simplilearn

your pace, your place

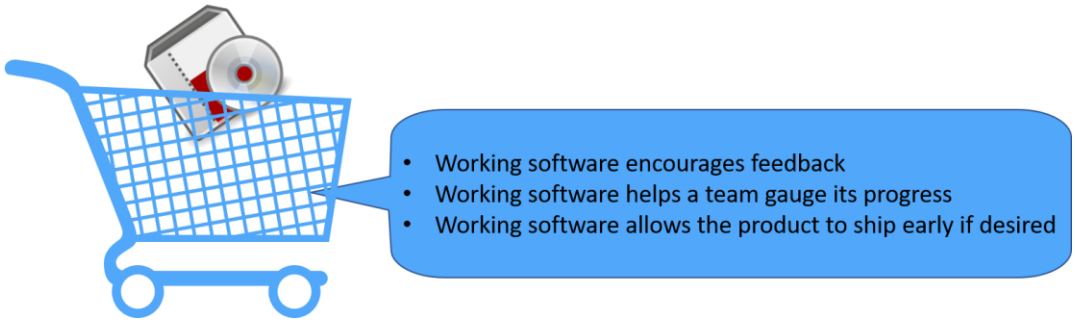
Successful Scrum Adoption

Guidance for Teams – Sprints

Deliver working software each Sprint

- At the end of each Sprint, working software must be produced by the Scrum team.
- Agile manifesto clearly says "***Working Software over Comprehensive Documentation.***" This working software is both complete and potentially shippable.

Key reasons for emphasizing on Working Software



Working Software encourages feedback

- By showing a working but partial product to the users, a Scrum team can collect better feedback.
- By seeing the product and working with it, end users will be able to engage better in providing the feedback.

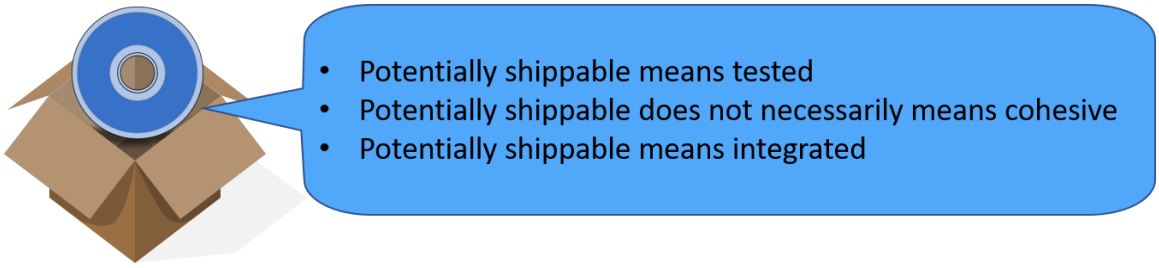
Working Software helps a team gauge its progress

- In traditional projects, sponsors and stakeholders may think the project is progressing normally and feel their deliverable will be on track. But it is extremely difficult to know how much effort is needed to produce the deliverable.
- By delivering working software in each Sprint, Scrum teams provide visible and verifiable progress to sponsors and stakeholders.

Working Software allows the product to ship early if desired

- Shipping working software early is very valuable.
- Though a few features might be missing, there is an option to test the market and provide some early feedback so that future releases may adapt to market conditions better

Guidelines for Potentially shippable product increment



Potentially shippable means tested

- The potential shippable product increment must be tested and bug free. No bugs should be introduced in already existing features.
- In a large project with a lot of integrations, teams may be able to do integration testing, regression testing, etc. after a few Sprints, not in each Sprint. Such conditions must be clearly articulated.

Potentially shippable does not necessarily mean cohesive

- In some situations, it would not be practical to ship product increments at the end of each Sprint.
- For example, there could be a set of features that are to be delivered over multiple Sprints. Only when all these set of features are completed, the product increment can be deployed.
- Even in such situations, teams should strive to produce potentially shippable product increment that can be tested independently.

Potentially shippable means integrated

- On projects where multiple teams are involved, Definition of Done should clearly include integration of development work of these multiple teams.
- To the extent possible, integration should be done continuously throughout the Sprint.

Deliver something valuable each Sprint

- Each Sprint should deliver something that is valuable to the actual users of the system, that is, end users.
- Business or customer must define what is valuable to them, and Scrum teams should deliver this.
- Each Sprint provides an opportunity to get valuable customer feedback. Therefore, Scrum teams should focus on delivering something valuable in each Sprint and get customer feedback

- There may be situations where some work might not be visible to end users. For example, in a banking system, payments are processed in nightly batch files. No end user will be able to see these batch files on their user interface. It may appear less valuable or of no value at all to users. But without this feature, the payment reconciliation might not happen, leading to multiple issues and escalations. In such a situation, the Scrum team can demonstrate this batch file processing to end users during Sprint Review to assure them that team is delivering something that is valuable but may not be shown on user interface.

