Agile

Able to move quickly and easily

SDLC

- Software Development Life Cycle

- It is a process used to design, develop, and maintain software systems.

Types of SDLC Models

1.Waterfall model

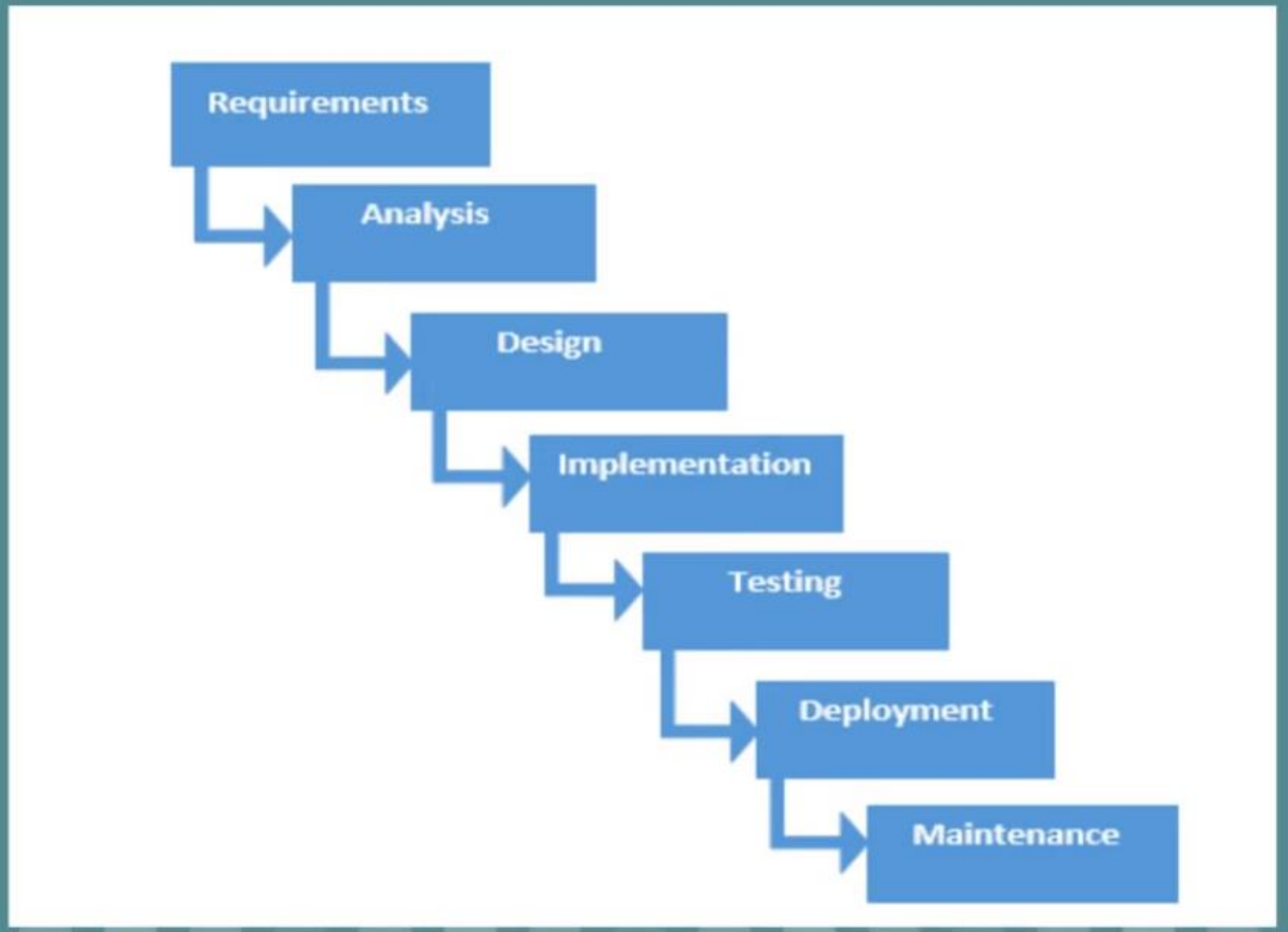
2.Spiral model

3.V-shaped model

4.Iterative model

5.Agile model

Waterfall


Requirements
Analysis
Design
Implementation
Testing
Deployment
Maintenance

Advantages

1. Clear project objectives

   The waterfall model is useful when the project objectives are well-defined and stable.

