Module 5 – Topic 5.2

Managing Project Plan

Topic Outline

- Daily scrum
- Velocity and burndowns for both releases and iterations

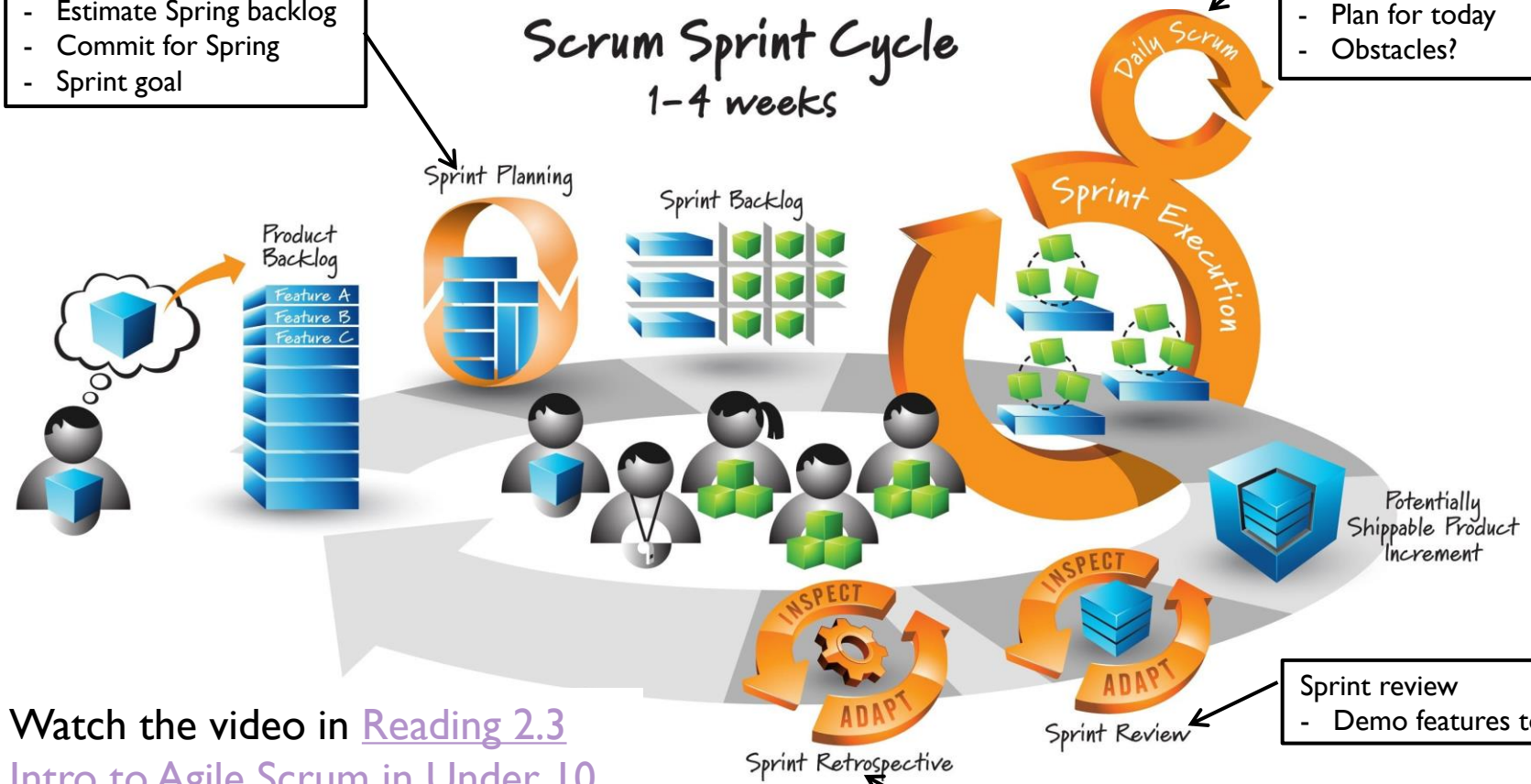
Daily Scrum

Occurs on each work day with the development team

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?



Sprint review
- Demo features to all

Sprint retrospectives
- Inspect and adapt

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

Daily Scrum Rules

- This meeting is strictly time-boxed at 15 minutes
- This meeting should happen at the same place at the same time each day
- Participants stand in a circle and answer three questions
  - This is why it is also called daily standup
- Not intended to be a status meeting for product managers or product owners to determine progress
  - For status updates, use progress tracking practices such as iteration burndowns

Answer Three Questions in Daily Scrum

- What did you accomplish yesterday?
  - e.g., yesterday I had a productive day and completed all the unit tests for the user profile page
- What will you do today?
  - e.g., today I am going to start writing the source code for that feature
- Are there any impediments in your way?
  - e.g., I am not that familiar with this programming language. Is there anyone who would like to pair program this feature with me today?

Who Speak in Daily Scrum?

