



Module 5



Managed Right – Project Planning and Monitoring

Module Overview


- ▶ **Agile planning techniques**
 - ▶ Story points
 - ▶ Velocity
 - ▶ Release plan
 - ▶ Iteration plan
- ▶ **Monitoring project plan**
 - ▶ Daily Scrum
 - ▶ Release burndown chart
 - ▶ Iteration burndown chart



Learning Outcomes

- ▶ Recap Agile planning basics
- ▶ Monitor and assess a project plan
- ▶ Adjust a project plan





Module 5 – Topic 5.1



Agile Planning for Software Products

Topic Outline

- ▶ Recall the key concepts related to Agile planning
 - ▶ Story points
 - ▶ Velocity
 - ▶ Release plan
 - ▶ Iteration plan





Story Points

Estimates, Targets and Commitments

- ▶ A task estimate is an approximation of how much effort a task will take
 - ▶ Made by the developers
 - ▶ **Non negotiable** since they are only figures of how much effort work will take, not commitments to when the work will be done
- ▶ A target is a deadline to be met
 - ▶ Usually set externally to the development team
 - ▶ **Non negotiable** since other parties may depend on the dates
- ▶ Commitments outline what work will be completed at what time
 - ▶ Based on target deadlines and realistic estimates
 - ▶ They are **negotiated** between the client and the development team



Issues of Time-based Estimates

- ▶ Time-based estimates
 - ▶ Base task estimates on the amount of time it has taken to implement similar functionality in the past
- ▶ Managers end up negotiating estimates with the developers
 - ▶ Developer – a task that took four hours in the past is now being estimated to be completed in ten hours
 - ▶ Manager – a ten-hour estimate is whittled down to a six hour estimate

Mistake estimates as commitments



Story Points

- ▶ Eliminate time as the unit of measurement for estimating work
- ▶ Story points estimate required work to be done
- ▶ Story points are unit-less and relative
 - ▶ How long some work will take in relation to other pieces of work

Estimates aren't supposed to be commitments, and so taking the time aspect out of them, helps prevent them from becoming commitments



How it Works?

- ▶ Choose a user story within the product backlog which is relatively easy to estimate
- ▶ Assign that user story an “arbitrary” integer value, i.e., story points
- ▶ Assign story points to the rest of your user stories based on it
 - ▶ About the same size, the same number of points
 - ▶ About twice as large, twice the number of points



Relativity of Story Points

- ▶ Avoid assigning very precise numbers of points
 - ▶ Set the first user story as 100 story points, the second user story as 42 points, etc.
- ▶ A good way to support relativity is to assign story points based on a Fibonacci Sequence
 - ▶ A Fibonacci sequence looks like 1, 1, 2, 3, 5, 8, 13 ...
 - ▶ If your estimate falls between two numbers, round up to the nearest available number

If your first task estimate is assigned a three, and your next task seems to be about twice as large, what estimate will be for the next task?



Quick Question

What about story points makes them well suited for estimation, without becoming commitments? You may choose more than one answer.

- A. Story Points are unitless
- B. Story Point values may change from project-to-project
- C. Story Points are harder to understand
- D. Story Points do not map directly to one-time value



Quick Question

- Which of the following would be the most appropriate value for a story point?
- A. 200
 - B. 3
 - C. 18
 - D. 50





Velocity

Velocity

- ▶ In physics
 - ▶ Distance travelled / time
- ▶ For software projects
 - ▶ Story points completed for the user stories done / the length of a sprint



Calculating Velocity

- ▶ You could measure productivity in different units, user stories or story points completed, but do not mix them
- ▶ Since user stories can vary in size, the total number of their story points would better represent the work done for them
- ▶ Velocity = the number of **story points completed** for the **user stories done** within the duration of a **sprint**



Velocity Driven Development

- ▶ Use past actual velocities to estimate future velocity
 - ▶ The only way to know how much work a development team can get done in a sprint is by seeing what they've done before
- ▶ Require a certain amount of stability in past velocity
 - ▶ just started your project and don't have any velocity data
 - ▶ in the middle of your project with at least some amount of past data



Quick Question

You are calculating your team's velocity achieved for a specific sprint. Which of the following would count towards your team's velocity?

