UNIT 1

Agile Methodologies

Agile Definition

❖ *Agility* hasbecome today's buzzword when describing a modern software process. Everyone is agile.

- An agile team is a nimble team able to appropriately respond to changes. Change is what software development much about.

✔ Changes in the software being built,

✔ Changes to the team members,

✔ Changes because of new technology,

✔ Changes of all kinds that may have an impact on the product they build or the project that creates the product.

Agile Development

What is it?

- Agile software engineering combines a philosophy and a set of development guidelines.
- The philosophy encourages customer satisfaction and early incremental delivery of software; small, highly motivated project teams; informal methods; minimal software engineering work products; and overall development simplicity.
- The development guidelines stress delivery over analysis and design and active and communication between developers and customers.

Who does it?

- Software engineers and other project stakeholders (managers, customers, end users) work together on an agile team.

Why is it important?

- It helps to ensure that development teams complete projects on time and within budget.
- It improves communication between the development team and the product owner.
- It can help reduce the risks associated with complex projects.
- It applies to nearly every industry.

What are the steps?

- Agile development might best be termed "software engineering lite."
  The basic framework activities—

- ❖communication
- ❖planning
- ❖modeling
- ❖construction
- ❖deployment

What is the work product? "software increment" that is delivered to the customer on the appropriate commitment date.

How to Be Agile?

- To “be agile,” you need to put the agile values and principles into practice.
- Agile Methods
- Agile methods consist of individual elements called practices.
- Don’t Make Your Own Method
- Find a Mentor

Principles behind the Agile Manifesto

We follow these principles:

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Continuous attention to technical excellence and good design enhances agility.

Simplicity, the art of maximizing the amount of work not done, is essential.

The best architectures, requirements, and designs emerge from self-organizing teams.

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Find a Mentor

- Other groups practicing XP in your organization
- Other companies practicing XP in your area
- A local XP/Agile user group
- XP/Agile consultants
- The XP mailing list:

  extremeprogramming@yahoogroups.com

Traditional Methods vs. Agile Methods

Factors	Traditional Approaches	Agile Methods
Software Development	Process oriented	People oriented
Project Management Style	Command and Control	Leadership and Collaboration
Team Role Assignment	Based on skill level of individual team members	Self-organizing teams
Communication	Formal	Informal
Client's Role	Important	Critical
Process Model	Waterfall, Spiral, Prototype	Evolutionary approach
Project Lifecycle	Based on tasks or activities	Based on software product features

Different Types of Agile Frameworks

Kanban

- Kanban methodology is about day-to-day workflows and processes. It is a simple, visual means of managing projects that enables teams to see the progress so far and what's coming up next.

- Kanban projects are primarily managed through a Kanban board, which segments tasks into three columns: "To Do," "Doing," and "Done."

- The main benefit of this methodology is the increased transparency, allowing team leaders to clearly see which tasks are assigned to which team members and what is yet to be completed to make meaningful progress.

Scrum

- Scrum is one of the most popular Agile methodologies, as it can bring teams together with a sharp focus and an efficient, collaborative approach to task execution.
- It is similar to Kanban in many ways.
- Scrum typically uses a Scrum board, similar to a Kanban board, and groups tasks into columns based on progress.
- Unlike Kanban, Scrum focuses on breaking a project down into sprints and only planning and managing one sprint at a time.

- Sprints are the Scrum way of breaking projects down into iterations that can last anywhere between one and four weeks each.
- Bringing team members together from different departments, these sprints help you channel a collective focus to your projects.

Scrum also features a robust set of principles and activities that dictate how you work. These include:

- **Sprint planning:** Planning sessions to identify the purpose behind your sprints
- **Roles:** Key roles in the Scrum project management process
- **Product backlog:** A list of tasks arranged according to priority level

Feature-driven development (FDD)

- Feature-driven development is another software-specific Agile framework.
- This methodology involves creating software models every two weeks and requires a development and design plan for every model feature.
- It has more rigorous documentation requirements than XP, so it's better for teams with advanced design and planning abilities.

