

# WHAT IS AGILE?



- The agile software development emphasizes on four core values.
- Individual and team interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

# Agile Methodology

- Agile is an iterative approach to project management and software development that helps teams deliver value to their customers faster and with fewer headaches.
- Instead of betting everything on a "big bang" launch, an agile team delivers work in small, but consumable, increments.
- Requirements, plans, and results are evaluated continuously so teams have a natural mechanism for responding to change quickly.

# Agile Manifesto

- The Agile Manifesto is the new bible for the software development industry. Its values and principles offer more concrete examples of how Agile software development should take place.
- To better understand the Agile Development, we need to briefly see what these values and principles mean, and assess whether they still matter.
- The Agile Development is standing on 4 core values and 12 principles. It is essential to understand these values and principles to understand the Agile approach clearly.

# Agile 4 core values

## Agile

### 4 Core Values



# Agile 12 Principles



**Customer  
Satisfaction  
First**



**Welcome  
Changing  
Requirements**



**Frequent  
Delivery**



**Business and  
Development  
Working  
Together**



**Motivated  
Individuals at  
the Heart of  
Projects**



**Face-to-Face is  
the Prime  
Communication  
Method**



**Working  
software is the  
primary  
measure of  
progress**



**Aim for  
Sustainable  
Progress**



**Continuous  
Attention to  
what Matters**



**Keep it Simple**



**Create  
Self-organizing  
Teams**



**Make Time for  
Reflection**

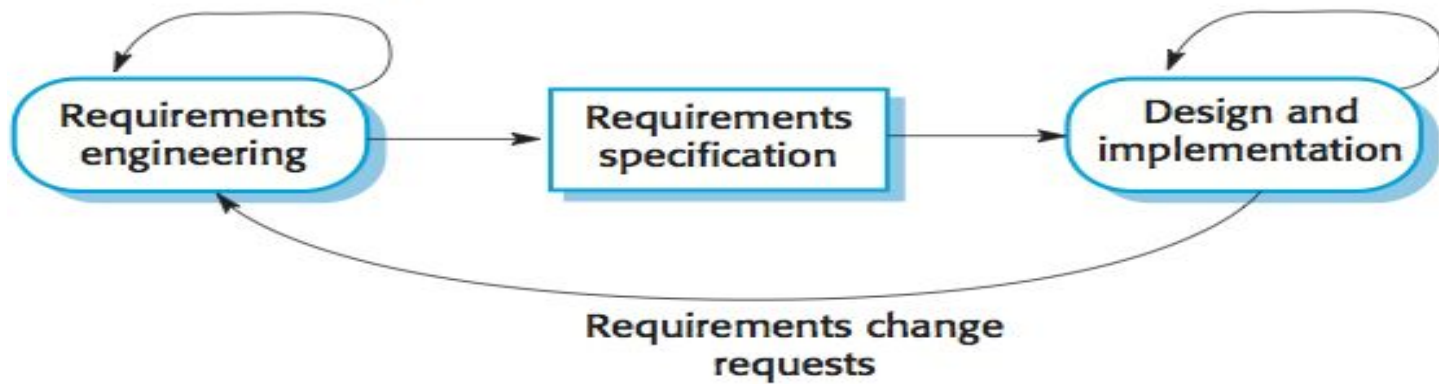
## The 12 Agile Principles

Taken from...

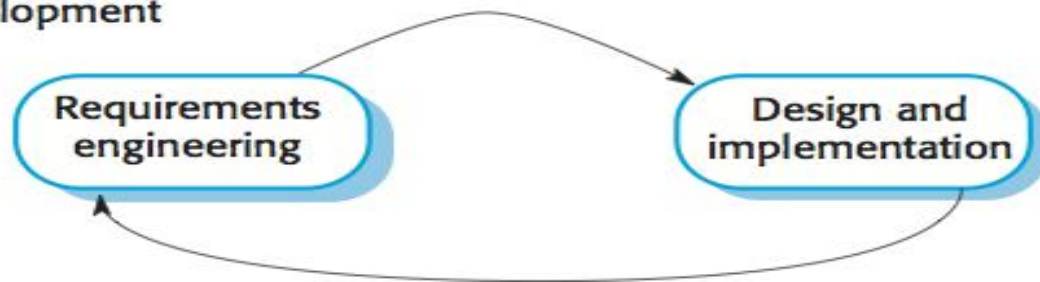
Principles behind the Agile Manifesto

[agilemanifesto.org/principles](http://agilemanifesto.org/principles)

