Week 13 - Monday

COMP 3100

Last time

- What did we talk about last time?
- Scheduling
  - Tasks
  - Gantt charts
  - Critical path method

Questions?

Project 4

Execution and Control

Management in traditional projects

- In traditional projects, managers hire people and make sure they're trained for the project
- Managers dictate quality assurance processes, data collection practices, risk monitoring, and other work processes
- Managers might delegate responsibility to people in charge of some of these processes

Management in agile projects

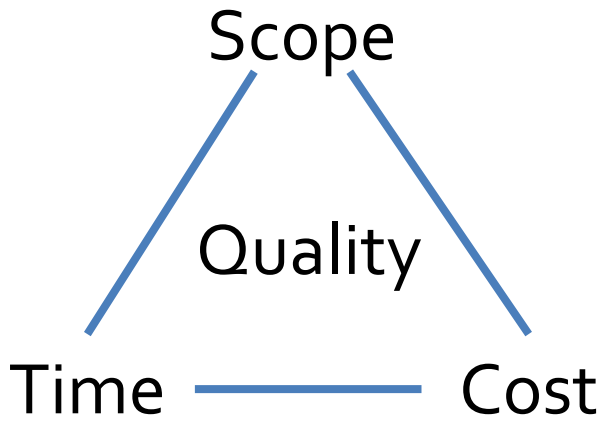
- Agile teams might be new for a project or existing from previous projects
- Agile methodologies like Scrum often require training so that people know how to work in the agile environment
  - Managers are responsible for this training
- The details of applying Scrum (or whatever agile methodology) are left up to team members (non-managers)
- Management is usually much more decentralized in agile

Control in traditional projects

- Managers have to track progress
  - If a task is done early (sometimes, it happens!), the project is ahead of schedule
  - If a task is done late, the project is behind schedule
  - Similarly, cost might be above or below expectations
- If progress doesn't meet expectations, actions must be taken:
  - Reduce the features but stay on schedule
  - Add more resources (people) but stay on schedule
  - Delay the delivery date, which probably also increases costs
  - Reduce costs by shrinking the scope, firing people, or shortening the delivery date

Back to the iron triangle

- There's a graphical depiction of project management used imply relationships between time, scope, cost, and quality



- This triangle is intended to indicate that you can't change scope, time, or cost without affecting the other two (at least if you want to maintain quality)
- Increasing scope means increasing time or cost (or both)
- It's obvious, but managers are sometimes tempted to push workers to work faster, for example, pretending there are no consequences

Brooks' Law

- Fred Brooks is a Turing Award-winning computer scientist who wrote *The Mythical Man-Month*, a book about software engineering
- **Brooks' Law:** "Adding programmers to a late project makes it later."
  - New people have to be trained on the project by existing employees
  - This effect gets worse with projects that are close to done since there's more to learn
- There are also lower bounds on how fast a project can get done no matter how many people you throw at it
  - Some tasks can only be done effectively by a single person
- Presumably, there's an ideal team size for a given project, but we do not (yet) know how to estimate it

Consulting with stakeholders

- Stakeholders get mad when bad news is sprung on them at the last minute
  - Product will be delayed
  - Product won't do what it's expected to do
  - Product will cost more than expected
- When problems arise, managers should consult with stakeholders to see how they want to proceed
- People prefer having input into the response to a bad situation

Control in Scrum

- Everything revolves around sprints in Scrum
- We measure progress against the amount of work to be done using burn charts
- **Burn charts** have a vertical axis of work and a horizontal axis of time
  - **Burn down charts** show work remaining
  - **Burn up charts** show work completed
- In addition to measuring progress *within* a sprint, burn up charts are sometimes used to measure progress toward a release
  - If the release is taking too many sprints, stakeholders can choose to reduce features or delay the schedule (and increase cost)