- Only the "pigs", that is the people who are substantially committed to the project, should speak
  - "Pigs" - the developers and the product owner if they have an update for the team
  - "Chicken" - developers from other teams, business executives who want a status update, or the product owner if they have no major updates

Scrum Master Role

- Keep the meeting in the 15 minute time box
  - Any off topic conversations be added to a list of discussion topics for after the meeting
- Records any impediments that are not directly solved in the meeting
  - See that these are fixed or assign someone to do so
- Signal the end of the meeting

Quick Question

You are at the daily scrum it is your turn to speak next. Which of the following should you include in your update?

A. Yesterday, you completed the design for the database structure
B. Yesterday, you watched a really cool movie after work
C. Today, you are going to start designing the user interface
D. The supply room is out of paper, so you can not draw up mockups for the user interface
E. I have a suggested feature improvement for the product!

Quick Question

You are at the daily scrum meeting, one of the developers, Danny, said at yesterday's meeting that he was going to complete the source code for the login feature. At today's meeting he said that he worked on it yesterday and intends to complete it today. He says he has no impediments.

As the Scrum Master, what should you do?

A. Get mad at him! He committed to completing the feature yesterday
B. Politely ask Danny why he couldn't complete the feature, and if there is anything the team can do to help him
C. Nothing, he said he would get it done today
D. Assign someone else to complete the feature

Quick Question

You are the scrum master at the daily scrum meeting. This meeting has been going very smoothly. The last developer is giving their update, and the meeting has only been ten minutes. Her update sparks an off-topic conversation among the developers.

What should you do?

A. Let the conversation continue. There is still five minutes left in the meeting and no more updates
B. Let the conversation continue. This was something we needed to talk about anyways. It does't matter if we go over the 15 minute time box
C. Cut the conversation short, end the meeting, and have everyone go back to work
D. Cut the conversation short, add it to the list of discussion items, and suggest that anyone who wants to continue this discussion can talk about it after the meeting

Release Burndown Chart

What a Release Burndown Chart Shows

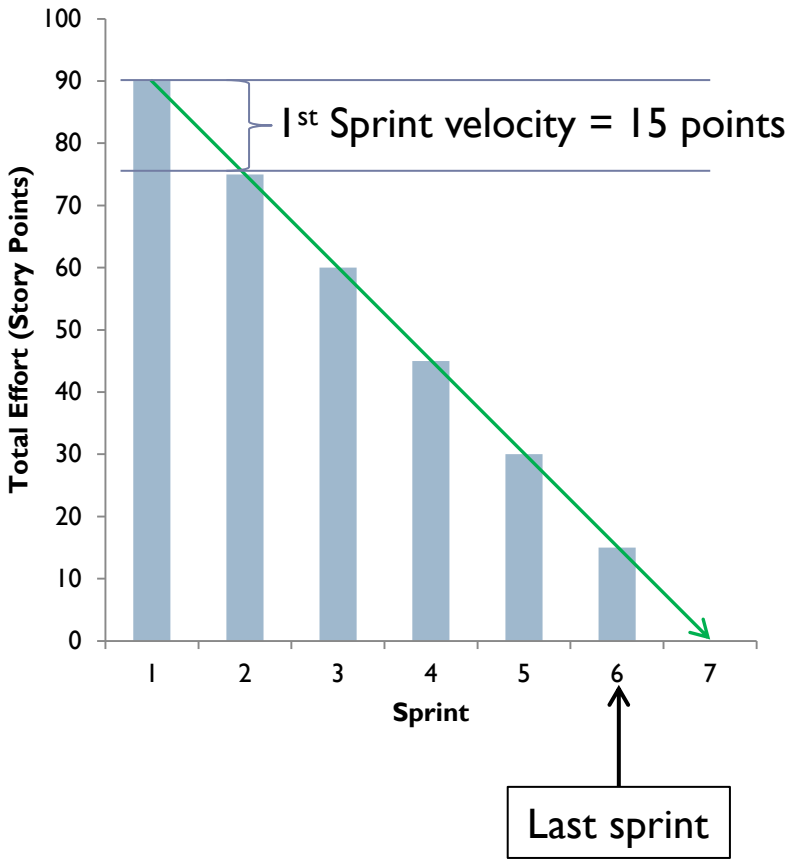
- How much work the development team has completed
- How much is left to complete
- The team's velocity in each sprint
- When you should expect to finish the product.

Types of Release Burndown Chart

- How much total work is left to complete
  - Basic release burndown chart
- The effect of changing requirements
  - Total work done release burndown chart
  - Adjustable floor release burndown
- Can be line chart or bar chart

Basic Release Burndown Chart

- X-axis – sprints
  - The last sprint is 6 in this example
- Y-axis – total effort
  - e.g., story points
- Bar – the total number of story points that are still remaining at the beginning of the sprint
- Bar difference – Sprint velocity
- Green line – prediction line
  - predict how many sprints this product will take to complete