- A. the number of user stories completed in that sprint
- B. the story points of the user stories completed in that sprint
- C. a feature that has half of its tasks complete, so you can count half of its story points
- D. the story points of a feature that is programmed and documented, but has not yet been tested



Quick Question

You and your team are preparing for the first sprint of development. You want to calculate an estimated velocity, but your team has never worked together. Also, they don't have experience with this type of product. You have planned to complete four user stories this sprint. User story A has been assigned 3 story points. User stories B and C have both been assigned 5 story points each. And user story D has been assigned 1 story point. You broke down the user stories into tasks and created task time estimates. You have estimated that these stories will take 110 hours to complete in total. Your development team has only 95 available hours this sprint. You decide to remove user story A from the sprint because it was estimated to take 15 hours.

What would be your team's estimated velocity for Sprint 1?

- A. 3 user stories
- B. 14 story points
- C. 11 story points
- D. 110 hours
- E. 95 hours.





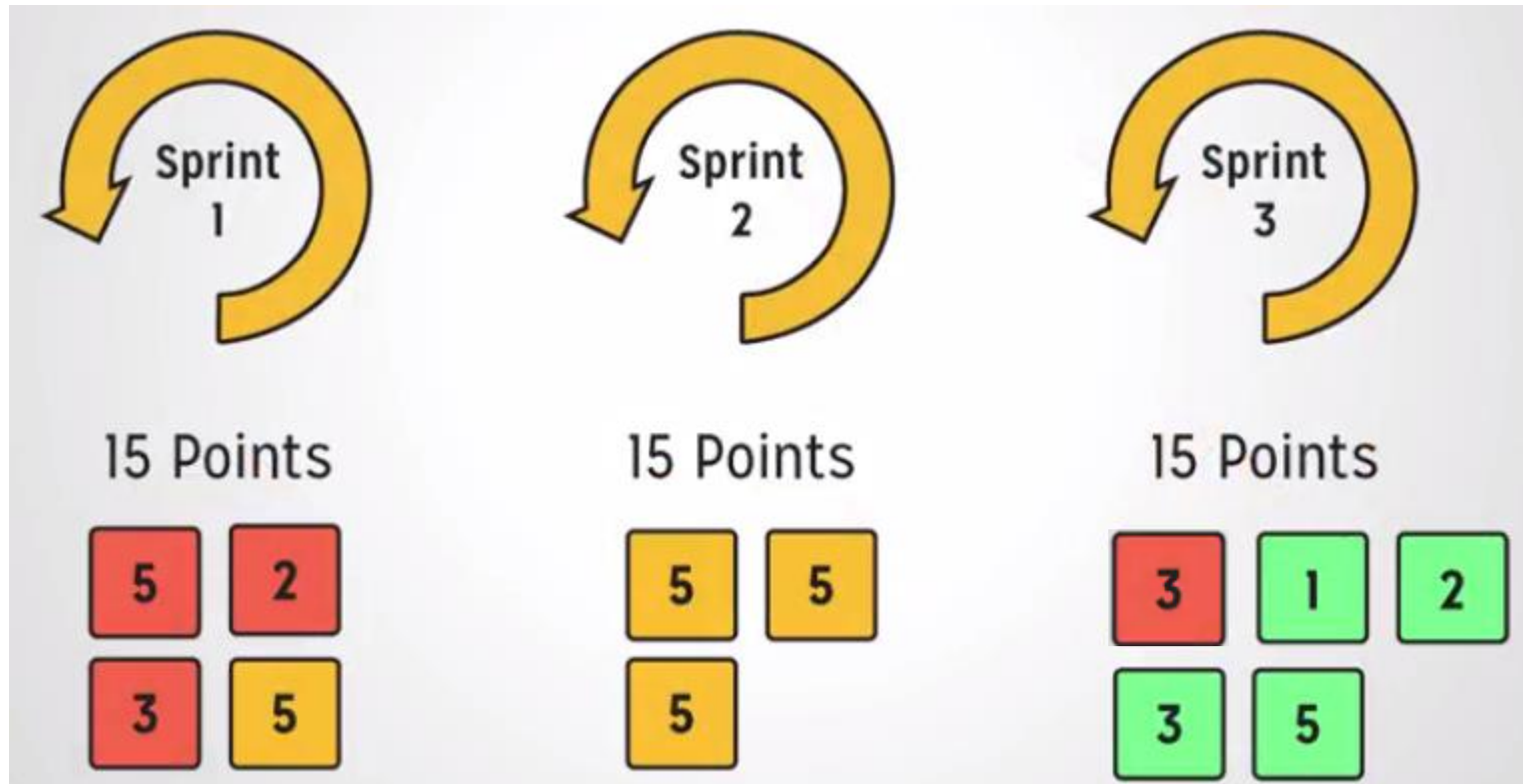
Release Plan

Release Planning

- ▶ Release planning is coarse grained, dealing with user stories as the pieces of work
 - ▶ Which user story should be released and reviewed by the product owner or client, in which sprint?
- ▶ User stories are scheduled into the upcoming sprints based on their priority