Earned value management

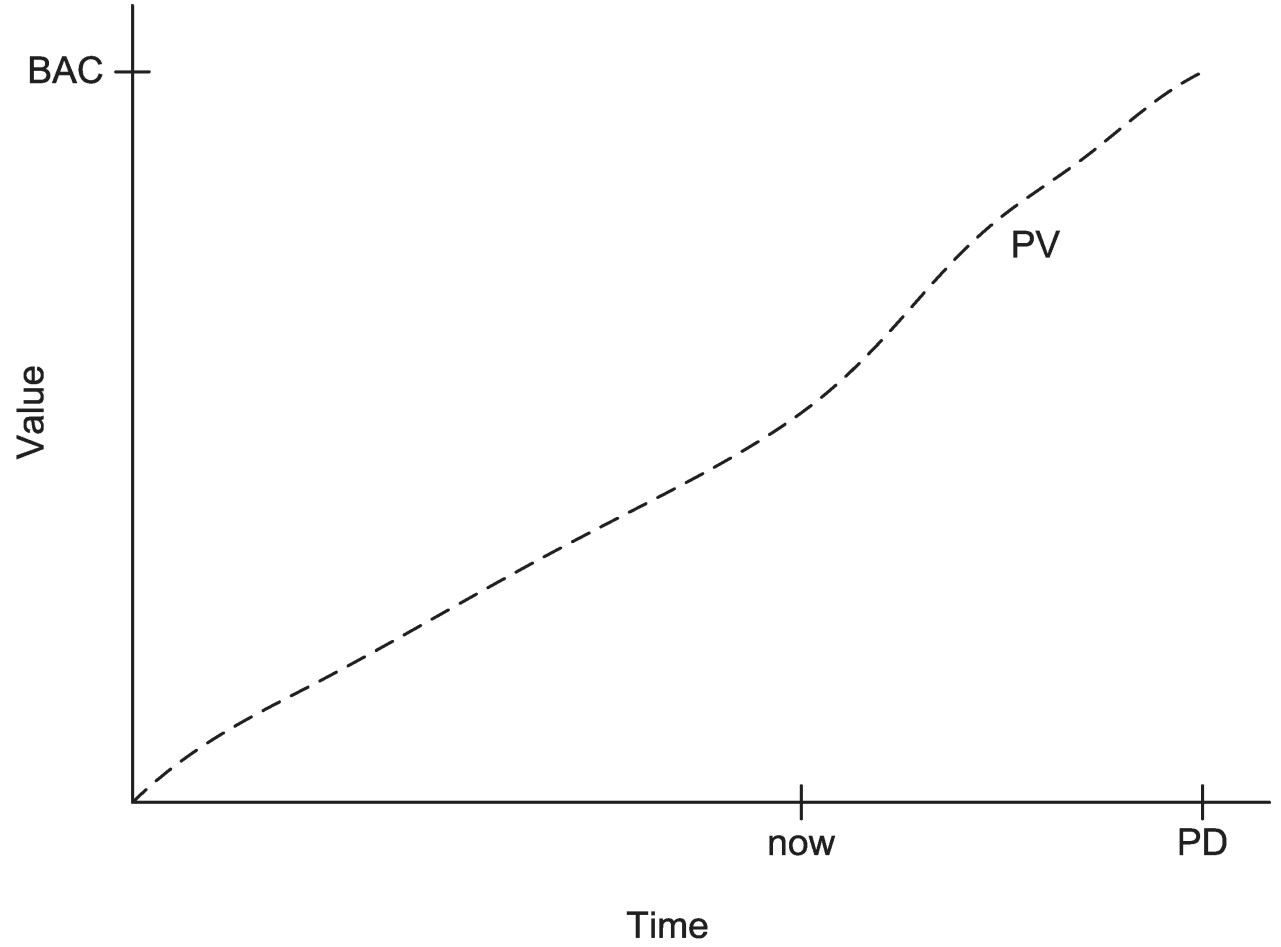
- If a job is expected to take 100 hours, and you've worked for 50, are you halfway done?
  - Probably not!
- **Earned value management** (**EVM**) (or **earned value analysis** (**EVA**)) tries to solve this problem
  - **Progress** is how much of the overall project is complete
  - **Health** is a comparison of how much you thought you'd get done with how much you did get done
  - Value can be measured in person-days or in dollars