Prepare in this Sprint for the next

- When working on a Sprint, teams may think about next Sprint only toward end of the Sprint or during next Sprint Planning meeting. This is a bad approach.
- Teams should spend about 10% of their time in current Sprint preparing for the upcoming Sprint.

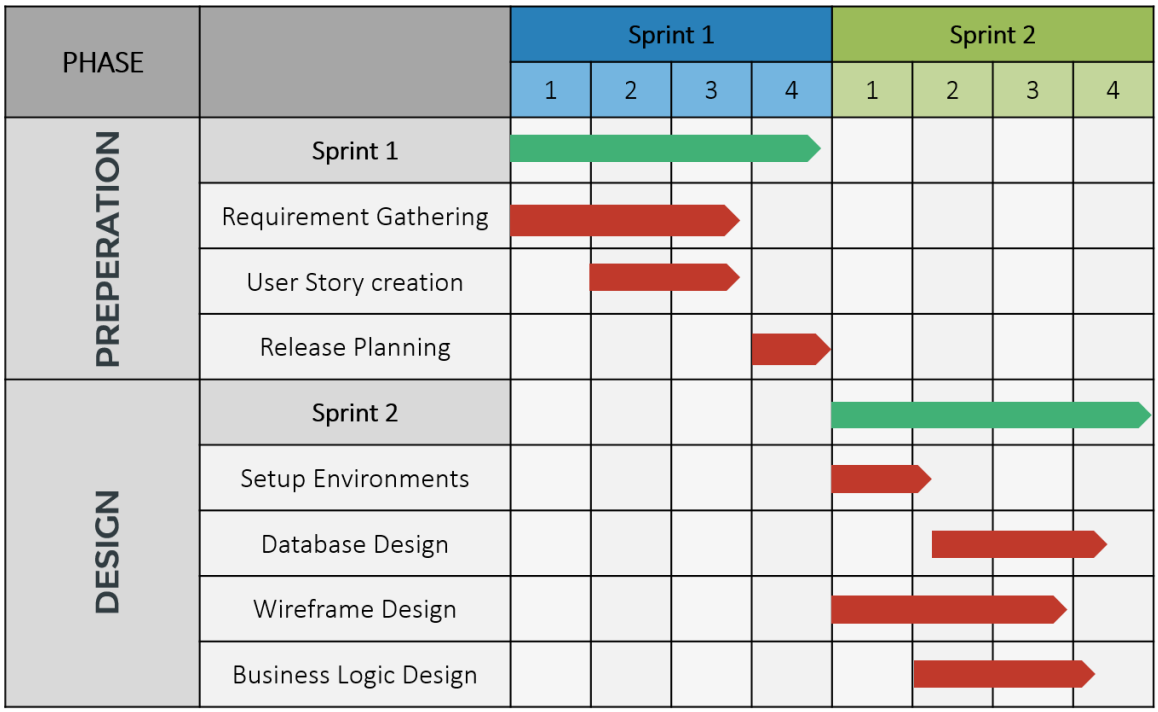
Teams should pick only those items that can be completed in a Sprint

- During Sprint Planning, teams should pick the User Stories that can be completed within the Sprint.
- If they find that a User Story is large and can't be completed in a Sprint, that large User Story must be broken down into smaller User Stories.
- Product Owner should collaborate with the team during the Sprint to provide required details about the User Stories so team can complete them during the Sprint.

Work together throughout the Sprint

- Deep collaboration should exist in a Scrum team.
- Rather than working in groups, the Scrum team should have cross-functional teams working together.
- Cross functional teams work on products in parallel.

Avoid activity-specific Sprints



- In this example, we can see that teams are doing 4-week Sprints.
- The First Sprint is dedicated to Preparation phase of the project and the second Sprint is specific to design.
- These kinds of activity-specific Sprints must be avoided.

Disadvantages of activity specific Sprints

- Too many hand-offs and a lack of whole team responsibility
- There is increased schedule risk
- It takes longer to go from idea to a running, tested feature
- It doesn't really solve the problem of overlapping work

Keep timeboxes Regular and Strict

- Keeping the Sprint length fixed has these advantages:
    - Teams benefit from regular cadence
    - Sprint Planning becomes easier
    - Release Planning becomes easier

Never extend the Sprint

- At times, teams will face situations where they will not able to complete committed User Stories in the Sprint.
- Most teams would be tempted to extend the Sprint to complete these incomplete items.
- No matter what happens, the Sprint must finish on time.

- If the teams extend the length as they want, we lose discipline and value for our timeboxes.

Don’t change Sprint Goal

- Every Sprint Planning meeting begins with the team setting a goal that it intends to achieve in the Sprint.
- Due to any reason, if the team is not progressing as expected, it may try to change the Sprint goal.
- Sprint Goal should never be changed during the Sprint. If changed, teams lose focus.