2. Easy to understand:

   The process is straightforward and easy to understand, making it easier for developers to identify requirements and plan accordingly.

3. Easier to manage:

   Since the phases are completed in a sequential manner, it's easier to manage and control the development process.

4. Documentation:

   The waterfall model places emphasis on documentation, which can be helpful for knowledge transfer and future maintenance.

Disadvantages

1. Lack of flexibility:
   The waterfall model is not well-suited for projects where requirements are likely to change, as it doesn't accommodate changes easily.
2. No customer involvement:
   Customers are not involved until the very end of the development process, which can result in an end product that doesn't meet their needs.
3. High risk:
   The waterfall model is high-risk, as there is a lot of uncertainty in the early stages of development, and changes become more difficult and expensive as the process continues.
4. Lengthy development cycle:
   Since each stage of the development process is completed in sequence, the development cycle can be lengthy, which can lead to delays and increased costs.
5. Limited feedback:
   The waterfall model doesn't allow for feedback from stakeholders until the end of the development process, which can lead to issues being discovered late in the process and result in higher costs to fix them.

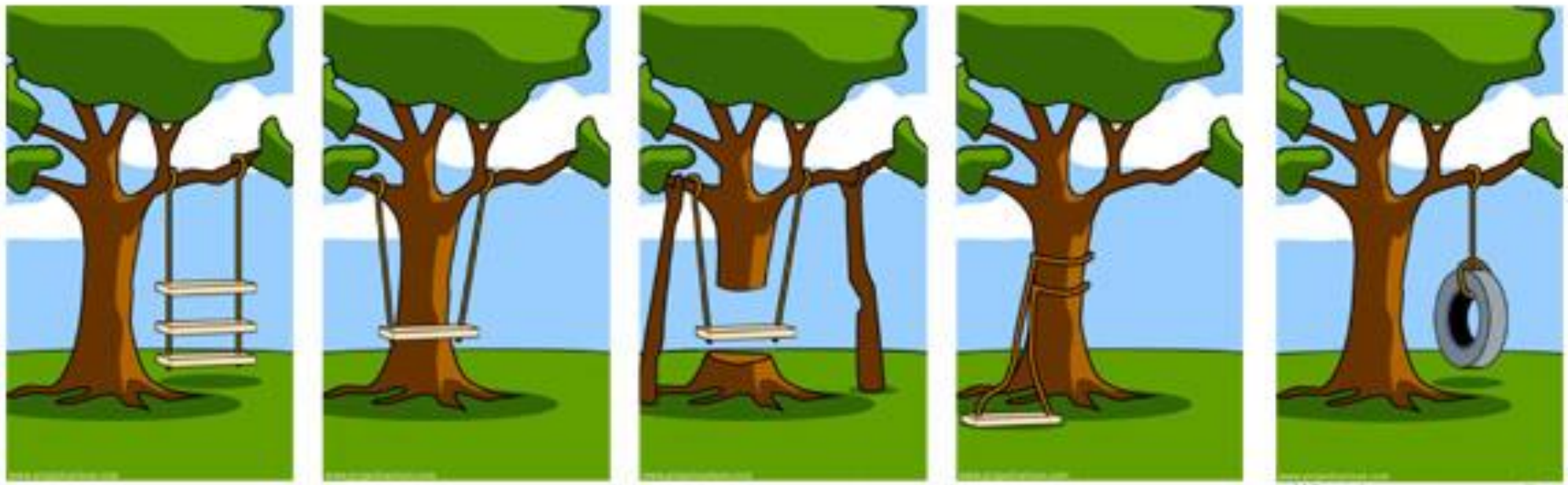
How the customer explainened it.

How the project leader understood it.

How the analyst designed it.

How the programmer wrote it.

What the customer
really wanted.


How the customer explainened it.
How the project leader understood it.
How the analyst designed it.
How the programmer wrote it.
What the customer really wanted.

Waterfall can either be a huge success or a huge failure

WHY AGILE?

ALL THE REASONS WE HAVE SEEN IN THE PREVIOUS SLIDE


WOMEN'S WORLD CUP
FRANCE 2019
NOR
AUS
ENG
CMR
FRA
BRA
ESP
USA
ITA
CHN
NED
JPN
GER
NGA
SWE
CAN
WINNER
Sports Illustrated

Examinations during our childhood followed agile
Mid terms, Quarterly, Half yearly and then only
Annual!