More on EVM

- We have talked about ways to decompose a project into tasks and how to estimate the effort or cost of each task (even if that's still a hard problem)
  - **Planned value** (**PV**) is the estimated cost of a given task
- Using a Gantt chart or CPM, you can make a schedule for all of your tasks
  - **Planned duration** (**PD**) is the estimated time for the entire project
- With the PV for every task and a schedule, you can graph the growth of PV over time
- This line ends at the **PD**, giving the **Budget At Completion** (**BAC**), the estimated cost of the whole project

Earned value management example

- The graph below shows an example of what the PV for a product might be

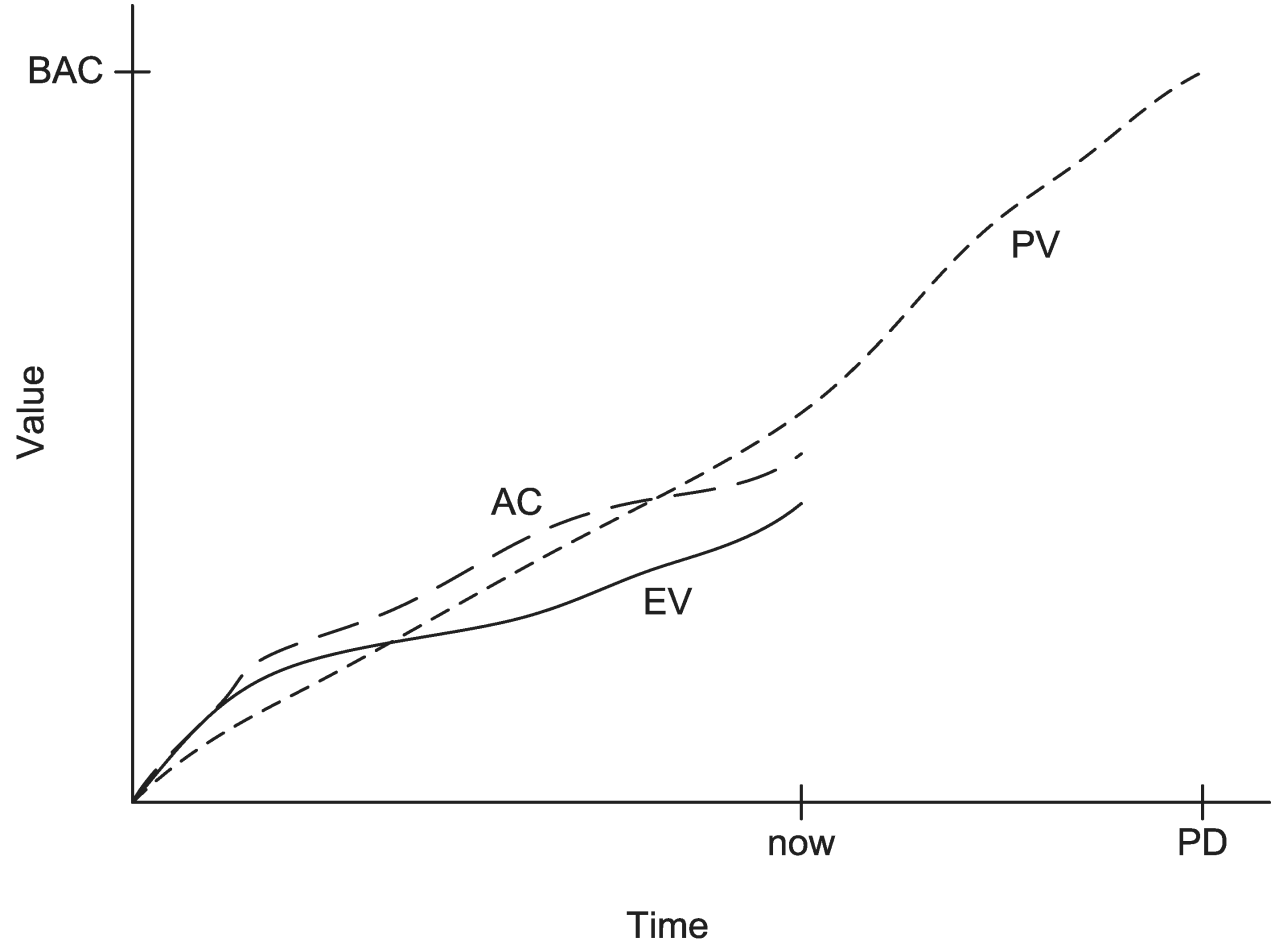


Earned value and actual cost

- We can map out PV before the project is really underway
- While the project is going, we're interested in two more values for each point in time:
  - **Earned value** (**EV**): The planned value of the tasks that are done
  - **Actual cost** (**AC**): The effort or money spent on getting those tasks done
- It seems strange that we can have *three* different values for each point in time, but what we plan to do differs from what we get done, and what we get done doesn't always cost what we think it will