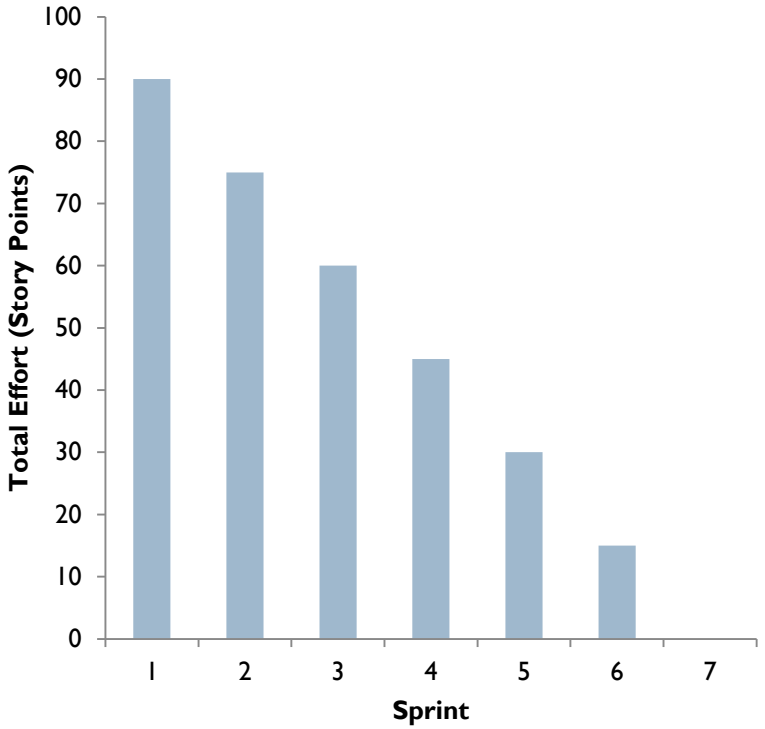


Quick Question

Your development team is about to start their sixth and final sprint. This is the release burndown chart for the project.

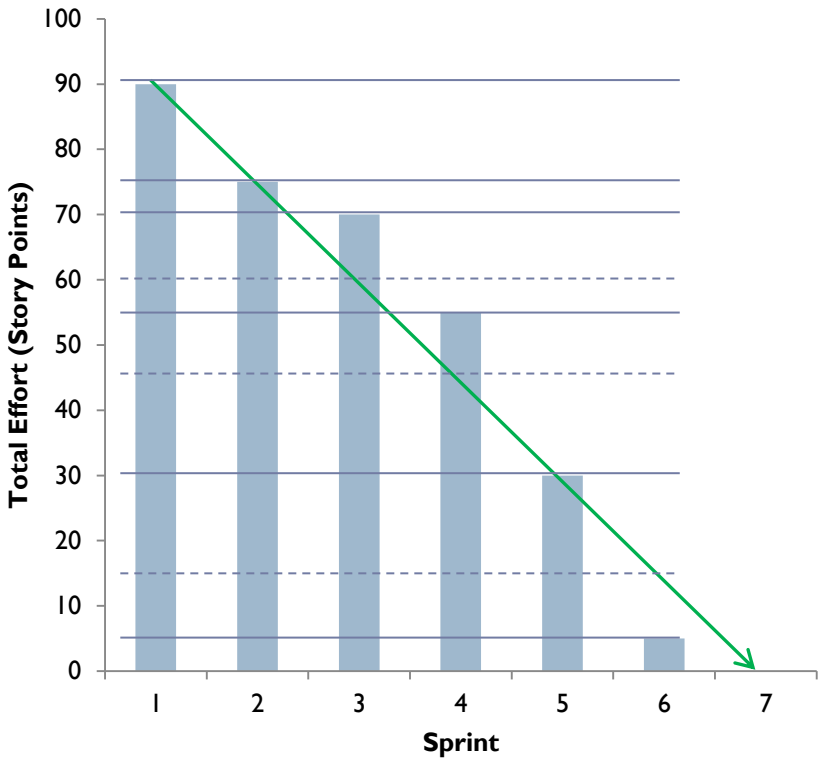
How many story points has your team completed, and how many do they have remaining?

A. 90 story points completed, 15 story points remaining
B. 90 story points completed, 0 story points remaining
C. 75 story points completed, 15 story points remaining
D. 75 story points completed, 0 story points remaining

