CSC 171 – Module 7

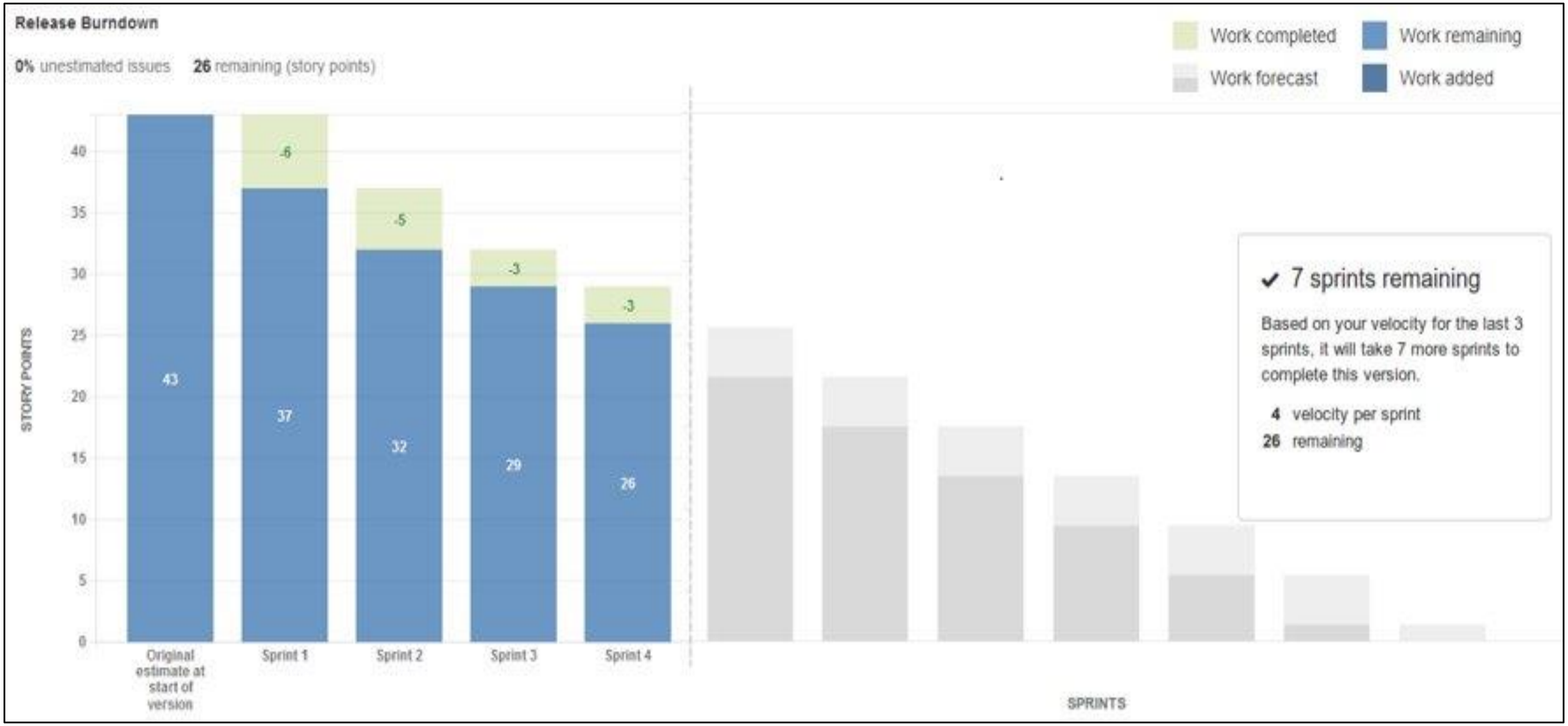
Tracking and Communicating

Monitor the Release Plan

- Where we are relative to the goal of completing a certain amount of work in a given amount of time?
  - Factors to consider
    - progress made, changes in the scope, changes in estimate
- Velocity
  - Number of story points (ideal days/hours or features) completed per iteration
  - Complete means code
    - is well written, well factored, checked-in, and clean
    - complies with coding standards
    - passes all tests
  - If a user story is not completed by the end of iteration
    - do not count it in the iteration's velocity (recommended)
      - It is hard to measure incomplete work
        - » e.g., How far along if a user story is partially implemented and tested?
      - It breaks down the trust between the development team and the customer team
      - It increases WIPs and reduces throughput
    - or split it into smaller user stories, including completed and unfinished ones
      - In rare cases, such as the incomplete user story will not be worked on in the next iteration
- Release burndown/burnup charts, cumulative flow charts, iteration contents charts, product backlog burnup chart