Earned value management with PV, EV, and AC

- The graph below shows an example of relationships between PV, EV, and AC



Quantifying project problems

- Where the EV is relative to the PV shows how much has been gotten done relative to what was planned
  - If the EV line is below the PV, the project is behind
  - If the EV line is above the PV, the project is ahead
- Where the AC is relative to the EV shows how much value has been expended relative to how much value was completed
  - If the AC is above the EV, the project is over budget
  - If the AC is below the EV, the project is under budget

More on quantifying project problems

- The **Schedule Performance Index** (**SPI**) is EV/PV
  - When SPI = 1, the project is right on schedule
  - When SPI < 1, the project is behind schedule
  - When SPI > 1, the project is ahead of schedule
  - Example: Here, the SPI = 360/400 = 0.9, meaning behind schedule
- The **Cost Performance Index** (**CPI**) is EV/AC
  - When CPI = 1, the project is right on budget
  - When CPI < 1, the project is over budget
  - When CPI > 1, the project is under budget
  - Example: Here, the CPI = 360/378.94 = 0.95, meaning over budget

Even more on quantifying project problems

- Using these numbers, we can predict how well the project is doing
- **Forecast Project Duration** (**FPD**) is PD/SPI = PD/(EV/PV)
  - The FPD estimates the true project duration based on how far or ahead the project is at a given time
  - Example: In this case, assuming a planned duration of 30 months, FPD = 30/0.9 = 33.33 months
- **Estimate At Completion** (**EAC**) is BAC/CPI = BAC/(EV/AC)
  - The EAC estimates the true project cost based on how much it has cost to get tasks done so far
  - Example: In this case, assuming a budget at completion of 1140 (whatevers), EAC = 1140/0.95 = 1200
- Using these estimates and EVM charts, managers can make decisions about what to do when things are going wrong