FDD breaks projects down into five basic activities:

- ❖Develop an overall model
- ❖Build a feature list
- ❖Plan by feature
- ❖Design by feature
- ❖Build by feature

Dynamic Systems Development Method (DSDM)

- The Dynamic Systems Development Method (DSDM) was born of the need for a common industry framework for rapid software delivery.
- Rework is to be expected, and any development changes that occur must be reversible.
- Like Scrum, XP, and FDD, DSDM uses sprints.

This framework is based on eight fundamental principles:
Focus on the business need

- ❖Deliver on time
- ❖Collaborate
- ❖Never compromise quality
- ❖Build incrementally from firm foundations
- ❖Develop iteratively
- ❖Communicate continuously and clearly
- ❖Demonstrate control

Lean

- Lean development is often grouped with Agile, but it's an entirely different methodology that happens to share many of the same values.
- The main Principles of the Lean Methodology include:

❖Eliminating waste
❖Build quality in
❖Create knowledge
❖Defer commitment
❖Deliver fast
❖Respect people
❖Optimize the whole

Crystal

- Crystal is a family of Agile methodologies that includes Crystal Clear, Crystal Yellow, Crystal Orange, Crystal Red, etc.
- Each has a unique framework.
- Your choice depends on several project factors, such as your team size, priorities, and project criticality.

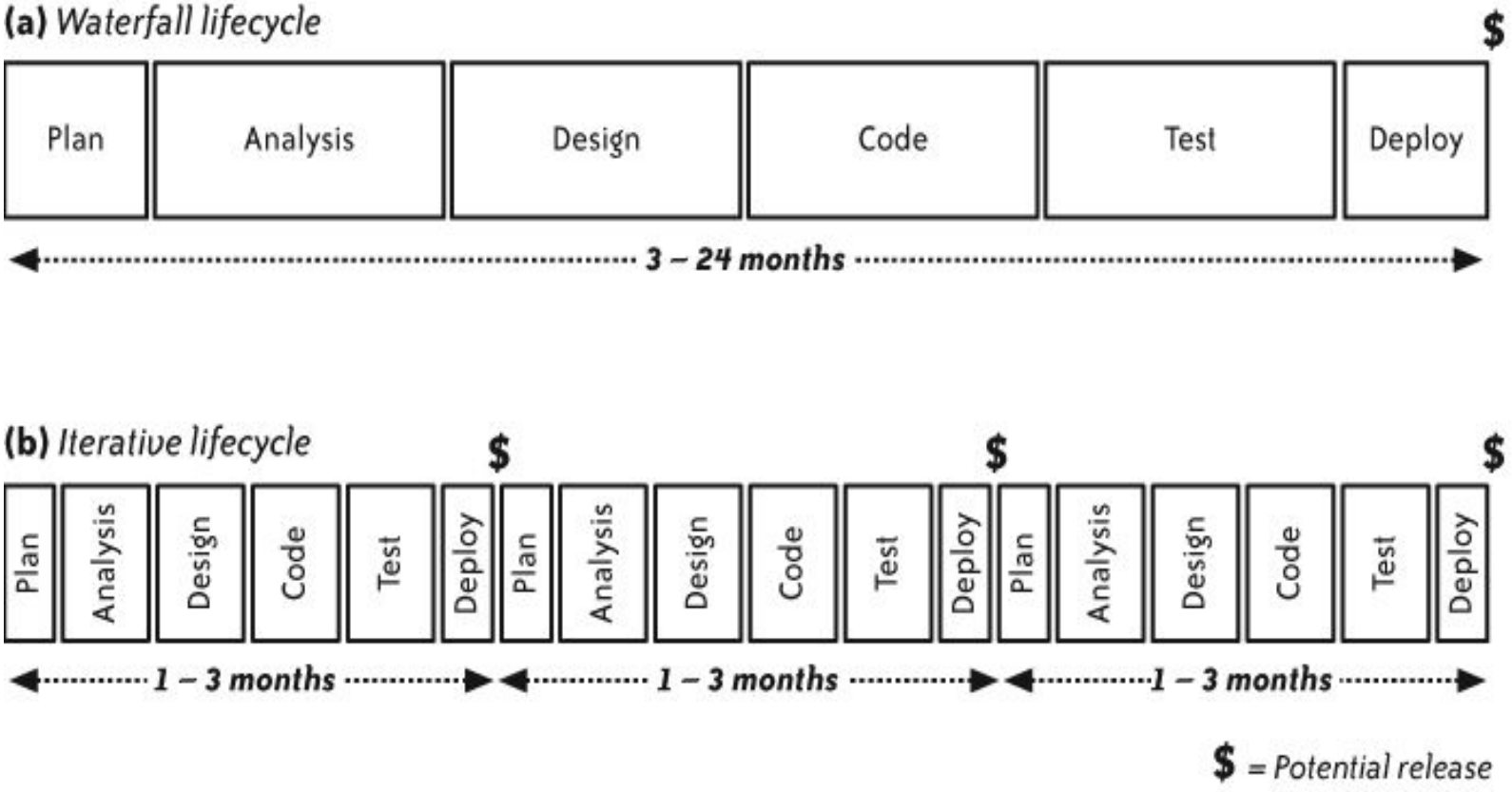
Extreme Programming (XP)