Release Burndown Chart
- Example without Scope Change

The vertical axis shows the number of story points (ideal days/hours or features) remaining
The horizontal axis shows iterations



Release Burndown

0% unestimated issues **26** remaining (story points)

✓ 7 sprints remaining

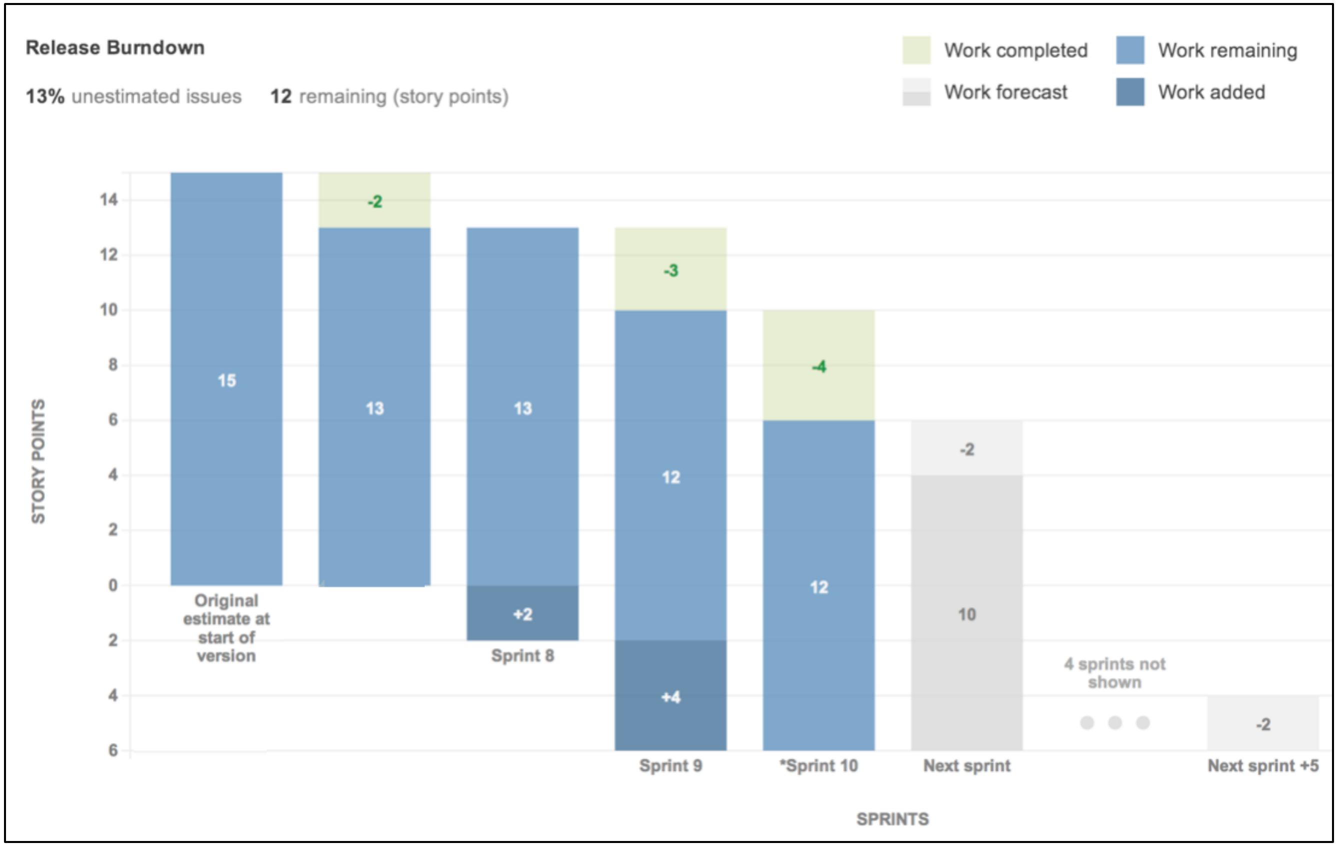
Based on your velocity for the last 3 sprints, it will take 7 more sprints to complete this version.