Burn charts

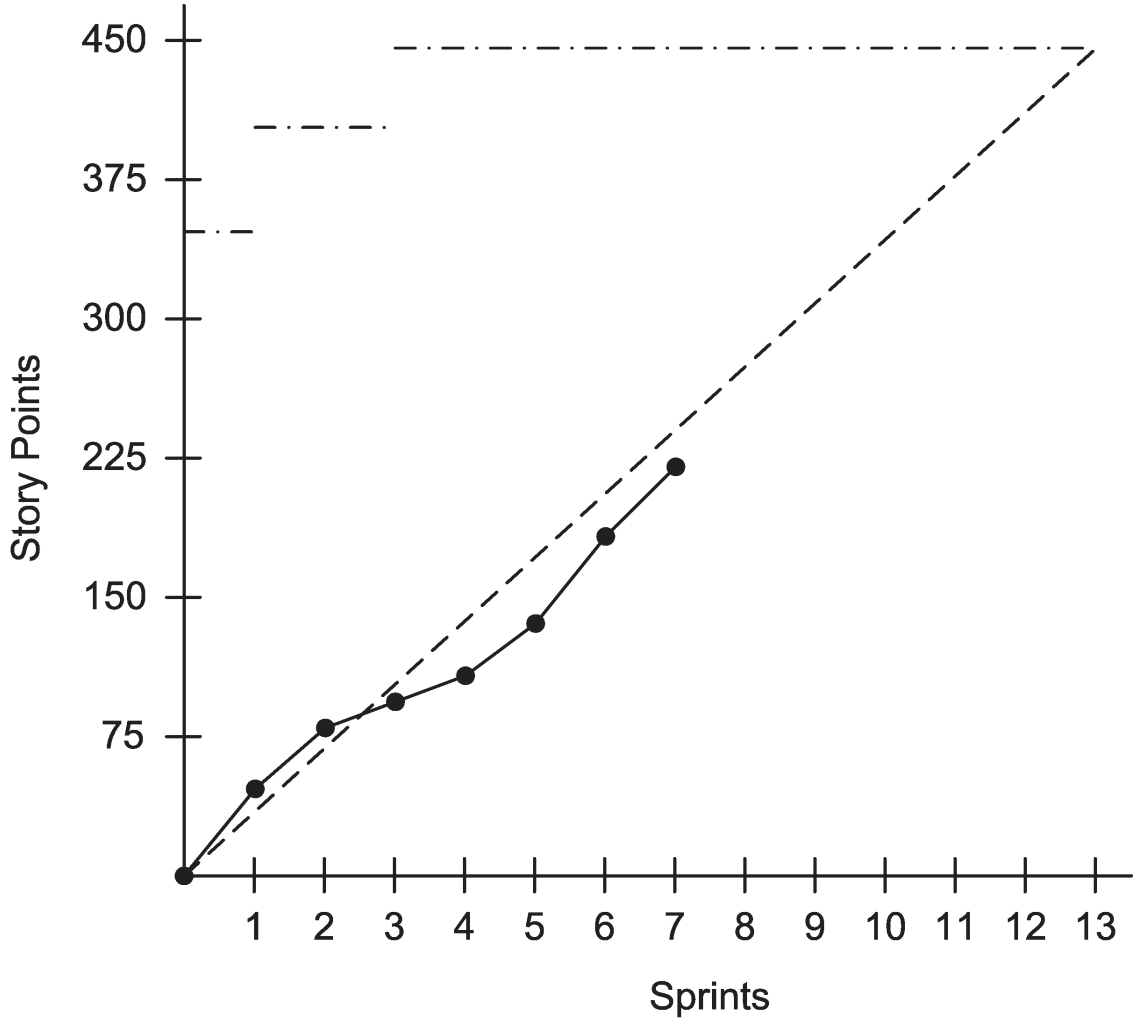
- In Scrum (and other agile methods), burn charts are used to track progress during sprints and over many sprints
- Burn charts are simplified EVM charts
  - Vertical axis: Some measure of value: PBIs, story points, ideal days, monetary units
  - Horizontal axis: time
- Planned value over time is plotted in whatever units are being used to measure value
  - These plots are linear, assuming a constant velocity (story points per day, for example) of the team
- Then, actual progress is plotted as work is done
- As with other EVM charts, the relationship of the work done to the planned work shows the health of the sprint or the overall project

Burn up and burn down

- Burn up charts show how much value has been completed over time
- Burn down charts show how much value is left to produce
- For (probably historical) reasons:
  - Burn up charts are more often used to track progress on full projects or releases
  - Burn down charts are more often used to track progress within a sprint