Release Plan – An Example



Knowing the amount of work your team can complete in a sprint, and the amount of work estimated for each user story, you can then fill in sprints you have based on user story estimates and velocity.



Quick Question

Alexis is a software product manager, working with her developing team to plan her project. They're creating a release plan and are choosing user stories to include in their first sprint.

How do Alexis and her team decide which user stories to complete first?

- A. They are prioritized by the development team
 - B. They are assigned at random to avoid bias
 - C. They are prioritized by the client
 - D. They are prioritized by the users
-





Iteration Plan

Recall Scrum Events

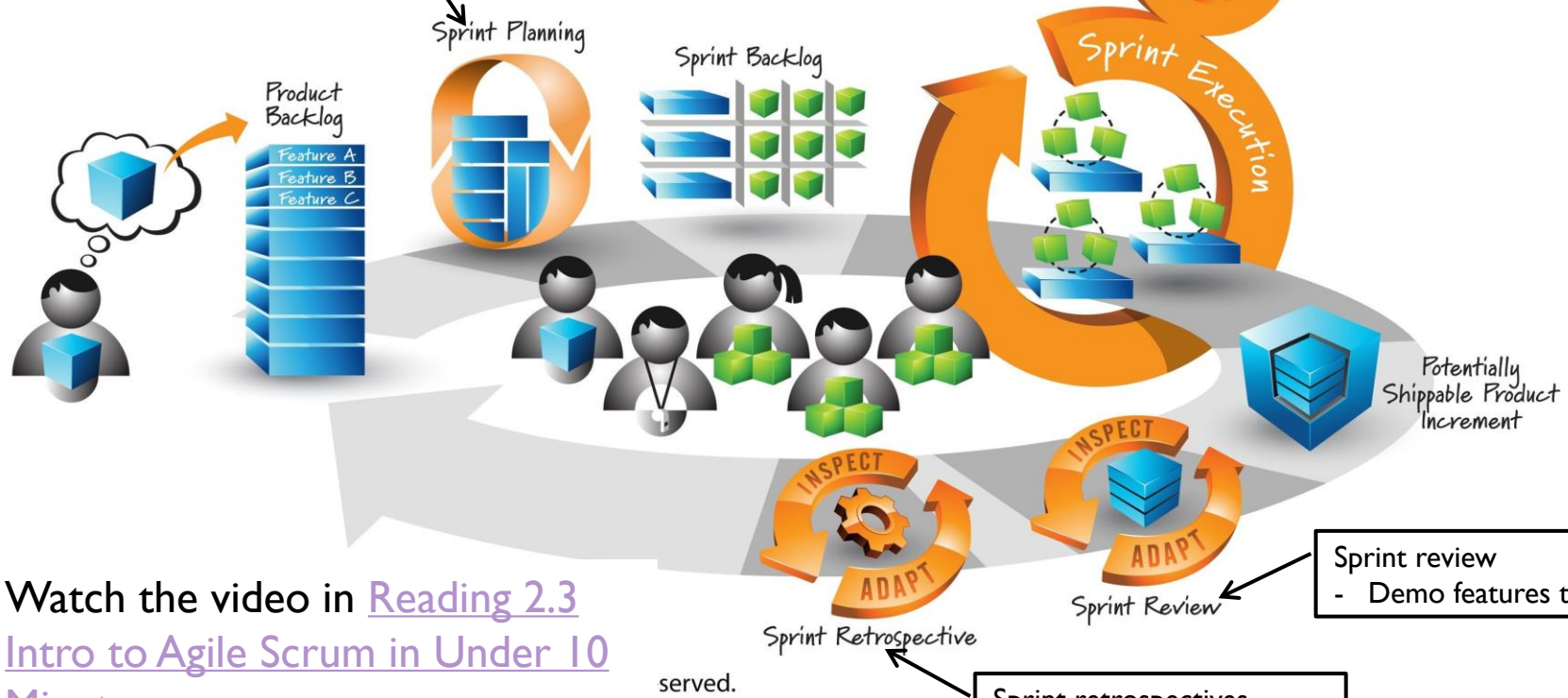
Spring Planning

- Review product backlog
- Estimate Spring backlog
- Commit for Spring
- Sprint goal

Daily scrum

- Done since last meeting
- Plan for today
- Obstacles?