- Extreme Programming (XP) was designed for Agile software development projects.
- It focuses on continuous development and customer delivery and uses intervals or sprints, similar to a Scrum methodology.
- However, XP also has 12 supporting processes specific to the world of software development:

- Planning game
- Small releases
- Customer acceptance tests
- Simple design
- Pair programming
- Test-driven development
- Refactoring
- Continuous integration
- Collective code ownership
- Coding standards
- Metaphor
- Sustainable pace

Extreme Programming puts the customer at the center of everything the team does. In XP, teams can involve the end users in a meaningful way and use the feedback they get to deliver the best possible product.

Understanding XP



The XP Lifecycle

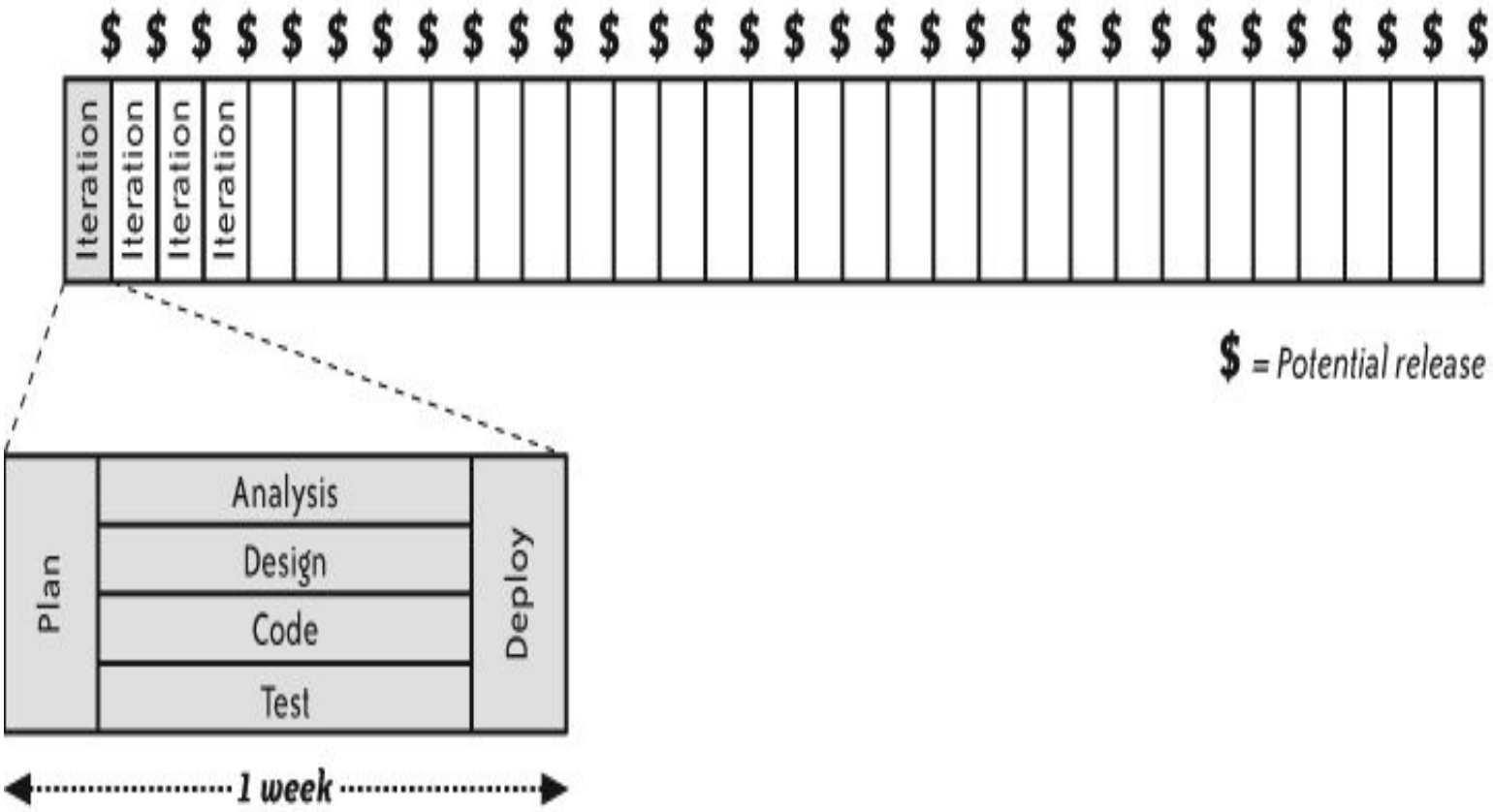


Figure. XP lifecycle

The XP Lifecycle

- XP teams work on activities every day.
- XP team produces deployable software every week.
- In each iteration, the team analyzes, designs, codes, tests, and deploys a subset of features.
- XP emphasizes face-to-face collaboration. This is so effective in eliminating communication delays and misunderstandings.
- This allows them to work on all activities every day—with simultaneous phases.
- Team is more productive. Team gets feedback much more frequently.

How It Works

- XP teams perform nearly every software development activity simultaneously.
- XP does it by working in iterations: week-long increments of work.
- They work on stories:
- Delivering four to ten stories.
- they work on all phases of development for each story. At the end of the week, they deploy their software for internal review.

The XP Lifecycle

Planning

- Every XP team includes several business experts—the on-site customers.
- Project vision, creating stories, constructing a release plan, and managing risks.
- estimates and suggestions
- The planning effort is most intense during the first few weeks of the project.
- In addition to the overall release plan, the team creates a detailed plan for the upcoming week at the beginning of each iteration.

The XP Lifecycle

Analysis

- On-site customers are responsible for figuring out the requirements for the software.
- They figure out the general requirements for a story before the programmers estimate it and the detailed requirements before the programmers implement it.