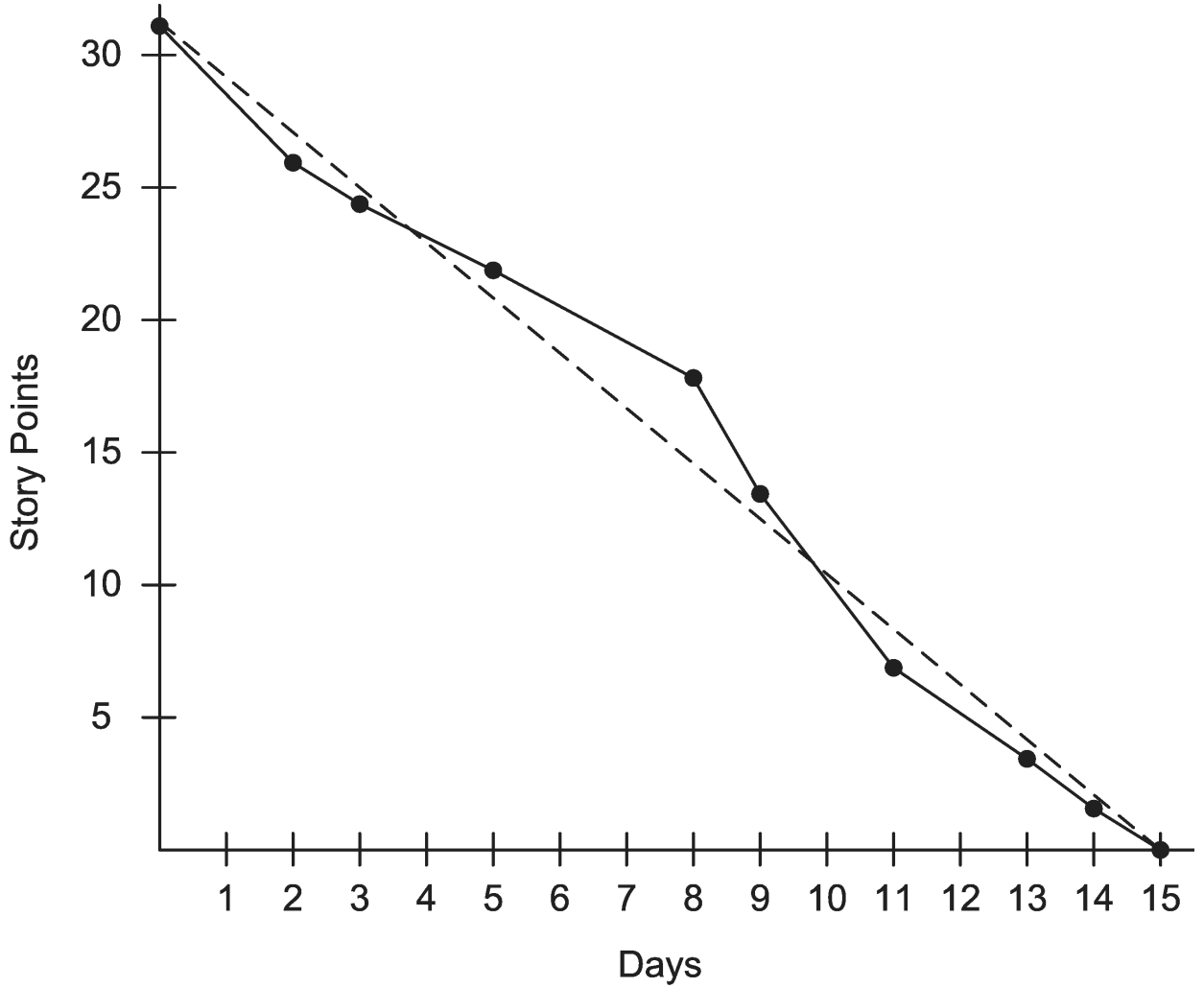
Burn up chart example

- This burn up chart shows progress over a whole project
  - 13 sprints are planned
- Solid black lines show real progress against planned progress
- The horizontal lines on the left show estimates of total story points needed for the project
  - They were revised up from about 350 to 450 over time
- Although velocity is improving, we'll probably have to add sprints or reduce features



