line of code repeated elsewhere. Quality is more important than any delays there are in a project, as customers tend to forget late delivery, but will not forget poor quality of work.