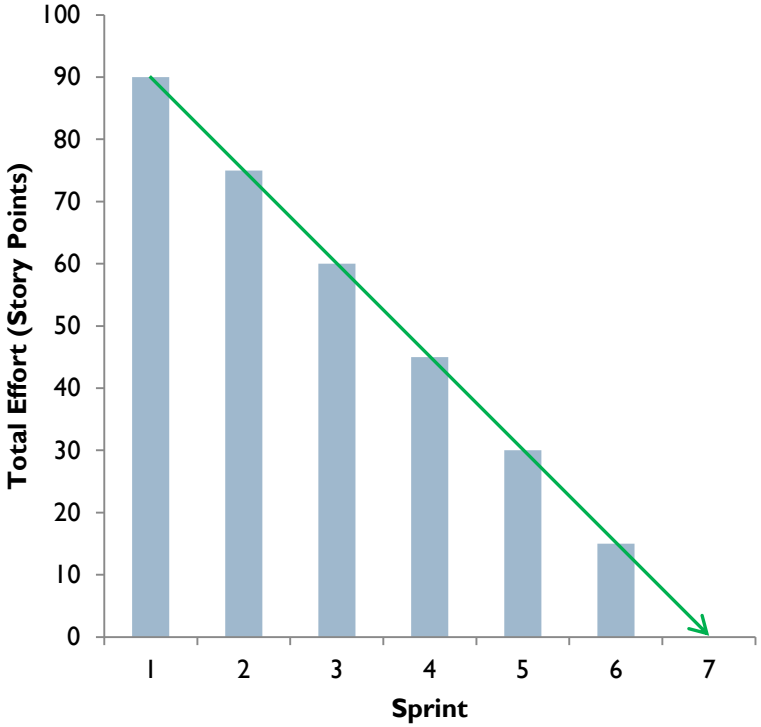


Track Development Progress

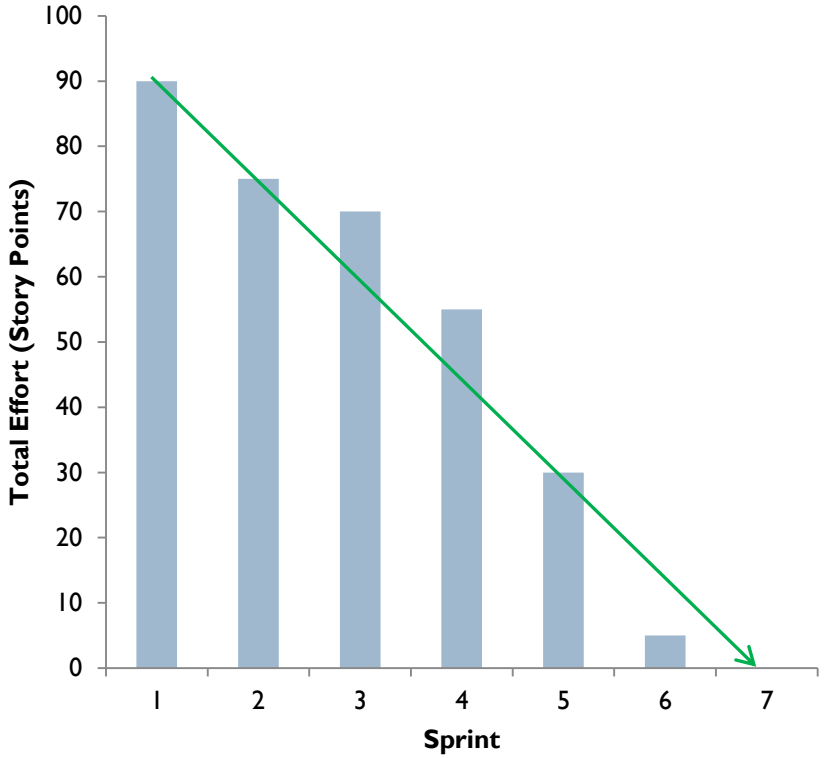
- Behind the schedule
  - bar above the prediction line
- Ahead of the schedule
  - bar below the prediction line



Ideal vs. Reality



Ideal, stable sprint velocity = 15 story points



Sprint velocity varies over time

The Definition of “Done”

- Done - only features that are programmed, tested, documented and ready for market

- Only features that have met the team's definition of done can be marked as complete for the release burndown, and have their story points deducted from the release burndown

Quick Question

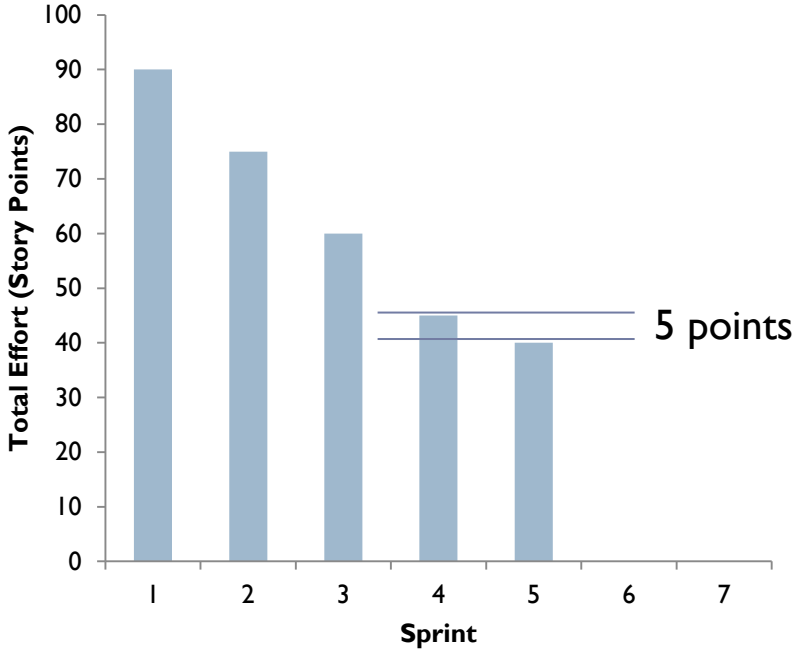
You are working on a team that is developing a mobile application, you have just completed your third sprint. Your team's definition of done includes only features that are programmed, documented and tested and ready for market.