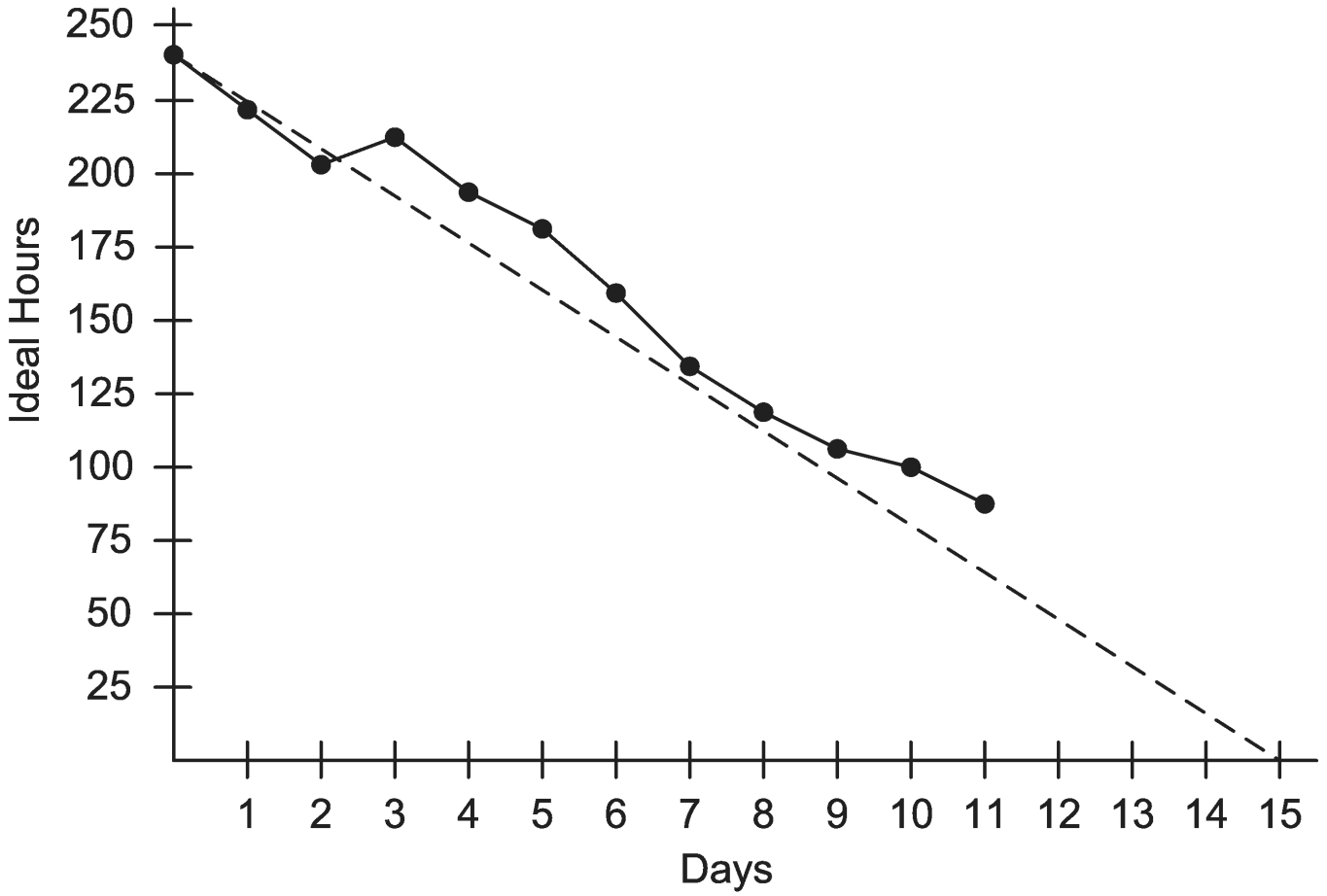
Burn down chart example

- This burn down chart shows progress made during a three-week sprint
- Each dot shows the completion of a PBI
- As before, the solid line shows real progress against the dotted line of planned progress
- If all the story points had been finished early, the team could work with the PO to add more to the sprint backlog
- If some PBIs had not been completed, they go back to the product backlog



Another burn down chart example

- Instead of story points, estimations can be in terms of ideal hours
- Story points for a PBI don't usually change, but ideal hours might
- This burn down chart shows new estimates after day 3, when the team realized that some PBIs were going to take more ideal hours