Using this Agile approach, the teachers can identify the strong and weak zones of students and pay special attention to weaker areas.

Imagine if schools were to conduct only annual exams!!!

Agile Manifesto

- Set of guiding values and principles for Agile software development.
- https://agilemanifesto.org/

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

12 principles behind the Agile Manifesto

1. Customer satisfaction through early and continuous software delivery.
2. Accommodate changing requirements throughout the development process.
3. Deliver working software frequently, with a preference for shorter timescales.
4. Collaboration between developers and business people throughout the project.
5. Support, trust, and motivate the people involved.
6. Enable face-to-face interactions.

12 principles behind the Agile Manifesto

7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity—the art of maximizing the amount of work not done—is essential.
11. Self-organizing teams encourage great architectures, requirements, and designs.
12. Regular reflections on how to become more effective, then tuning and adjusting behavior accordingly.

Agile Software Development-Definition

- Agile Software Development is a software development methodology that emphasizes iterative and incremental development, continuous customer feedback, and rapid delivery of working software.

- It promotes adaptive planning, evolutionary development, and self-organizing teams, and prioritizes responding to change over following a plan.

- The Agile approach values collaboration, simplicity, and flexibility, and encourages continuous improvement and learning throughout the software development process.


AGILE METHODOLOGY
SPRINT 1
SPRINT 2
SPRINT 3
Test
Develop
Plan
Design
Deploy

Advantage

1. Faster Time-to-Market

   Agile methodologies promote rapid delivery of working software, which allows teams to get software into the hands of users more quickly than traditional development methods.

2. Flexibility and Adaptability:

   Agile methods prioritize responding to change over following a plan, so teams can adjust their development approach as they learn more about user needs and market requirements.

3. Improved Collaboration and Communication:

   Agile methods emphasize cross-functional teams, face-to-face communication, and frequent feedback from customers, which can improve team morale, productivity, and alignment with business goals.

4. Higher Quality Software:

   Agile methods promote continuous testing and feedback, which helps identify and fix defects earlier in the development process, leading to higher quality software.

5. Customer Satisfaction:

   Agile methodologies prioritize customer collaboration and feedback, which can result in software that better meets customer needs and expectations, leading to higher levels of customer satisfaction.

Disadvantage

- Lack of Upfront Planning:
  - Agile methodologies emphasize flexibility and adaptation, which can sometimes mean that there is less emphasis on detailed upfront planning. This can be challenging for teams that prefer a more structured and predictable development process.
- Communication Overload:
  - Agile methods prioritize frequent communication and collaboration, which can be overwhelming for some team members who may struggle to keep up with constant feedback and requests for input.
- Dependencies on Individuals:
  - Agile methodologies rely heavily on self-organizing teams and cross-functional collaboration, which can be challenging when key team members are unavailable or leave the team.
- Limited Documentation:
  - Agile methods prioritize working software over comprehensive documentation, which can be a disadvantage for teams that require detailed documentation for compliance or regulatory purposes.
- Customer Availability:
  - Agile methodologies rely heavily on customer collaboration and feedback, which can be a disadvantage if customers are not available or engaged in the development process.

Agile –Myths and Reality

- Myth: Agile means no documentation.

- Reality: While Agile methods prioritize working software over comprehensive documentation, they do not eliminate the need for documentation entirely. Documentation is still important for understanding requirements, design, and architecture, but Agile methodologies focus on creating just enough documentation that is needed.

Agile –Myths and Reality

- Myth: Agile doesn't require planning.

- Reality: Agile methodologies emphasize adaptive planning, which means that planning is still an essential component of the development process. However, Agile planning is more flexible and iterative than planning in traditional methodologies.

Agile –Myths and Reality

- Myth: Agile means constant change.

- Reality: While Agile methods prioritize responding to change over following a plan, they do not necessarily mean that there is constant change throughout the development process. Agile methodologies emphasize incremental and iterative development, which can provide stability and predictability to the development process.

Agile –Myths and Reality

- Myth: Agile can only work on small projects.

- Reality: While Agile methodologies were originally designed for small, co-located teams, they have since been successfully applied to large, distributed teams working on complex projects.

Agile –Myths and Reality

- Myth: Agile means no project management.

- Reality: Agile methodologies emphasize self-organizing teams, but they still require project management. In fact, Agile methodologies promote frequent communication, collaboration, and feedback, all of which are essential components of effective project management.