Which of the following features should be counted as completed and have its strong points deducted within the release burndown for the sprint?

A. A feature that was developed and documented but failed testing in the last sprint has now received a minor fix and has passed tests in the current sprint
B. A feature that you did not have time to test but you are sure that it works
C. An interface feature that is programmed, documented, and tested but has not been connected to the database yet
D. A feature that works perfectly, has passed all its tests in this spring, and is properly documented

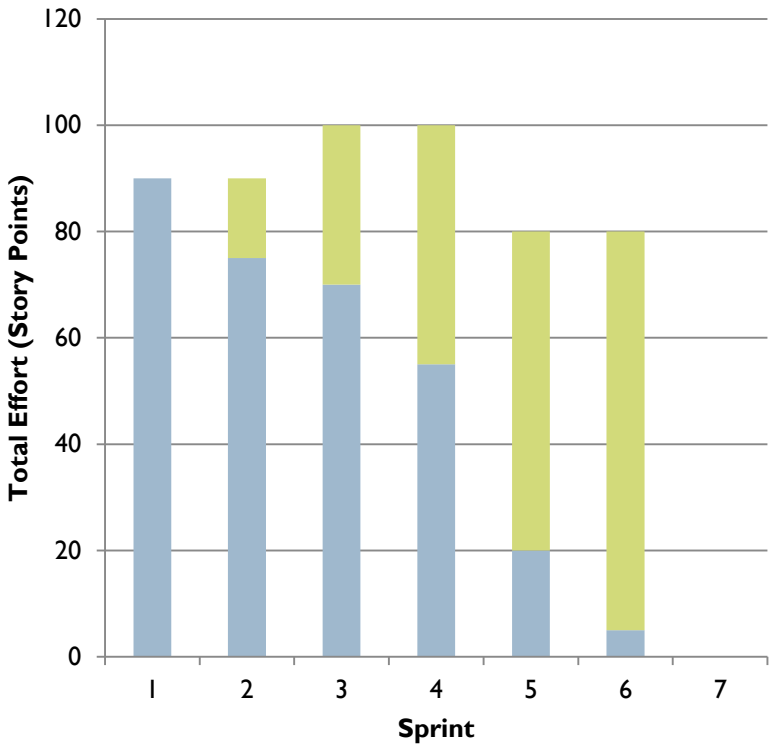
Changing Requirements

- Requirements change constantly
  - New/removed user stories
  - The estimate of story points for a user story changes
- For example,
  - In Sprint 4, the team completed 15 points
  - Before Sprint 5, two use stories are adjusted 5 more points each
  - Looks like the team only completed 5 points in Sprint 4



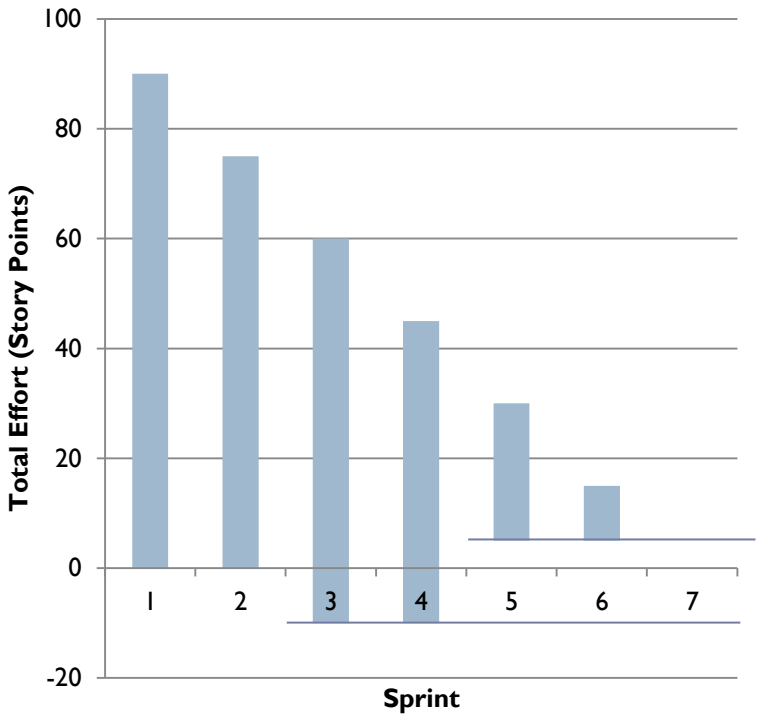
Total Work Done Release Burndown

- Stacked bar chart
  - Total work remaining on the bottom (blue)
  - Total work completed on the top (green)
- Shows your team's velocity, as well as the change in story points
  - Velocity – difference in total work completed
  - Change in story points – height of stacked bar