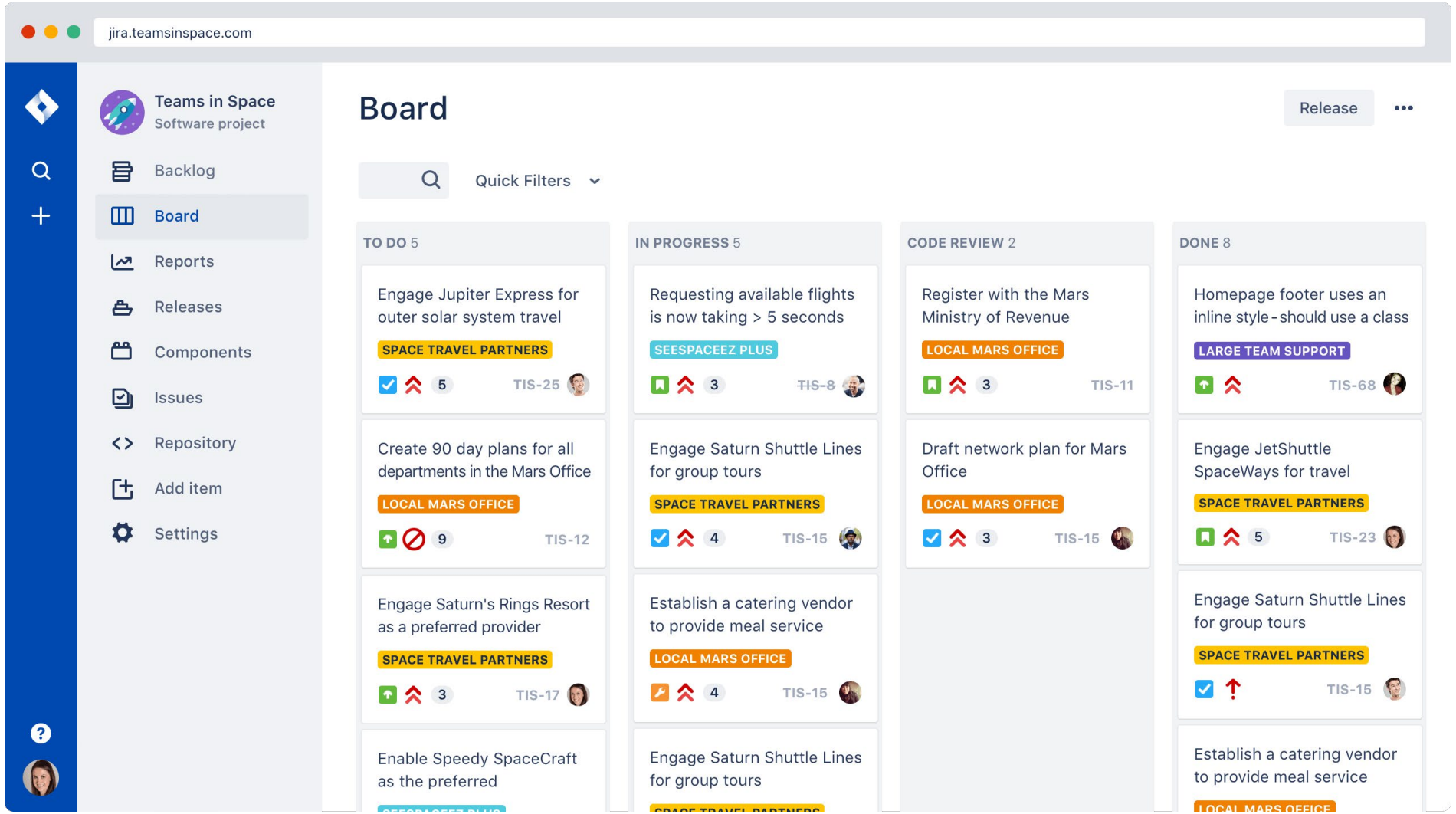
Task boards

- **Task boards** are boards showing the current status of all tasks for a sprint
- Tasks are often shown in rows corresponding to a PBI
  - Columns might be: To Do, In Progress, In Testing, Done, and similar
- Your Trello boards are approximations of the task boards used in real development
- Tasks boards are not as analytical as burn charts, but they can help in other ways:
  - Too many things in the To Do column late in a sprint means that PBIs aren't all going to get done
  - Something stuck in the In Progress column for a while means that more developers should help with it
  - If it looks like some PBIs are impossible to finish this sprint, the team can focus on tasks to get PBIs done that are possible

Task board example from book

PBI	To Do	In Progress	Done
As a user, I want to insert tree nodes	Inspect node insertion UI code	Inspect tree node insertion code Test node insertion UI code	Modify tree classes for node insertion Test tree node insertion code Code tree node insertion UI
As a user, I want to delete tree nodes	Inspect tree node deletion code Inspect node deletion UI code Test node deletion UI code	Test tree node deletion code Code tree node deletion UI	Modify tree classes for node deletion

Task board example from Jira



Upcoming

Next time…

- Wednesday is an extra work day

Reminders

- Keep working on Project 4