Agile frameworks

1. Scrum
2. Kanban
3. Lean
4. Extreme Programming (XP)
5. Crystal

SCRUM

- Iterative and incremental development through short sprints.

- Scrum is based on the principles of transparency, inspection, and adaptation, and promotes collaboration and continuous improvement.

Scrum Core Practices and Artifacts

1.Sprint:

A sprint is a time-boxed iteration of development, usually lasting two to four weeks. The team works on a set of features or user stories during the sprint, and at the end of the sprint, they deliver a working increment of software.

2.Product Backlog:

The product backlog is a prioritized list of features or user stories that describe the functionality of the software being developed. The product owner is responsible for prioritizing the backlog and ensuring that it is up-to-date.

3.Sprint Planning:

At the beginning of each sprint, the team holds a planning meeting to decide what work will be completed during the sprint. The team reviews the product backlog and selects a set of items to work on, based on the priority and the team's capacity.

Scrum Core Practices and Artifacts

1.Daily Stand-up:

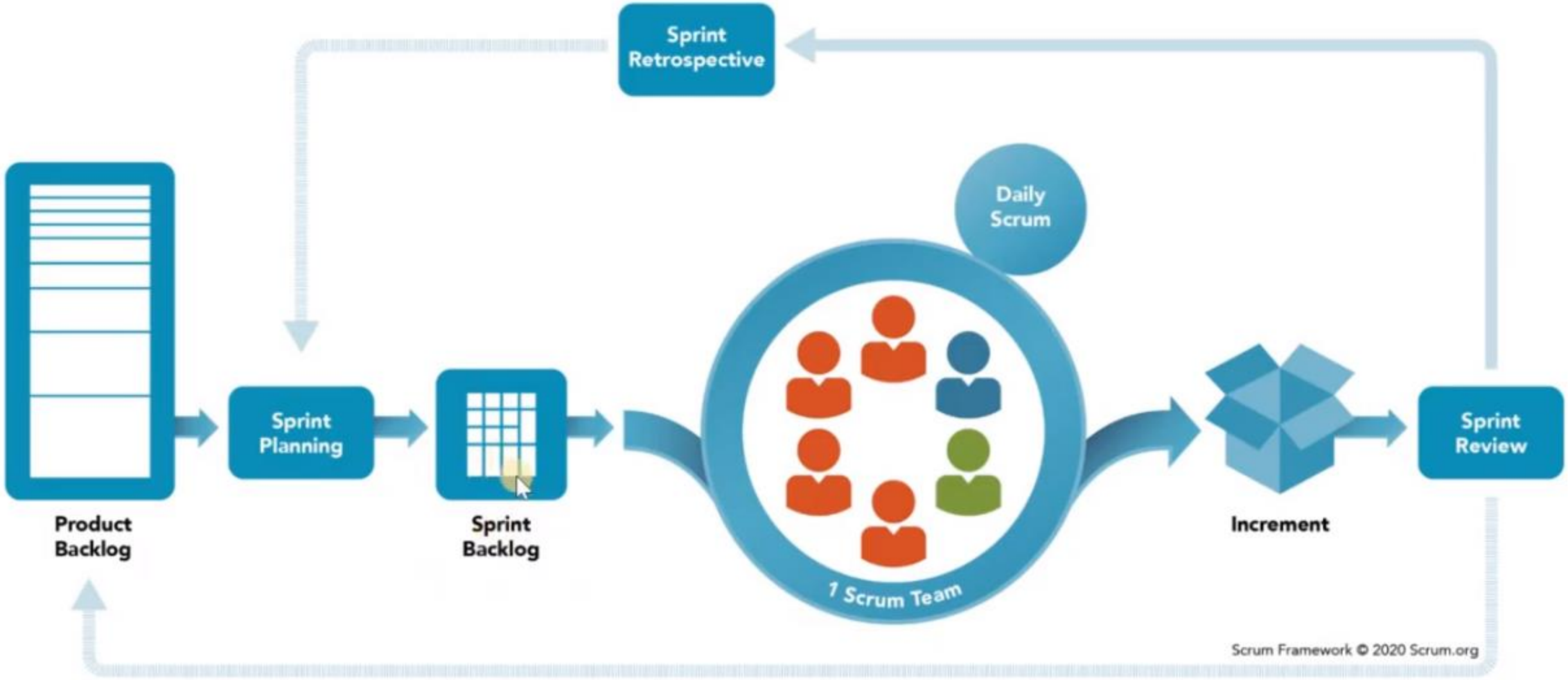
Every day during the sprint, the team meets for a brief stand-up meeting to discuss progress, identify any obstacles, and plan the work for the day.