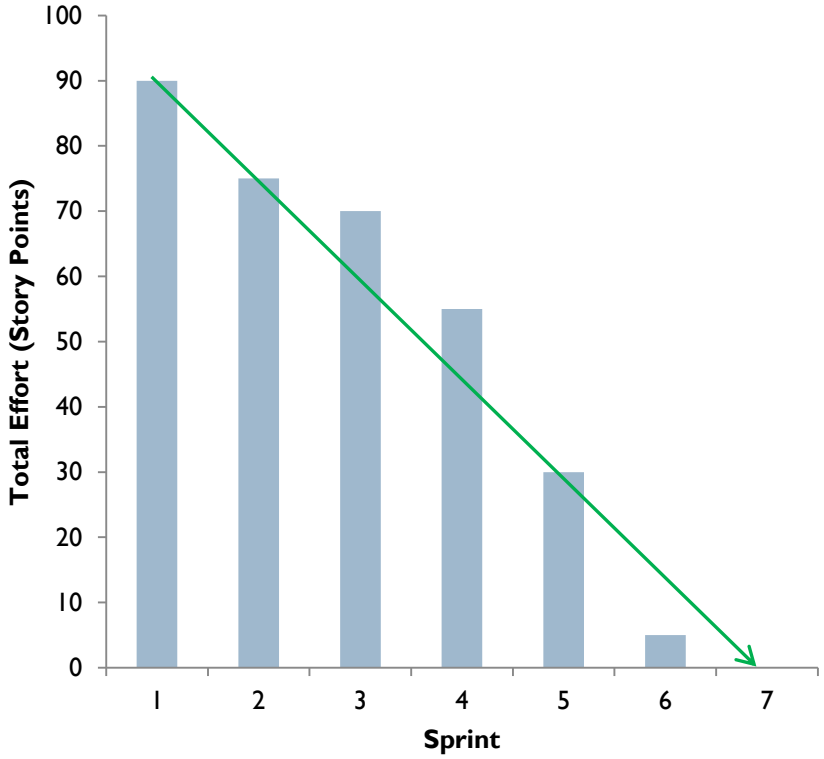


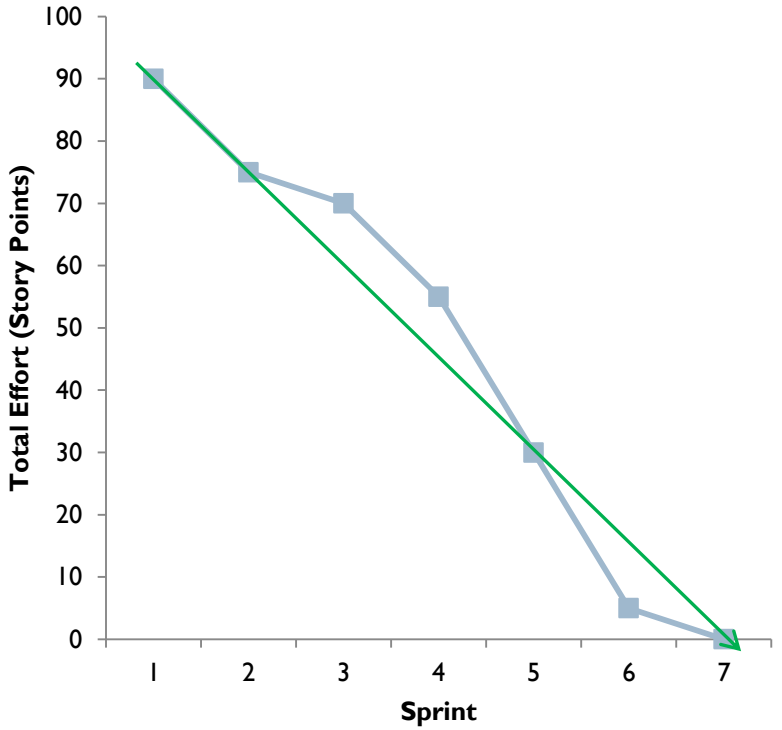
Adjustable Floor Release Burndown

- Demonstrate the change in story points by adjusting x-axis
  - Below original x-axis – added story points
  - Higher than previous x-axis – removed story points