### Plan-based development



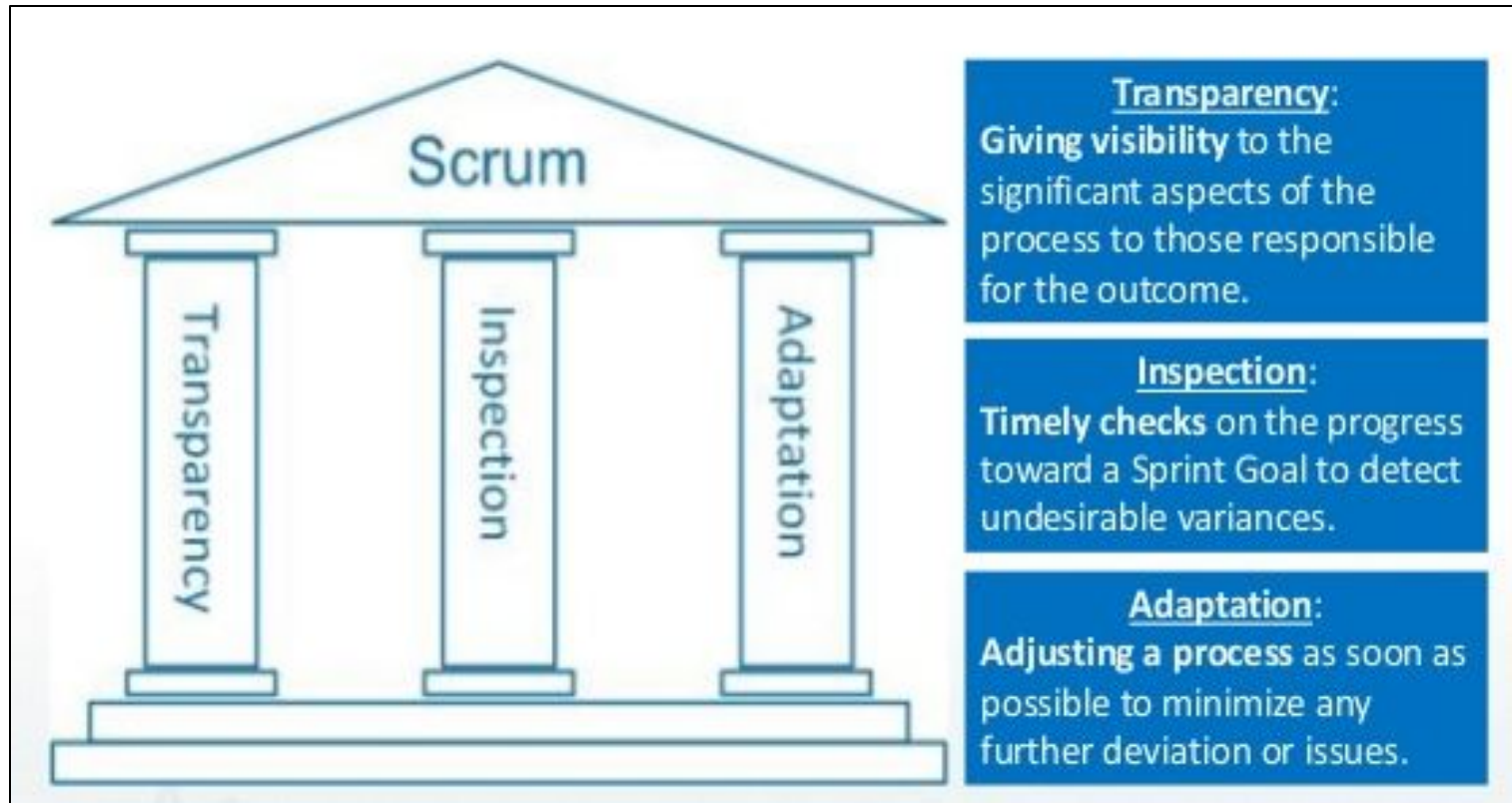
### Agile development



# SCRUM FRAMEWORK

- Scrum is a lightweight framework that helps people, teams and organizations generate value through adaptive solutions for complex problems.
- Scrum requires a Scrum Master to foster an environment where:
  - A **Product Owner** orders the work for a complex problem into a Product Backlog.
  - The **Scrum Team** turns a selection of the work into an Increment of value during a Sprint.
  - The Scrum Team and its stakeholders inspect the results and adjust for the next Sprint.
  - *Repeat*

