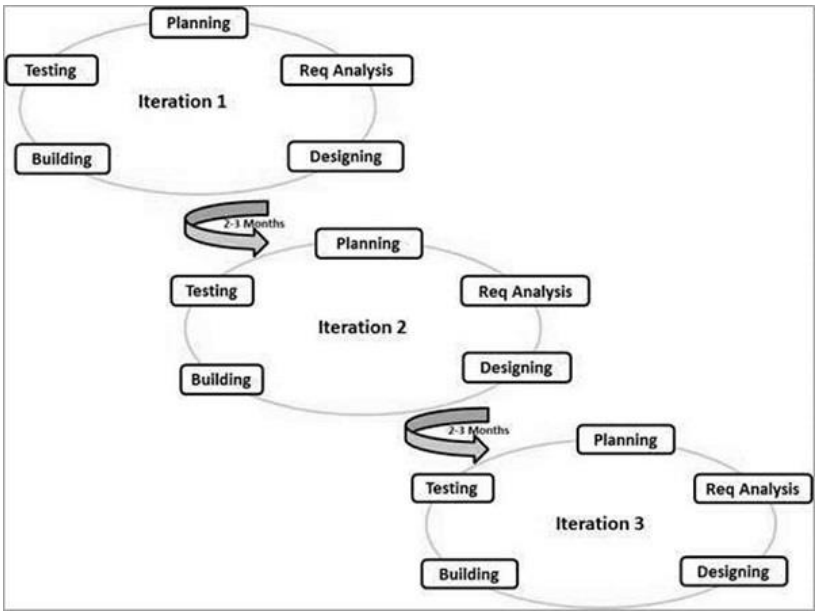
Unit 3: Software Engineering

AGILE methodology is a practice that promotes **continuous iteration** of development and testing throughout the software development lifecycle of the project.

In the Agile model, both development and testing activities are concurrent, unlike the Waterfall model.

This is a diagram to illustrate Agile Process-



Following are the Agile Manifesto principles −

- **Individuals and interactions** − In Agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.
- **Working software** − Demo working software is considered the best means of communication with the customers to understand their requirements, instead of just depending on documentation.
- **Customer collaboration** − As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.
- **Responding to change** − Agile Development is focused on quick responses to change and continuous development.

The **advantages of the Agile Model** are as follows −

- Is a very realistic approach to software development.
- Promotes teamwork and cross training.
- Suitable for fixed or changing requirements
- Delivers early partial working solutions.
- Easy to manage.
- Gives flexibility to developers.

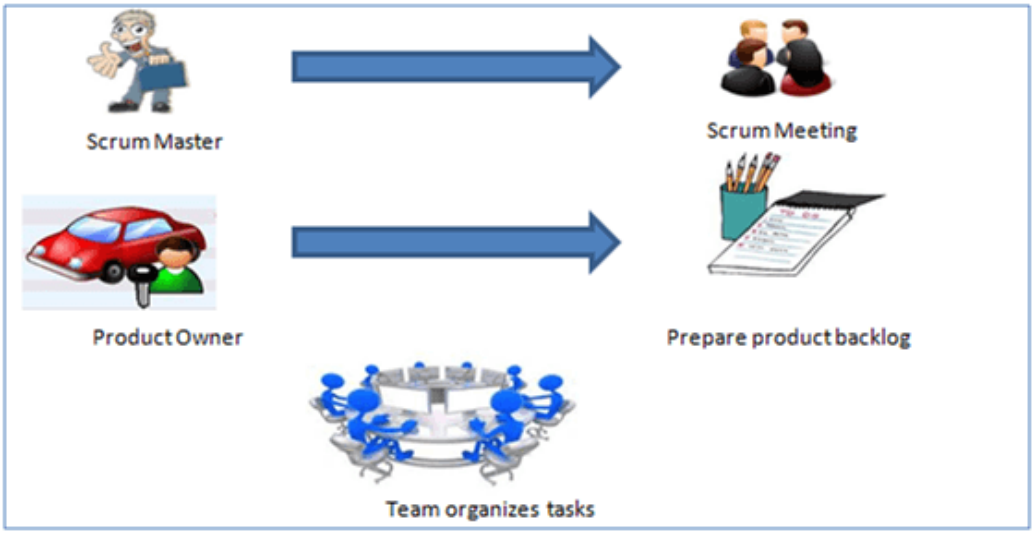
The **disadvantages of the Agile Model** are as follows −

- Not suitable for handling complex dependencies.
- More risk of sustainability, maintainability and extensibility.
- Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.
- Transfer of technology to new team members may be quite challenging due to lack of documentation.

