

# **Chapter 9:**

## **Agile Methodology: Agile Model in Software Testing/ Scrum**

Prepared By: Subina Maharjan



# Agile Software Development

- also referred to simply as Agile
- is a type of development methodology that anticipates the need for flexibility
- focuses on the clean delivery of individual pieces or parts of the software and not on the entire application.
- provides the opportunity to make changes as needed and alert teams to any potential issues.



# The 12 principles of Agile

1. Satisfy customers through early and [continuous delivery](#) of valuable work.
2. Break big work down into smaller tasks that can be completed quickly.
3. Recognize that the best work emerges from self-organized teams.
4. Provide motivated individuals with the environment and support they need and trust them to get the job done.
5. Create processes that promote sustainable efforts.
6. Maintain a constant pace for completed work.

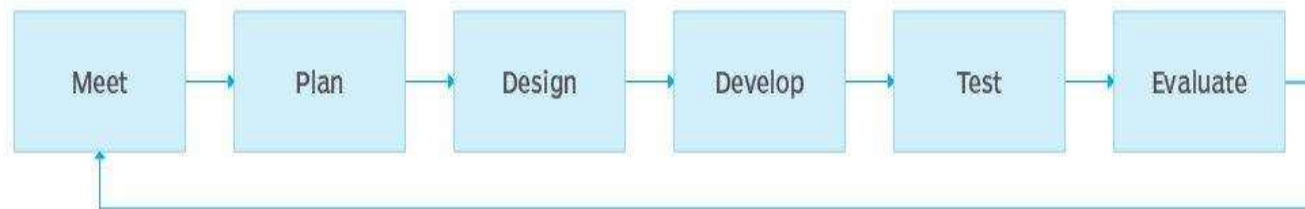


## The 12 principles of Agile cntd...

7. Welcome changing requirements, even late in a project.
8. Assemble the project team and business owners on a daily basis throughout the project.
9. Have the team reflect at regular intervals on how to become more effective, then tune and adjust behavior accordingly.
10. Measure progress by the amount of completed work.
11. Continually seek excellence.
12. Harness change for a competitive advantage.

# Agile Software Development Cycle

## Agile software development cycle





# Types of Agile methodologies

- [Scrum](#)
- [Lean software development](#)
- [Extreme programming](#)
- [Kanban](#)
- [Feature-driven development](#)

# Agile vs Waterfall

Agile	Waterfall
It separates the project development lifecycle into sprints.	Software development process is divided into distinct phases.
It follows an incremental approach	Waterfall methodology is a sequential design process.
Agile methodology is known for its flexibility.	Waterfall is a structured software development methodology so most times it can be quite rigid.
Agile can be considered as a collection of many different projects.	Software development will be completed as one single project.
Agile is quite a flexible method which allows changes to be made in the project development requirements even if the initial planning has been completed.	There is no scope of changing the requirements once the project development starts.

# Scrum vs Kanban


Scrum	Kanban
It recommends collection of <b>time measurements</b> made during sprints	Kanban <b>recommends graphs</b> to get an overview of team's progress over time.
Scrum <b>no longer</b> asks for a commitment from teams. Instead, it is about the sprint goals and forecasts.	Kanban relies on <b>time-boxing and forecasts</b> .
It stresses on planning, and so <b>estimation has a very important role</b> in Scrum	Kanban has <b>no mandatory requirements</b> for estimation.
Every <b>individual has their role</b> and responsibilities.	No <b>set roles so flexibility</b> in term of individual responsibilities.
The iterations/Sprints are fixed in duration. This duration varies from 2 weeks to 1 month.	Kanban is <b>not based on duration</b> . This thing is measured regarding Cycle times.
Teams are <b>required to commit</b> a specific amount of work.	<b>Commitment not necessary</b> it is optional for teams.





# QA Challenges with Agile Development

- Insignificant Information
- Constant Testing
- Repetitive Regression Cycles
- Haphazard Quality measurement
- Frequently changing requirements



# Boundary Value Analysis and Equivalence Partitioning Testing

Boundary testing is the process of testing between extreme ends or boundaries between partitions of the input values.

- So these extreme ends like Start- End, Lower- Upper, Maximum-Minimum, Just Inside-Just Outside values are called boundary values and the testing is called “boundary testing”.
- The basic idea in normal boundary value testing is to select input variable values at their:
  1. Minimum
  2. Just above the minimum
  3. A nominal value
  4. Just below the maximum
  5. Maximum



# Equivalence Partitioning Testing

**Equivalence Partitioning** or Equivalence Class Partitioning is type of black box testing technique which can be applied to all levels of software testing like unit, integration, system, etc. In this technique, input data units are divided into equivalent partitions that can be used to derive test cases which reduces time required for testing because of small number of test cases.

- It divides the input data of software into different equivalence data classes.
- You can apply this technique, where there is a range in the input field.

[More on Boundary value and Equivalence partitioning](#)



# Ethics of Software Testing

- **Honesty and Transparency** (Accurate Reporting, Full Disclosure)
- **Confidentiality** (Data Privacy, Non-Disclosure)
- **Integrity** (Avoiding Bias)
- **Respect for Intellectual Property** (Proper Use, Acknowledgement)
- **Professional Competence** (Continuous Learning, Adherence to Standards)
- **Responsibility to Users** (User Safety, Accessibility)
- **Responsibility to the Organization** (Avoiding conflicts of interest, Honoring Agreements)



*Any queries ?*

**THANK YOU !**