# SCRUM THEORY





# SCRUM VALUES



## **COURAGE**

Scrum Team members have courage to do the right thing and work on tough problems

## **FOCUS**

Everyone focuses on the work of the Sprint and the goals of the Scrum Team

## **COMMITMENT**

People personally commit to achieving the goals of the Scrum Team

## **RESPECT**

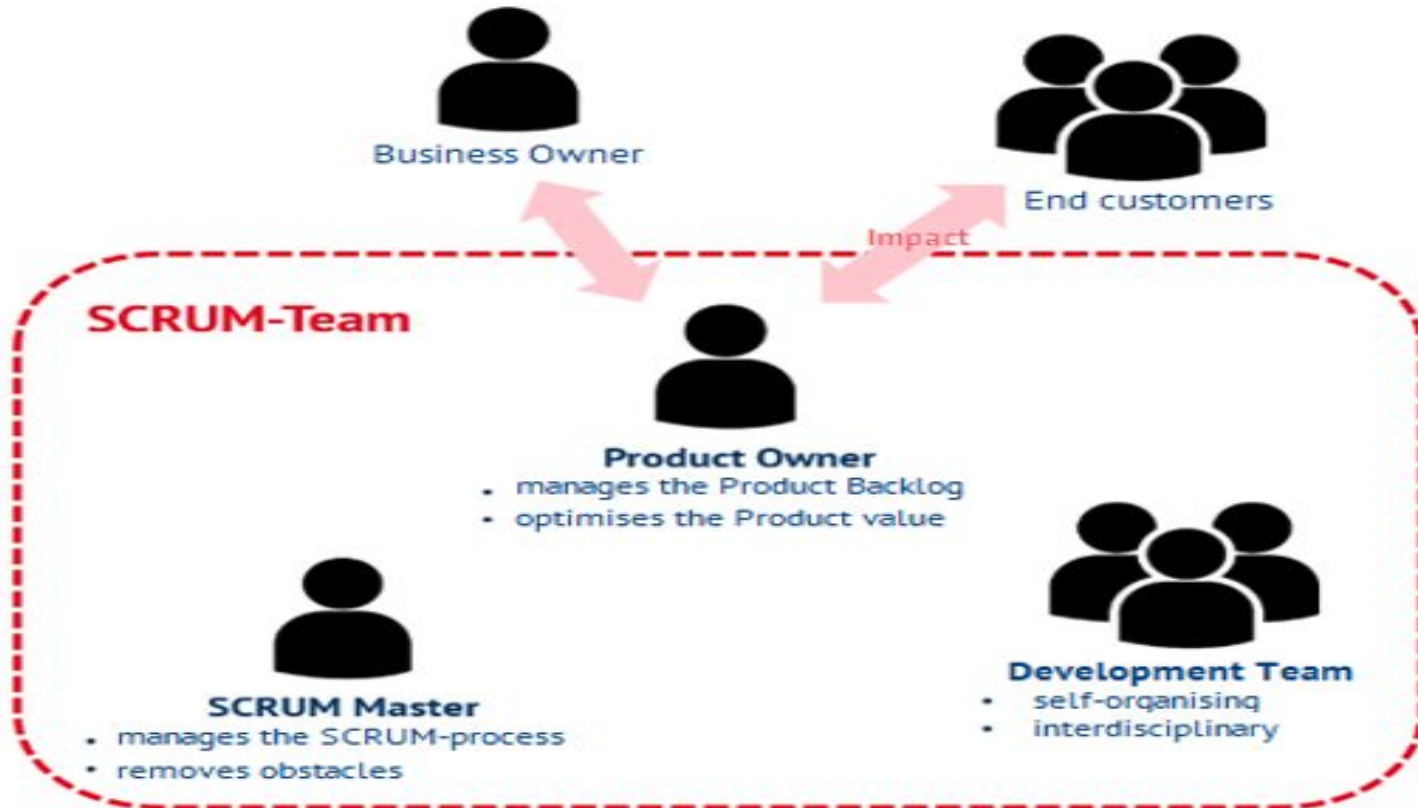
Scrum Team members respect each other to be capable, independent people