Line Chart Release Burndown

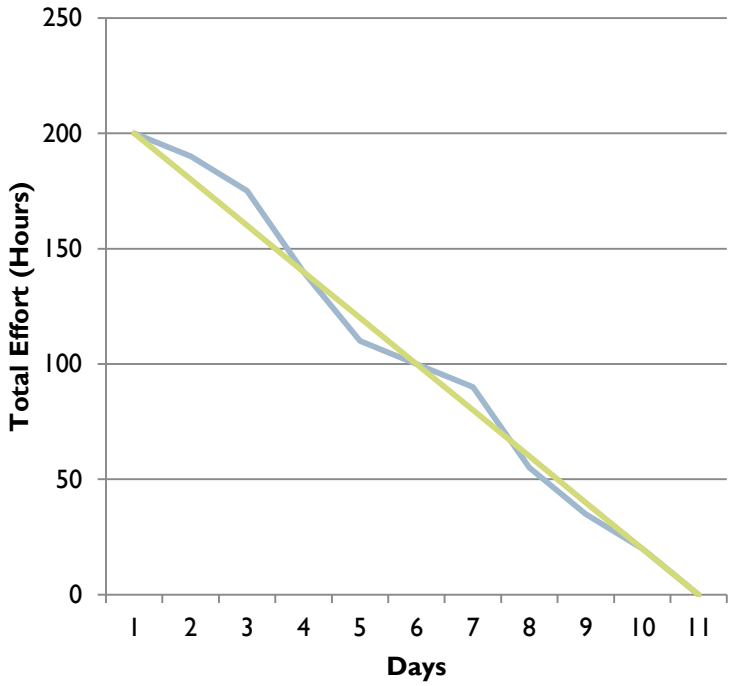




Iteration Burndown Chart

Iteration Burndown Line Chart

- X-axis – days of a sprint
- Y-axis – total effort
  - e.g., hours



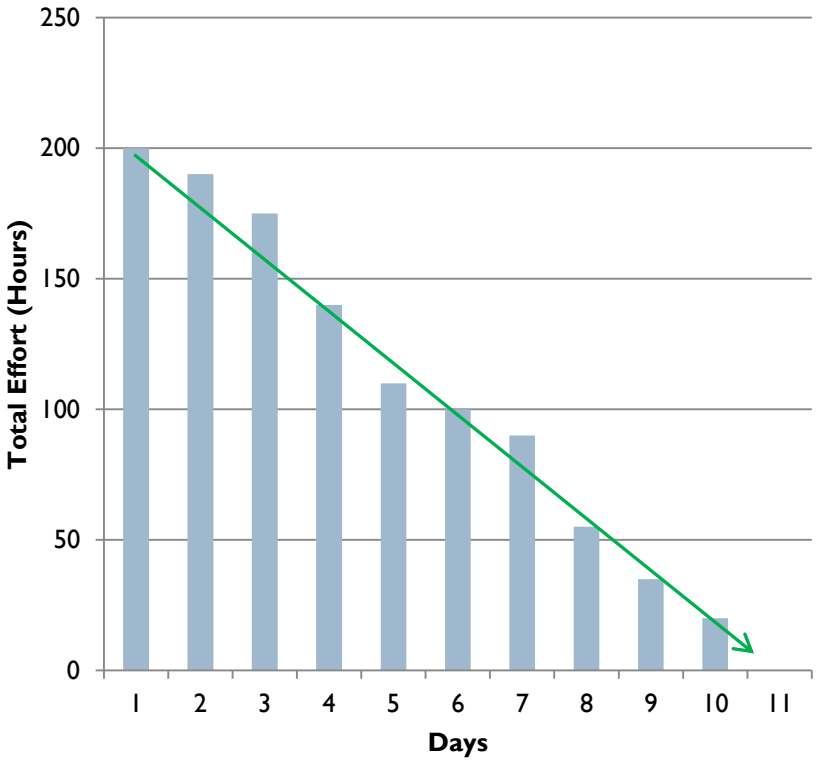
- Blue line – the total of hours that are still remaining at the beginning of a day

- Green line – prediction line

Only working days should appear on the iteration burndown. They should not include weekends or holidays, unless work is scheduled to be done on those days.

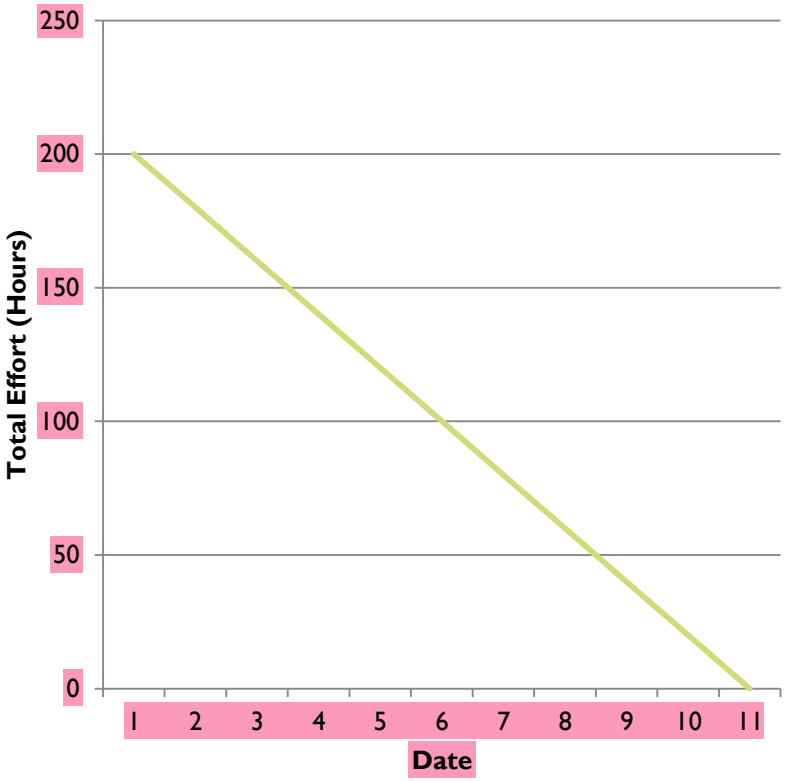
Iteration Burndown Bar Chart

- X-axis – days of a sprint
- Y-axis – total effort
  - e.g., hours
- Bar – the total of hours that are still remaining at the beginning of a day
- Green line – prediction line



Quick Question

Based on this iteration burndown chart how many person hours of work is the team aiming to complete each day of the sprint?



