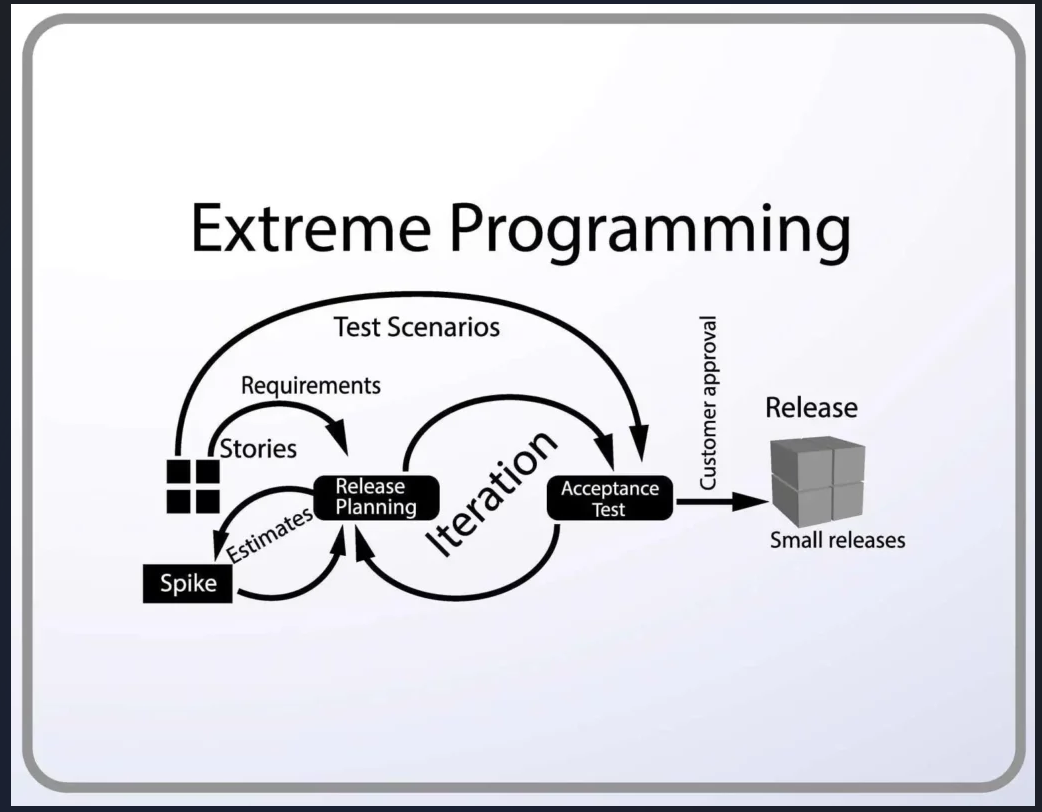
*EXTREME PROGRAMMING*

**Definition:**

Extreme Programming (XP) is an agile software development framework that aims to produce higher quality software, and higher quality of life for the development team. XP is the most specific of the agile frameworks regarding appropriate engineering practices for software development.

This type of methodology is used when customers are constantly changing demands or requirements, or when they are not sure about the system's performance.



**When Applicable?**

* Dynamically changing software requirements
* Risks caused by fixed time projects using new technology
* Small, co-located extended development team
* The technology you are using allows for automated unit and functional tests

Due to XP’s specificity when it comes to it’s full set of software engineering practices, there are several situations where you may not want to fully practice XP. The post When is XP Not Appropriate on the C2 Wiki is probably a good place to start to find examples where you may not want to use XP.

While you can’t use the entire XP framework in many situations, that shouldn’t stop you from using as many of the practices as possible given your context.

**Values:**

These values represent the fundamental principles that guide the decisions and behaviors of the development team. They are called "values" because they are the essential beliefs and principles that underpin the XP methodology. The five values of XP are communication, simplicity, feedback, courage, and respect and are described in more detail below.

**Communication:**

Software development is inherently a team sport that relies on communication to transfer knowledge from one team member to everyone else on the team. XP stresses the importance of the appropriate kind of communication – face to face discussion with the aid of a white board or other drawing mechanism.

**Simplicity:**

Simplicity means “what is the simplest thing that will work?” The purpose of this is to avoid waste and do only absolutely necessary things such as keep the design of the system as simple as possible so that it is easier to maintain, support, and revise. Simplicity also means address only the requirements that you know about; don’t try to predict the future.

**Feedback:**

Through constant feedback about their previous efforts, teams can identify areas for improvement and revise their practices. Feedback also supports simple design. Your team builds something, gathers feedback on your design and implementation, and then adjust your product going forward.

**Courage:**

This definition shows a preference for action based on other principles so that the results aren’t harmful to the team. You need courage to raise organizational issues that reduce your team’s effectiveness. You need courage to stop doing something that doesn’t work and try something else. You need courage to accept and act on feedback, even when it’s difficult to accept.

**Respect:**

The members of your team need to respect each other in order to communicate with each other, provide and accept feedback that honors your relationship, and to work together to identify simple designs and solutions.

**There 5 roles in Extreme Programming:**

* Tracker
* Customer
* Programmer.
* Tester
* Coach

**Advantages of eXtreme Programming:**

* Close contact with the customer
* No unnecessary programming work
* Stable software through continuous testing
* Error avoidance through pair programming
* No overtime, teams work at their own pace
* Changes can be made at short notice
* Code is clear and comprehensible at all times

**Disadvantages of eXtreme Programming:**

* Additional work
* Customer must participate in the process
* Relatively large time investment
* Relatively high costs
* Requires version management
* Requires self-discipline to practice