2.Sprint Review:

At the end of each sprint, the team holds a review meeting to demonstrate the working software to stakeholders and gather feedback. The team discusses what was accomplished during the sprint, and what they plan to work on in the next sprint.

3.Sprint Retrospective:

Also at the end of each sprint, the team holds a retrospective meeting to reflect on what went well and what could be improved in the next sprint. The team identifies actions to take in the next sprint to improve the development process.



Source: scrum.org

User Story

- Brief, high-level description of a feature or functionality that is required by a user or customer
- Used to capture the customer's requirements in a simple and concise format
- Example
  - I want to search a products by category

SCRUM Roles

- Product owner
- Scrum Master
- Cross-functional development team

Product Owner

- Providing the vision of the product to the team
- Managing the product backlog in a clear and transparent way
- Prioritize the backlog items to achieve the best
- Explaining the product backlogs in a clear manner to the team to help the team to make better decisions.

Team (Product backlog -> shippable item)

- Self organised (what to commit and how to commit)
- cross functional(analysis, dev, testing, DB) and structured team)
- Team is accountable as a whole (No blame game on a single person)
- Usually 7 to 10 persons

Scrum Master (Agile, Scrum coach)

- Helps team to adopt Scrum
- Helps team to self organise
- Protects the team from outside interference
- Conducts scrum events
- Eliminates obstruction for the progress of the team

NOTE: Scrum Master is NOT and should NOT be the product owner/Manager

Scrum Ceremonies

- Sprint Planning
- Daily Stand-up
- Sprint Review
- Sprint Retrospective

Ceremony: Sprint Planning
Duration : ~ 2 hours

Participants : Team, Scrum Master, Product Owner
When: Occurs before the start of the sprint

Purpose:

- The dev team will discuss about the each item on the prioritized backlog that the PO has and the entire group collectively estimates the effort involved.
- Dev team will make a forecast on how much work can be done on that sprint considering several factors and risks
- The shortlisted list of backlog becomes Sprint Backlog

Ceremony: Daily Standup
Duration : ~ 15 minutes

Participants : Team, Scrum Master, Product Owner
When: occurs daily

Purpose:

- To quickly inform everyone on what's going on across the Team.
- It's not a detailed one. It should be short and to the point.
- The following questions are answered by the team
  - What did I complete yesterday?
  - What will I work on today?
  - Am I blocked by anything?

Ceremony: Sprint Review
Participants : Team, Scrum Master, Product Owner
Duration : ~ 30 min to 1 hour
When: At the end of each sprint

Purpose:

- The team will showcase the work they did.
- This event is to celebrate the completion of iteration and get feedback about the completed work.
- This event usually happens on a Friday and is more like a fun meeting rather than a formal one!

Ceremony: Retrospective **Participants** : Team, Scrum Master, Product Owner
Duration : ~ 1 hour When: at the end of the sprint

Purpose:

- Retrospective meetings help the team understand “what went well” and “ What didn’t” for the team.
- Rather than complaining, team will keep doing things which are under “what went well” category and will try to find creative solutions for things that didn’t go well.
- Continuously getting adapted and learning is the ESSENCE of Agile, and retrospective is the KEY place for that!

Scrum.org

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


Definition of Done

Definition of Done

- ‘Potentially Shippable’ gives a state of ‘confidence’ that what we developed in the Sprint is actually ‘done’.
- Indicates that the built product doesn’t have any ‘undone work’ and is ready to ship.
- If the Scrum teams have produced potentially shippable product, there must be a well-defined, agreed-upon “Definition of Done”.

Features of Definition of Done

- The code should be peer-reviewed
- Code is Checked-In
- Code is deployed to the test environment
- Regression testing should be passed by feature/code
- Documentation of code
- Help documentation is updated
- The Stakeholders approve the feature.