## **OPENNESS**

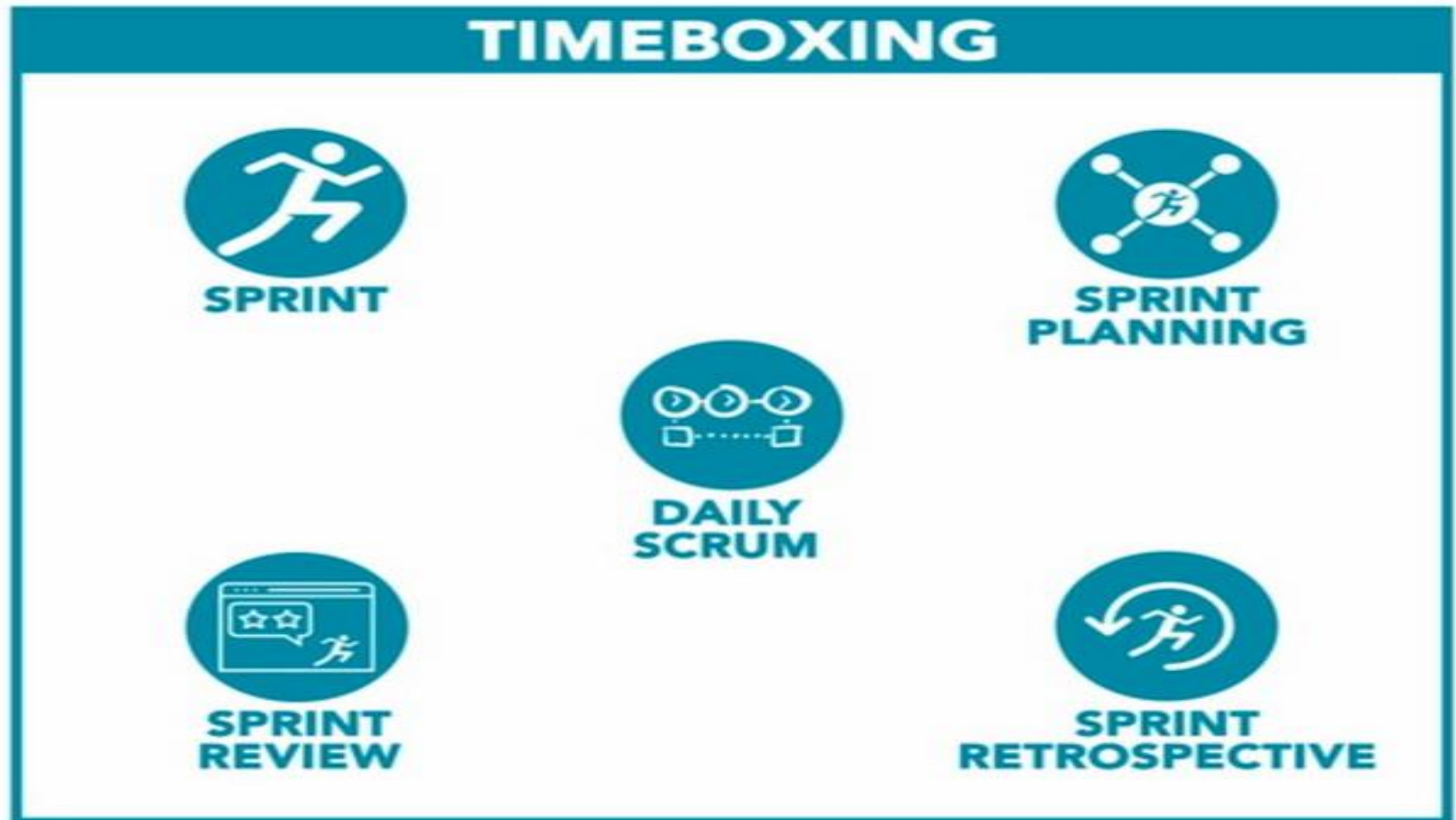
The Scrum Team and its stakeholders agree to be open about all the work and the challenges with performing the work