- Formalize the requirements.
- Ubiquitous language

The XP Lifecycle

Design and coding

- XP uses incremental design and architecture to continuously create and improve the design in small steps.
- Test-driven development
- To support this process, programmers work in pairs
- Programmers integrate their code every few hours and ensure that every integration is technically capable of deployment.
- programmers also maintain coding standards and share ownership of the code.

The XP Lifecycle

Testing

- Each member of the team makes his own contribution to software quality.
- Programmers provide the first line of defence with test-driven development.
- Customer tests help ensure that the programmers' intent matches customers' expectations.
- Testers help the team understand whether their efforts are in fact producing high quality code.

The XP Lifecycle

Deployment

- XP teams keep their software ready to deploy at the end of any iteration.
- They deploy the software to internal stakeholders every week in preparation for the weekly iteration demo.
- Deployment to real customers is scheduled according to business needs.

The XP Team

- How to design and program the software (programmers, designers, and architects)
- Why the software is important (product manager)
- The rules the software should follow (domain experts)
- How the software should behave (interaction designers)
- How the user interface should look (graphic designers)
- Where defects are likely to hide (testers)
- How to interact with the rest of the company (project manager)
- Where to improve work habits (coach)

The Whole Team

- XP teams sit together in an open workspace.
- At the beginning of each iteration, the team meets for a series of activities.
- These typically take two to four hours in total.
- The team also meets for daily stand-up meetings, which usually take five to ten minutes each.
- Team members work out the details of each meeting when they need to.

On-Site Customers

- Responsible for defining the software the team builds.
- The rest of the team can and should contribute suggestions and ideas
- Ultimately responsible for determining what stakeholders find valuable.
- Release planning, project's vision; identify features and stories; determine how to group features into small, frequent releases; manage risks.