Planning poker

Planning poker

- Planning poker, also known as “scrum poker” and “pointing poker”
- Is a gamified technique that development teams use to guess the effort of project management tasks.
- These estimations are based on the entire group’s input and consensus, making them more engaging and accurate than other methods

How does planning poker work

Step 1: Hand out the cards to participants

- To start a poker planning session, the product owner or customer reads an agile user story or describes a feature to the estimators.
- For example:
- “Customer logs in to the reservation system”
- “Customer enters search criteria for a hotel reservation”
- Team members of the group make estimates by playing numbered cards face-down to the table without revealing their estimate (Fibonacci values: 1,2,3,5,8,13,20,40)
- Cards are simultaneously displayed
- The estimates are then discussed and high and low estimates are explained
- Repeat as needed until estimates converge

Card(s)	Interpretation
0	Task is already completed.
1/2	The task is tiny.
1, 2, 3	These are used for small tasks.
5, 8, 13	These are used for medium sized tasks.
20, 40	These are used for large tasks.
100	These are used for very large tasks.
<infinity>	The task is huge.
?	No idea how long it takes to complete this task.
<cup of coffee>	I am hungry ☺

Organization of the game

- Full team will involve in Planning Poker game.
- Will require 2–4 hours
- Also require a large table where all members will sit.
- Each team member will have a set of cards
- In each card set there will have 10 cards with numbering 0, 1, 2, 3, 5, 8, 13, 21, 40 and 100 i.e. 1st card’s no will 0, 2nd card’s no will 1, 3rd card’s no will 2, 4th card’s no will 3, 5th will 4, 6th will 8, 7th will 13, 8th will 21, 9th will 40 and 10th will 100


13
8
20
?
40
13

Step 2: Read the story out loud

- The moderator (either the product owner or product manager) narrates the story to the group.
- If participants have any questions, the moderator answers them.

Step 3: Discuss the story

- Once the group finishes listening to the story, everyone shares their views on it.
- Some of these discussion points will likely include:

    1. How should we handle the work?
    2. How many people are expected to get involved?
    3. What skills will be needed to work on the story?
    4. How should we tackle any roadblocks that delay progress?

The group will also try to learn more about the story and ask questions to understand it better.


PRODUCT BACKLOG

Step-04: Playing Cards

- After discussion team members will ask to point for that story i.e. Product owner will ask them Now Select your card for assign Story Point to this story
- Each team member will select their card and put it on the table with face down, so that others can’t see his/her point.
- Once everyone selected their card the product owner will say Now Show
- As soon as the Product owner say this everyone will turn over their card so that everyone present their can see the estimated number


5
5
5
8
Susan
Vadim
Ann
Chris

Step-05: Achieving Consensus

- Most of the time one thing will identify which is a huge mismatch between given numbers.
- In that case members with highest and lowest numbers will ask why their chose estimates
- They will give some justification and discuss with the team regarding this issue.
- This will also help the team if they have any misunderstanding or vogue idea about the story and adjusting their estimation
- After justification the process i.e. Step-04 will continue again
- This process will continue until all members agrees to a common estimate

- Once all team members agrees to a common estimate then Product owner will record that point and go for next story and will continue from step-02 to step-05
- If after 5–6 rounds of playing the game estimation fail meet all members agreement then put it aside for revisit later

Putting a story aside allows for:

- The Product Owner do more research and provide more clarity.
- The developers to investigate the technical options or details (reducing the technical uncertainty).


Story Points

Story point

- Story point is nothing but a measuring unit.
- In simple terms it’s a number that tells the team how complex and big the story is.
- Story size could be related to complexity, uncertainties etc.

- Say you are going on a vacation from city A to city B which is 295 miles away.
- I ask how much time would it take to reach B. Can you give me the exact estimate in hours considering traffic, tolls, weather conditions and of course the vehicle condition.
- I guess no, because If you say 10 hours then that is inaccurate and a vague estimate.
- Now what if I tell you
- On this route (for past 6 months) average speed for all medium load vehicles is 50 miles/hours
- Gets easy ?
- Now you can tell, it would roughly take around 6 hours. (estimate gets better by 4 hours)
- Similarly in software development, product manager has a feature. She translates the feature into multiple small stories and can easily make out the required time to deliver as per the velocity of the team.

- A story size can be easily evaluated after some practice. During initial stages, it is not easy to estimate story points so we defined some baseline stories for each story pointer.
- We used these baselines stories to estimate story points. Depending upon the tasks involved and complexity of the feature, one can easily compare it with other stories and identify the story points.