© Scrum.org

# SCRUM TEAM

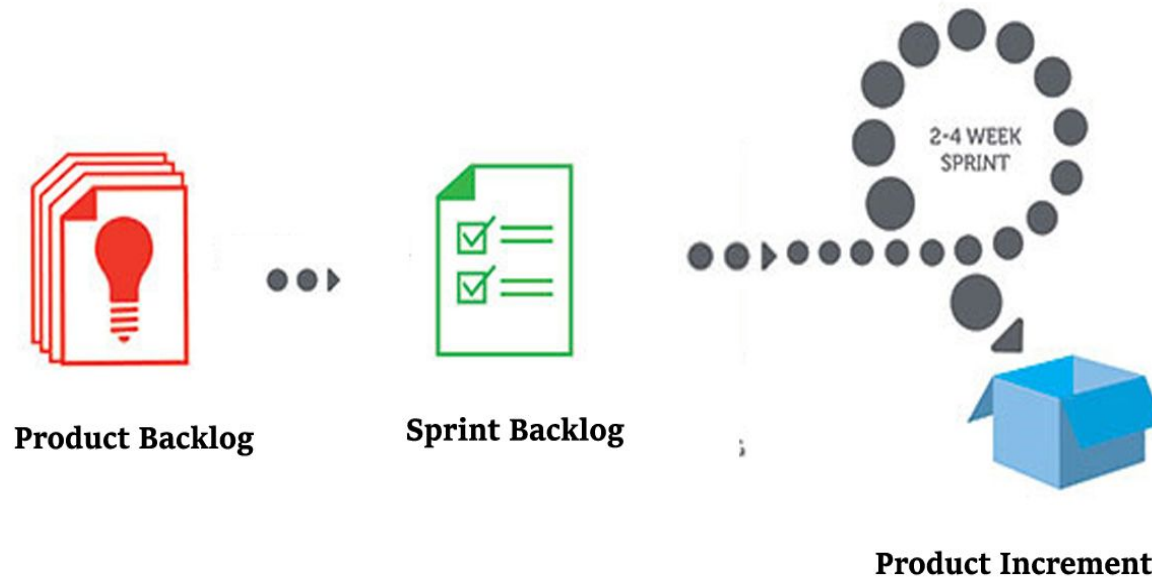


# SCRUM EVENT



# SCRUM ARTIFACTS

## Scrum Artifacts



# Kanban

- Kanban means “Visual Card” in Japanese.
- Kanban is a method applying just-in-time delivery while not overloading the team members.
- In Kanban, the process, from definition of a task to its delivery to the customer, is displayed for stakeholders to see.
- Team members pull work from a queue when they have excess capacity.

# Kanban

- Visualize the workflow
  - Kanban literally means "signboard" or "billboard"
- Just-in-time (JIT)
  - identify and eliminate wasteful activities, only build what you need
- Limit Work In Process (WIP)
  - establish and respect your ideal capacity, do what works best to get the work done
- Manage and optimize the flow/process
  - continually seeks ways to reduce the lead time for getting the work done
  - make process policies explicit
  - improve collaboratively

*It's not an inventory control system; it's a pull scheduling system that tells you what to produce, when to produce it, and how much to produce.*