On-Site Customers

- On-site customers may or may not be real customers, depending on the type of project.
- Responsible for refining their plans
- Responsible for providing programmers with requirements details.
- Typically, product managers, domain experts, interaction designers, and business analysts play the role of the on-site customer.
- Customer involvement makes a huge difference in product success.

The product manager

- Maintain and promote the product vision.
- Have deep understandings of their markets.
- Understanding of what the software will provide and why it’s the most important thing.
- Product manager is committed to the project full-time.
- Once a team is running smoothly, the product manager might start cutting back on his participation.

Domain experts

- Domain experts are responsible for figuring out domain rules.
- Domain experts, also known as subject matter experts, are experts in their field.
- Financial analysts and PhD chemists.
- Spend most of their time with the team, figuring out the details of upcoming stories and standing ready to answer questions when programmers ask.

Interaction designers

- Interaction designers help define the product UI.
- Focuses on understanding users, their needs, and how they will interact with the product.
- They perform such tasks as interviewing users, creating user personas, reviewing paper prototypes with users, and observing usage of actual software.
- Interaction designers divide their time between working with the team and working with users.
- Fill this role with a graphic designer, the product manager, or a programmer.

Business analysts

- On non agile team- Between the customers and developers, by clarifying and refining customer needs into a functional requirements specification.
- Clarify and refine customer needs.
- Express technical trade-offs in business terms.

Programmers

- A great product vision requires solid execution. The bulk of the XP team consists of software developers in a variety of specialties.
- Each of these developers contributes directly to creating working code.
- Minimize its cost.
- Finding the most effective way of delivering the stories.
- programmers provide effort estimates and suggest alternatives.

Programmers

- Programmers spend most of their time pair programming. Using test-driven development, they write tests, implement code, refactor, and architect the application.
- Establish coding standards
- programmers build their software to use a ubiquitous language.
- Help ensure the long-term maintainability of the product by providing documentation at appropriate times.

Designers and architects

- Everybody codes on an XP team, and everybody designs.
- Guiding the team's incremental design and architecture efforts.
- They act as peers— as programmers—rather than teachers
- Simplifying complex designs.

Technical specialists

- The XP “programmer” role includes other software development roles.
- Include a database designer, a security expert, or a network architect.
- XP programmers are generalizing specialists.

Testers

- Testers help XP teams produce quality results from the beginning.
- They help customers identify holes in the requirements and assist in customer testing.
- Testers also act as technical investigators for the team.
- Provide information about the software's non functional characteristics.
- XP teams don't include dedicated testers.

Coaches

- XP teams self-organize
- XP leaders are called coaches.
- A coach's work is subtle; it enables the team to succeed.
- Making sure that the team includes the right people.

Coaches

- Set up conditions for energized work, and they assist the team in creating an informative workspace.
- Help the team generate organizational trust and goodwill.
- Maintain their self-discipline, helping them remain in control of challenging practices.

The project manager

- Project managers help the team work with the rest of the organization.
- They are usually good at coaching non programming practices.

Team Size

- With 4 to 10 programmers (5 to 20 total team members). For new teams, four to six programmers is a good starting point.
- Team size of 12 people (6 programmers 4 customers, 1 tester, and a project manager)
- XP teams can be as small as one experienced programmer and one product manager
- Smallest team – XP team consists of five people

Team Size

- Four programmers (one acting as coach)and one product manager (who also acts as project manager, domain expert, and tester).
- Starting with 10 programmers produces a 20-person team that includes 6 customers, 3 testers, and a project manager.
- Reduce the overall productivity.

XP Concepts

- XP has its own vocabulary. XP uses this vocabulary.
  - Refactoring
  - Technical Debt
  - Time boxing
  - Stories
  - Iterations
  - Velocity
  - Mindfulness

Values and Principles

- Agile Manifesto, a collection of 4 values and 12 principles.