A. 200
B. 20
C. 18
D. You need more information

Quick Question

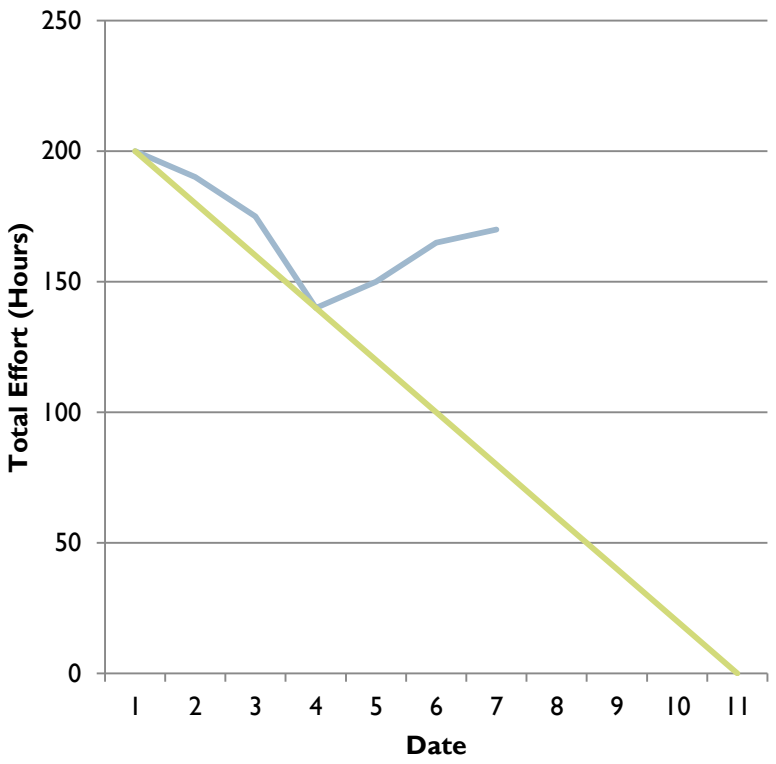
You're creating an iteration burn down chart for a development team. The team does two week sprints and has a two day weekend in the middle of the sprint. No developers work on the weekend. The team is also attending a conference for one day during the sprint, and will not be able to work on the product that day.

How many days should your iteration plan include?

A. 14 days
B. 10 days
C. 9 days

Burning Up

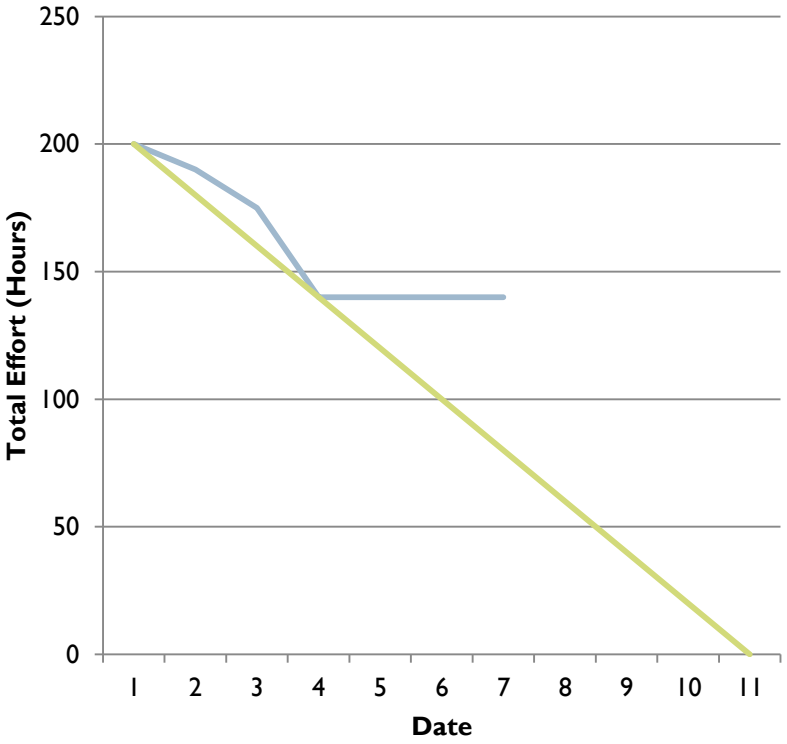
- When you are adding more tasks than you are completing, or removing
- Maybe there were tasks that were overlooked, or forgotten



Burn Across

This is the iteration burndown for your team. What could the horizontal burndown indicate?

A. your team is adding as many task hours as they are completing
B. your team is starting tasks but not completing them
C. your team is not doing any work (e.g., on vacation)
D. your team is completing tasks at a constant pace



Solutions to Burn-Up or Burn-Across

- Burn-up
  - Removing or prioritizing some tasks
- Burn-across
  - Prioritizing completing active tasks over starting new tasks

Make sure that all your task are not in half done states at the end of the sprint