## KANBAN BOARD

Stories



To Do



In Progress



Testing



Done

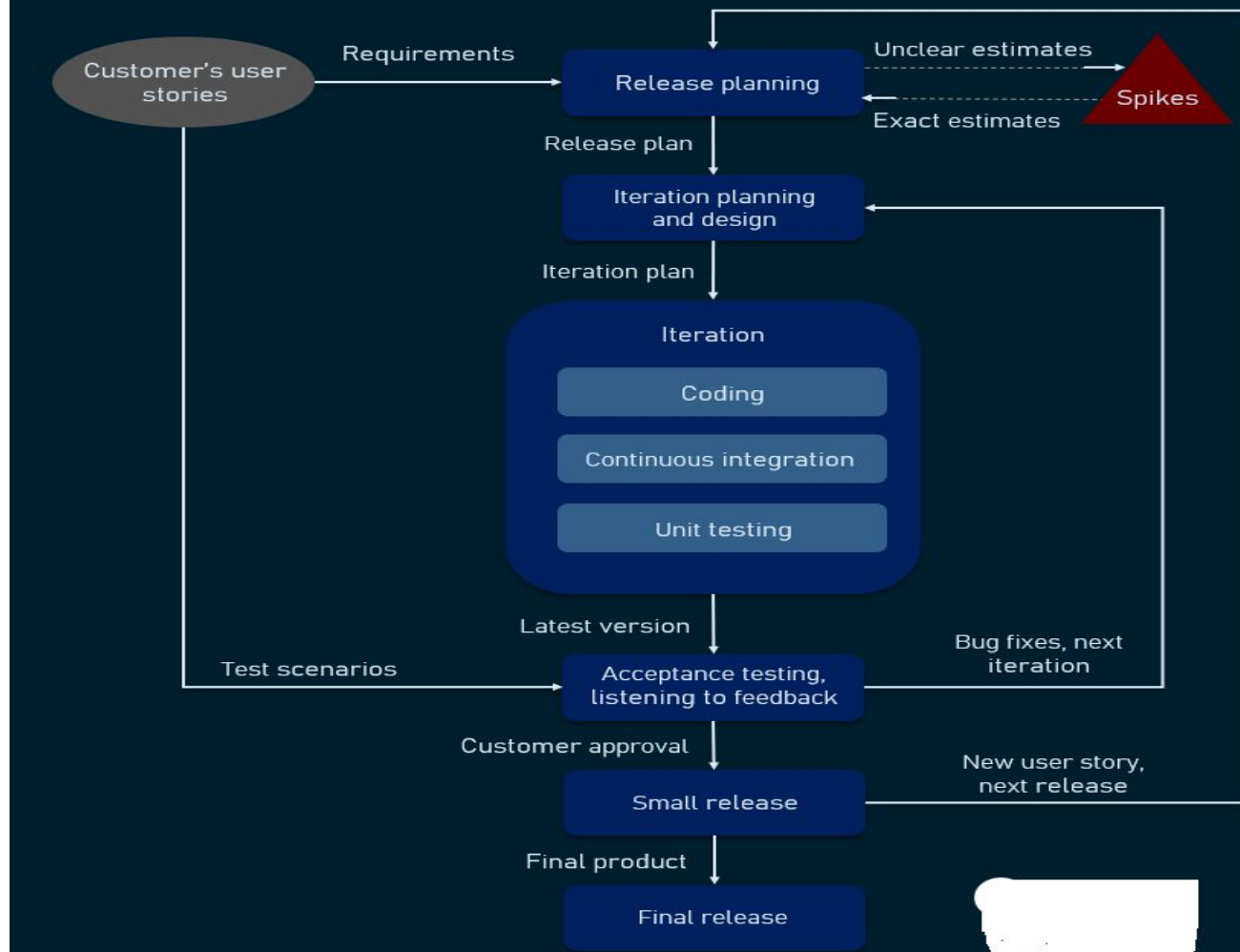


# Extreme Programming (XP)

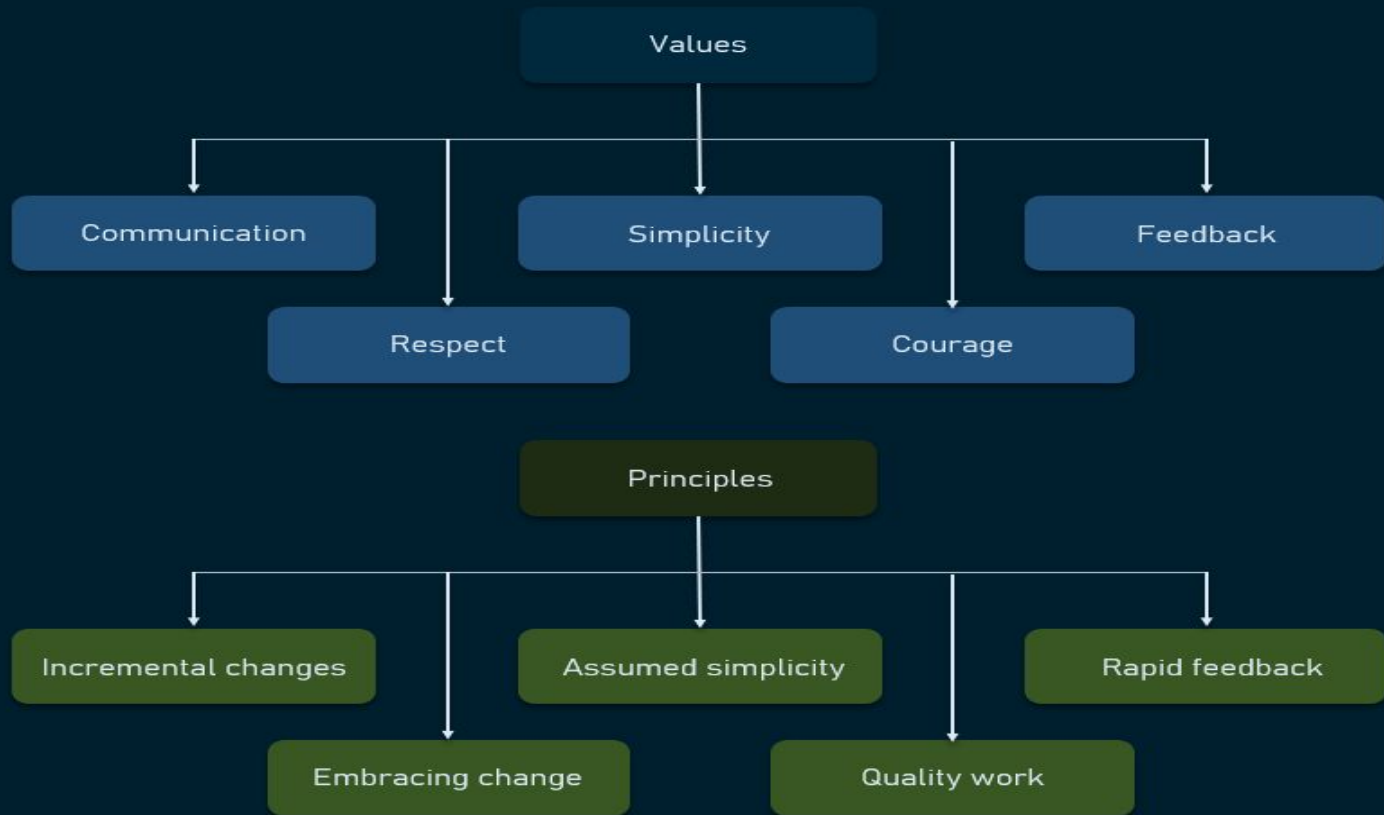
- Extreme Programming (also known as “XP”) is one of the numerous Agile frameworks applied by IT companies. But its key feature — emphasis on technical aspects of software development — distinguishes XP from the other approaches.



## EXTREME PROGRAMMING LIFECYCLE



# EXTREME PROGRAMMING VALUES AND PRINCIPLES



# Values of extreme programming

- XP has simple rules that are based on 5 values to guide the teamwork:
- **Communication.** Everyone on a team works jointly at every stage of the project.
- **Simplicity.** Developers strive to write simple code bringing more value to a product, as it saves time and effort.
- **Feedback.** Team members deliver software frequently, get feedback about it, and improve a product according to the new requirements.
- **Respect.** Every person assigned to a project contributes to a common goal.
- **Courage.** Programmers objectively evaluate their own results without making excuses and are always ready to respond to changes.