Agile Processes: Scrum

SCRUM is an agile development method which concentrates specifically on how to manage tasks within a team-based development environment.

It consists of three roles, and their responsibilities are explained as follows:



- Scrum Master
    - Master is responsible for setting up the team, sprint meeting and removes obstacles to progress
- Product owner
    - The Product Owner creates product backlog, prioritizes the backlog and is responsible for the delivery of the functionality at each iteration
- Scrum Team
    - Team manages its own work and organizes the work to complete the sprint or cycle

Process flow of Scrum Methodologies:

- Each iteration of a scrum is known as Sprint [fundamental 30-day development cycle].
- Product backlog is a list where all details are entered to get the end-product
- During each Sprint, Product backlog are selected and turned into Sprint backlog
- Team works on the defined sprint backlog
- Team checks for the daily work
- At the end of the sprint, team delivers product functionality

<u>Project Management Concepts/Process:</u>

(Explain the below points in detail)

1. Determining the scope of the project
2. Project Planning
3. Implementation of project in timely manner
4. Review of project activities
5. Listing of lessons learnt during the project

Project Estimation and Scheduling

Project Size * Project Attributes = - Scheduling

- Effort

- Cost

Estimation involves answering the following question:

1. How much calendar days or time is needed to complete each activity?
2. How much effort is required to complete each activity?
3. What is the total cost of each activity?

Here are some points for cost estimation:

1. Models used to development
2. The programming languages used
3. The presence or absence of reusable artifacts
4. Tools used for development
5. The environment of the office work space
6. If there are multiple teams working in different locations
7. Travel and Training costs
8. Hardware and software costs including maintenance

Project Scheduling

1. Identify all the functions required to complete the project.
2. Break down large functions into small activities.
3. Determine the dependency among various activities.
4. Establish the most likely size for the time duration required to complete the activities.
5. Allocate resources to activities.
6. Plan the beginning and ending dates for different activities.
7. Determine the critical path. A critical way is the group of activities that decide the duration of the project.

Risk Management

These potential issues might harm cost, schedule or technical success of the project and the quality of our software device, or project team morale.

Types of Risk:

1. Generic
2. Product Specific
3. Project Risk
4. Product Risk
5. Business Risk

1. These are the risk's that can occur in all phases of software development. Ex- Change in requirement, loss of team members, loss of funding etc.,
2. These are the risk's associated with the type of product being developed. Ex- Availability of testing resources
3. These risk's affect project scheduling of resources
4. These risk's affect quality or performance of software
5. These risk's affect the viability of the software

Other risk categories

1. Known risks: Those risks that are known in advance but still the organization accepts the project.

2. Predictable risks: Those risks that are hypothesized from previous project experience.

3. Unpredictable risks: Those risks that can and do occur, but are extremely tough to identify in advance.

Maintenance and Re-Engineering

Software maintenance is a part of the Software Development Life Cycle. Its primary goal is to modify and update software application after delivery to correct errors and to improve performance. Software is a model of the real world. When the real-world changes, the software require alteration wherever possible.

Need for Maintenance

- Correct errors
- Change in user requirement with time
- Changing hardware/software requirements
- To improve system efficiency
- To optimize the code to run faster
- To modify the components
- To reduce any unwanted modules.

Types of Maintenance:

1. **Corrective Maintenance**

Corrective maintenance aims to correct any remaining errors regardless of where they may cause specifications, design, coding, testing, and documentation, etc.

2. **Adaptive Maintenance**

It contains modifying the software to match changes in the ever-changing environment.

3. **Perfective Maintenance**

This includes modifications and updates done in order to keep the software usable over a long period of time.

4. **Preventive Maintenance**

This includes modifications and updates to prevent future problems of the software.

----------*----------