Extreme Programming (XP)

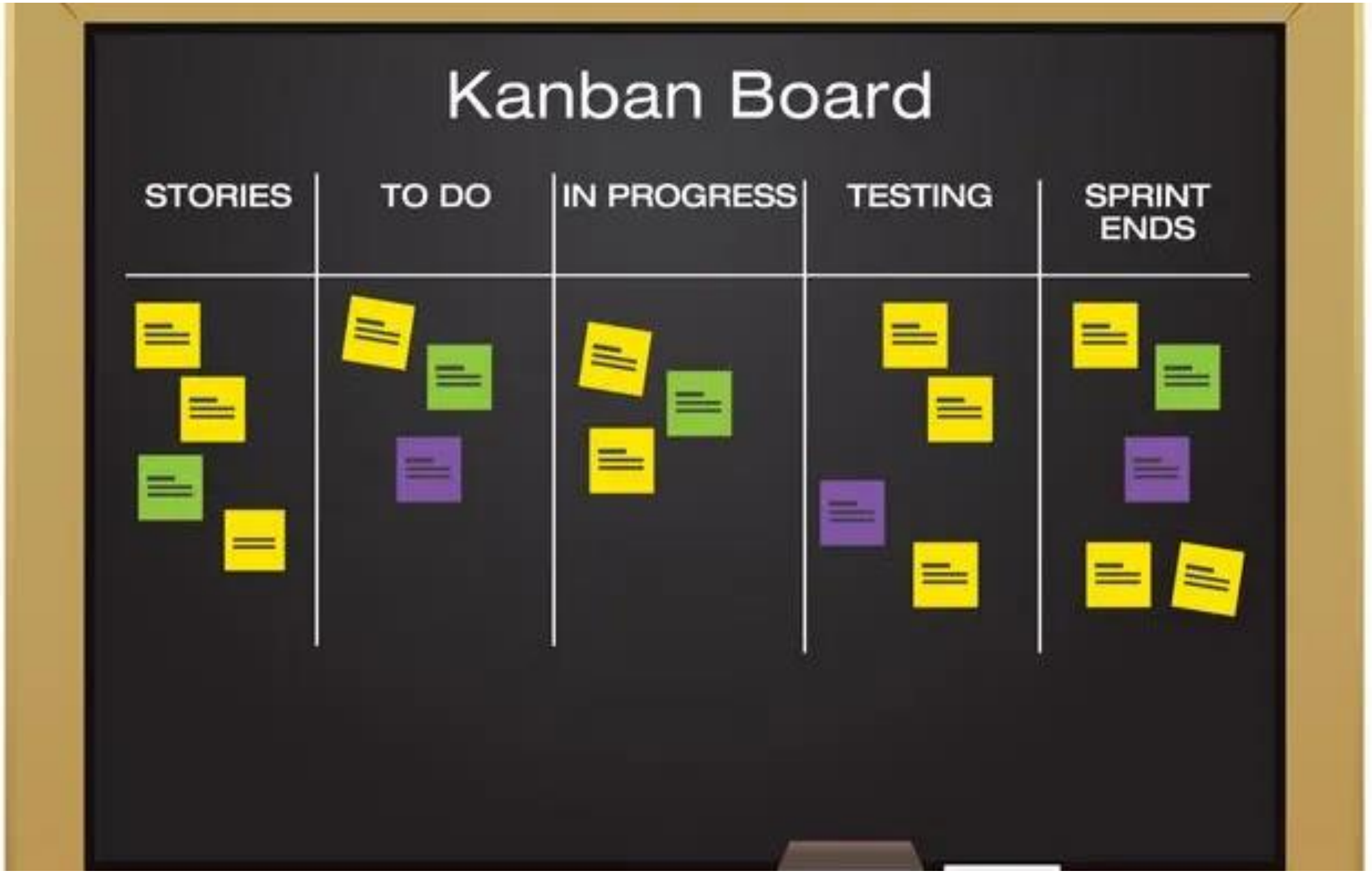
Extreme Programming (XP)

- Is an agile software development framework that aims to produce higher quality software and higher quality of life for the development team.

Kanban

Kanban

- Kanban is all about visualizing our work, limiting work in progress, and maximizing efficiency (or flow).
- Kanban teams focus on reducing the time a project takes (or user story) from start to finish.
- They do this by using a kanban board and continuously improving their flow of work.


Kanban Board
STORIES
TO DO
IN PROGRESS
TESTING
SPRINT ENDS