Scrum Sprint Cycle 1-4 weeks



Watch the video in [Reading 2.3](#)
[Intro to Agile Scrum in Under 10](#)
[Minutes](#)

- Inspect and adapt

served.

Sprint Planning Meeting

- ▶ Sprint planning meeting should be time boxed
 - ▶ e.g., four hours
- ▶ What happens in a sprint planning meeting?
 - ▶ Establish a sprint goal
 - ▶ Report the project velocity from the last sprint
 - ▶ Select the user stories for the sprint
 - ▶ Break down the chosen user stories into developer tasks
 - ▶ Create an estimate for each task
 - ▶ Revisit the chosen user stories
 - ▶ Sign up for tasks



Establish a Sprint Goal

- ▶ **Sprint goal**
 - ▶ The general focus for the sprint being planned



Report Project Velocity from the Last Sprint

- ▶ All development commitments for this sprint will be based on this velocity
- ▶ What if your last sprint was an abnormal sprint that is not representative of the team's actual development rate?
 - ▶ Use the lowest velocity from your previous sprints



Quick Question

You and your development team have reached the fourth sprint of the project. You are conducting the sprint planning meeting for this sprint. In the first sprint your development team was able to complete ten story points of work from the backlog. In your second iteration your team completed 12 story points. In the third sprint your team experienced a hardware malfunction that resulted in some lost development days. This malfunction has since been fixed. However, your team was only able to complete five story points of work that was scheduled for that sprint. You look at your release plan and you'll have 13 story points of user stories scheduled.

How many story points of user stories do you think you should base your commitment on?

- A. 10 story points
- B. 12 story points
- C. 5 story points
- D. 13 story points



Selecting User Stories

- ▶ **Determine all the potential user stories**
 - ▶ All the user stories that have been planned to be completed in this sprint, based on the release plan
 - ▶ Any user stories that were not completed but supposed to be from the previous iteration
- ▶ From the potential user stories, the product owner selects the stories they would like completed
- ▶ The product owner can keep choosing user stories as long as the sum of their story points is less than or equal to the estimated velocity for this sprint



Break Down User Stories into Developer Tasks

- ▶ This is done by the development team without input from the product owner
- ▶ Developer tasks should be small enough that they can be completed in a day or two

As a music listener, I want to be able to pause my music, so that I can keep my place in my song when I need to listen to something else



- Add a pause button to the graphical interface
- Create function that saves user's place in the current song
- Create function that stops playing when pause button is pressed
- Create function which restores saved place in song



Quick Question

You are the software product manager for a team that is developing a music streaming application. Your development team is determining developer tasks based on the user stories for this iteration. You come across the user story: “As a listener I want to save songs to a playlist so that I can easily find these songs to be played later.”

Which of the following would be developer tasks for this user story?

- A. Write unit tests to ensure saved songs appear on a playlist
- B. Program functionality to save a song to a playlist
- C. Generate a wireframe for the application
- D. Add instructions to user manual for saving a song to a playlist



Create an Estimate for Each Task

- ▶ Because a Scrum team is cross-functional, each task estimate is really an agreed average across the team
- ▶ If a task requires specialized skills and must be assigned to a specific person
 - ▶ the estimate can be based on that person. Still the team must agree to it.
- ▶ The task estimate should be in hours, half days or days
 - ▶ Small tasks under an hour should be grouped together
 - ▶ Large task that are estimated to take more than 3 days should be broken into smaller tasks



Revisit the Chosen User Stories

- ▶ If all the user stories cannot be achieved
 - ▶ the developers need to adjust their commitment
 - ▶ the product owner needs to remove user stories
- ▶ If the user stories can be achieved with significant time left
 - ▶ the product owner can choose additional user stories

Given the total available time of the developers and the task estimates, can the developer tasks for the chosen user stories be fitted and realistically done within the sprint?



Sign Up for Tasks

- ▶ The development team self assigns tasks
 - ▶ Developers should self-assign and choose tasks based on their available time