Values

- Individuals and Interactions Over Processes and Tools
- Working Software Over Comprehensive Documentation
- Customer Collaboration Over Contract Negotiation
- Responding to Change Over Following a Plan

Values and Principles

Principles

- Customer satisfaction through early and continuous software delivery
- Accommodate changing requirements throughout the development process
- Frequent delivery of working software
- Collaboration between the business stakeholders and developers throughout the project

Values and Principles

Principles

- Support, trust, and motivate the people involved
- Enable face-to-face interactions
- Working software is the primary measure of progress
- Agile processes to support a consistent development pace

Values and Principles

Principles

- Attention to technical detail and design enhances agility
- Simplicity
- Self-organizing teams encourage great architectures, requirements, and designs
- Regular reflections on how to become more effective

Values and Principles

XP Values

Values are ideals. They're abstract and distinct.

- Courage
- Communication
- Simplicity
- Feedback
- Respect

Values and Principles

- Principles are applications of those ideals to an industry.
- The value of simplicity leads us to focus on the essentials of development.
- Practices are principles applied to a specific type of project.

Improve the Process

- Understand Your Project
  - Feedback
  - Encourage open discussion.
- Tune and Adapt
- Break the Rules

Eliminate Waste

Work in Small, Reversible Steps

- The easiest way to reduce waste is to reduce the amount of work you may have to throw away.
- Incremental change
- Taking baby steps.
- Pair Programming
- Test Driven Development
- Refactoring

Eliminate Waste

Fail Fast

- Failure is another source of waste.
- Projects with no real risks are losers.
- Instead of trying to avoid failure, embrace it.
- Failing fast applies to all aspects of your work.

Eliminate Waste

Maximize Work Not Done

- Simplicity is the art of maximizing the work not done.
- Everything should be made as simple as possible, but not one bit simpler.
- Feedback, communication, self-discipline, and trust.
- Maximize the time you spend producing releasable software and improve the team's ability to focus on what's really important.

Eliminate Waste

Pursue Throughput

- –Unreleased software
- –Partially done work represents unrealized investment.
- –Partially done work also hurts throughput
- –Low throughput introduces more waste.
- –To minimize partially done work and wasted effort, maximize your throughput.

Deliver Value

Exploit Your Agility

- Improves your ability to recognize and take advantage of new opportunities.
- Agility requires you to work in small steps
- A small initial investment of time and resources, properly applied, begins producing quantifiable value immediately.
- Delivering value to your customer is your most important job.

Deliver Value

- Developing features closely with the on-site customer allows you to identify potential misunderstandings and provides nearly instant responses to questions.
- Including real customers in the process with frequent deliveries of the actual software demonstrates its current value to them.

Deliver Value

Only Releasable Code Has Value

- Code that meets customer needs perfectly has little value.
- Delivering actual value means delivering real software. Un releasable code has no value.
- Working software is the primary measure of your progress.
- Only code that you can actually release to customers can provide real feedback.

Deliver Value

Deliver Frequently

- Delivering working, valuable software frequently makes your software more valuable.
- The highest priority of any software project is to deliver value, frequently and continuously, and by doing so, to satisfy the customer.

_____Maintain and promote the product vision.

a)Product Manager

b)Programmer

c)Coach

d)Tester

Which of the following statement is correct?

a) XP teams work on activities every day.

b) XP team produces deployable software every week.

c) XP emphasizes face-to-face collaboration

d) All of the above

Incremental development in Extreme Programming (XP) is supported through a system release once every month.

a)True

b) False

In XP Increments are delivered to customers every _____ weeks.

a) One

b)Two

c) One week or two weeks

d) None

Which four framework activities are found in the Extreme Programming(XP) ?

a) analysis, design, coding, testing
b) planning, analysis, design, coding
c) planning, Analysis, design, coding, testing
d) planning, analysis, coding, testing

Select the option that suits the Manifesto for Agile Software Development

a) Individuals and interactions
b) Working software
c) Customer collaboration
d) All of the mentioned

Agile Software Development is based on

a) Incremental Development

b) Iterative Development

c) Linear Development

d) Both Incremental and Iterative Development

Which on of the following is not an agile method?

a) XP

b) 4GT

c)Scrum

d) All of the mentioned

Agility is defined as the ability of a project team to respond rapidly to a change.

a) True

b) False

_____is the process of changing the structure of code without changing its behavior.

a)Testing

b)Refactoring

c)Reverse Engineering

d)None