# Principles of extreme programming

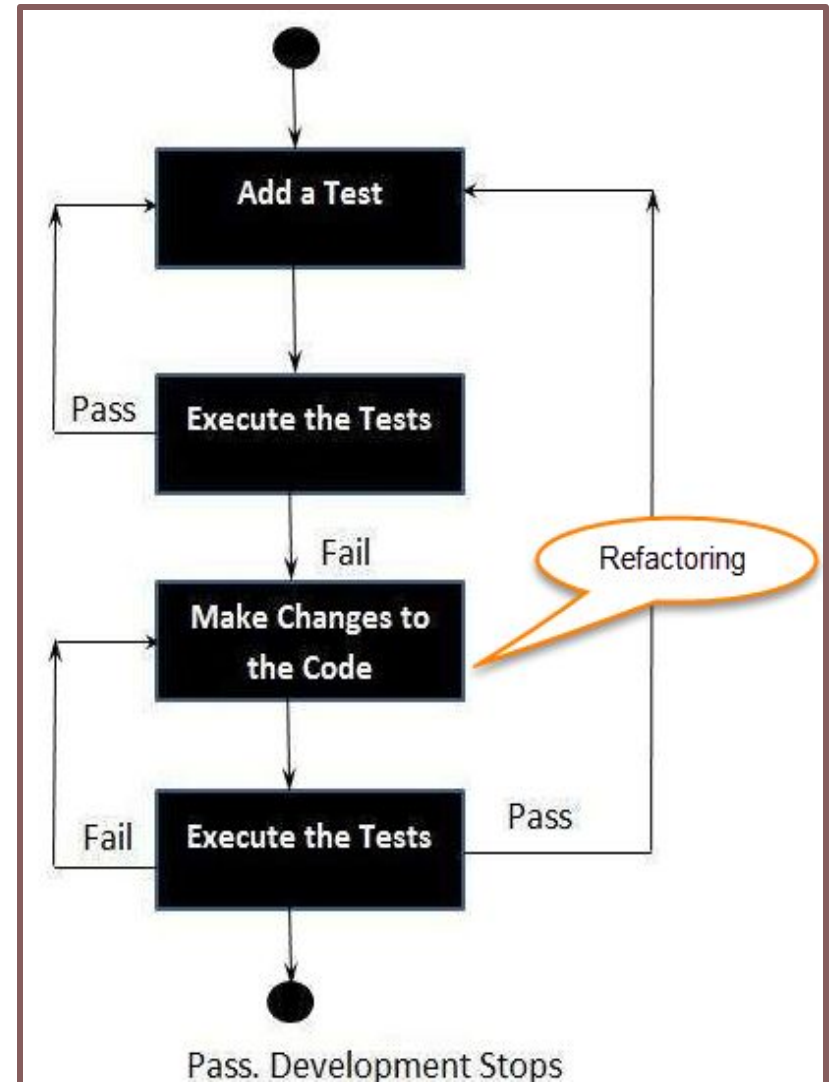
- Most researchers denote 5 XP principles as:
- **Rapid feedback.** Team members understand the given feedback and react to it right away.
- **Assumed simplicity.** Developers need to focus on the job that is important at the moment and follow YAGNI (You Ain't Gonna Need It) and DRY (Don't Repeat Yourself) principles.
- **Incremental changes.** Small changes made to a product step by step work better than big ones made at once.
- **Embracing change.** If a client thinks a product needs to be changed, programmers should support this decision and plan how to implement new requirements.
- **Quality work.** A team that works well, makes a valuable product and feels proud of it.

# Extreme programming practices

- Test-Driven Development
- The Planning Game
- Pair Programming
- Code Refactoring
- Continuous Integration
- Small Releases
- Simple Design
- Coding Standards
- Collective Code Ownership
- System Metaphor
- 40-Hour Week

# Test Driven Development(TDD)

- **Test Driven Development (TDD)** is software development approach in which test cases are developed to specify and validate what the code will do. In simple terms, test cases for each functionality are created and tested first and if the test fails then the new code is written in order to pass the test and making code simple and bug-free.



# Feature Driven Development (FDD)

- This method is focused around "designing & building" features. Unlike other agile methods, FDD describes very specific and short phases of work that has to be accomplished separately per feature. It includes domain walkthrough, design inspection, promote to build, code inspection and design.
- Five basic activities exist during FDD:
  - Develop overall model
  - Build feature list
  - Plan by feature
  - Design by feature
  - Build by feature

- **Predictive Software Development Life Cycle:**

- As the name suggests, predictive SDLC assumes you can predict the complete workflow. It involves fully understanding the final product and determining the process for delivering it. In this form of project life cycle, you determine the cost, scope, and timeline in the early phases of the project.
- One of the most common predictive models is the waterfall model. It assumes various phases in the SDLC that can occur sequentially, which implies that one phase leads into the next phase. In simple words, in waterfall model, all the phases take place one at a time and do not overlap one another.

- **Adaptive Software Development Life Cycle:**

- Adaptive SDLC approaches have a mix of incremental and iterative development. It involves adding features incrementally and making changes and refinements according to feedback. In other words, the work can easily adapt to the changing requirements based on new feedback received from the client.
- Agile and other iterative methodologies fall under the umbrella of adaptive SDLC.