4 velocity per sprint
26 remaining

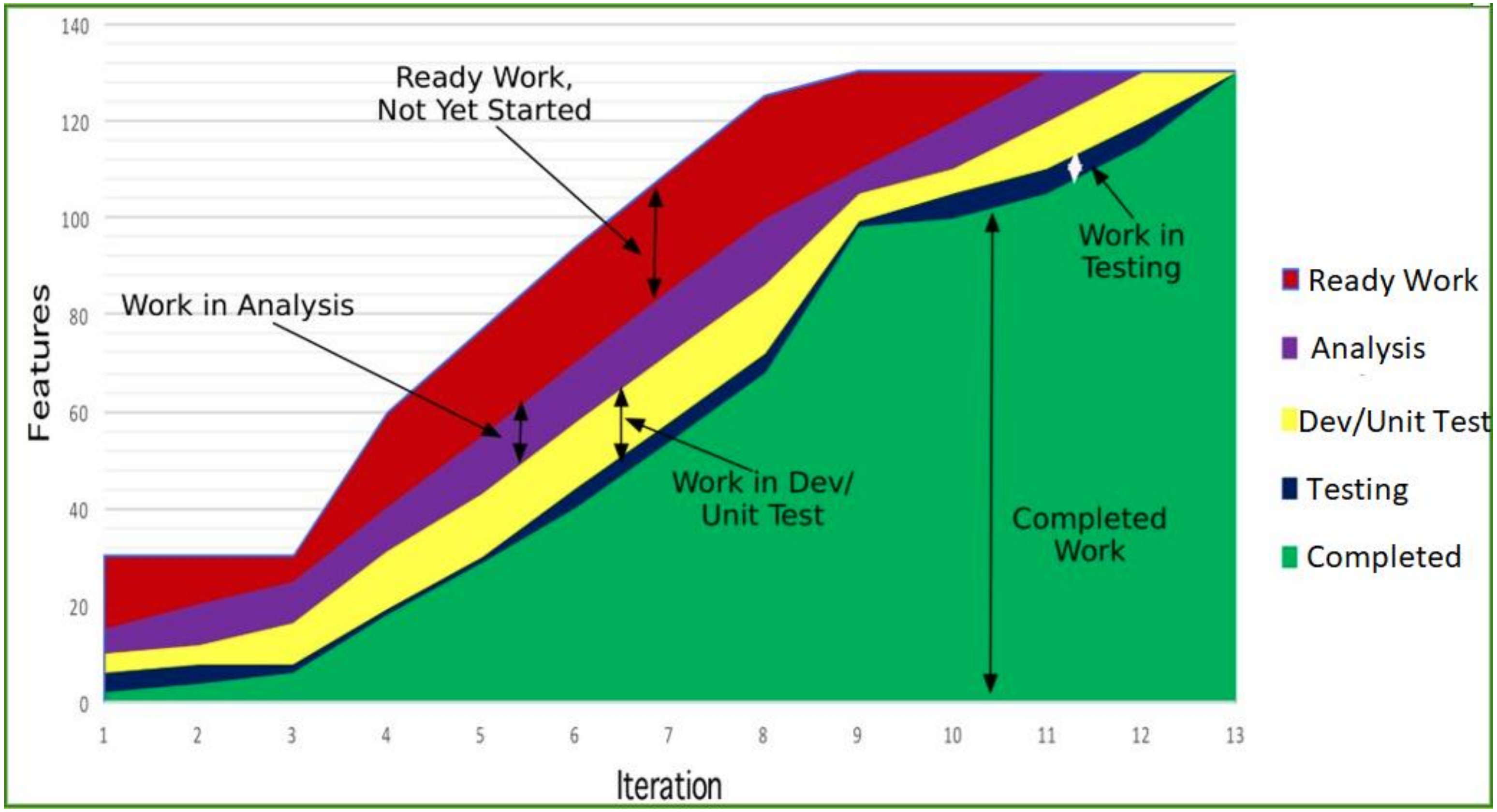
Release Burndown Chart
- Example with Scope Change

- Rules
  - Any time work is completed, the top is lowered
  - When work is re-estimated, the top moves up or down
  - When new work is added, the bottom is lowered
  - When work is removed, the bottom is raised

- The following burndown chart shows scope changes
  - 2 story points were added to Sprint 8
  - 4 story points were added to Sprint 9



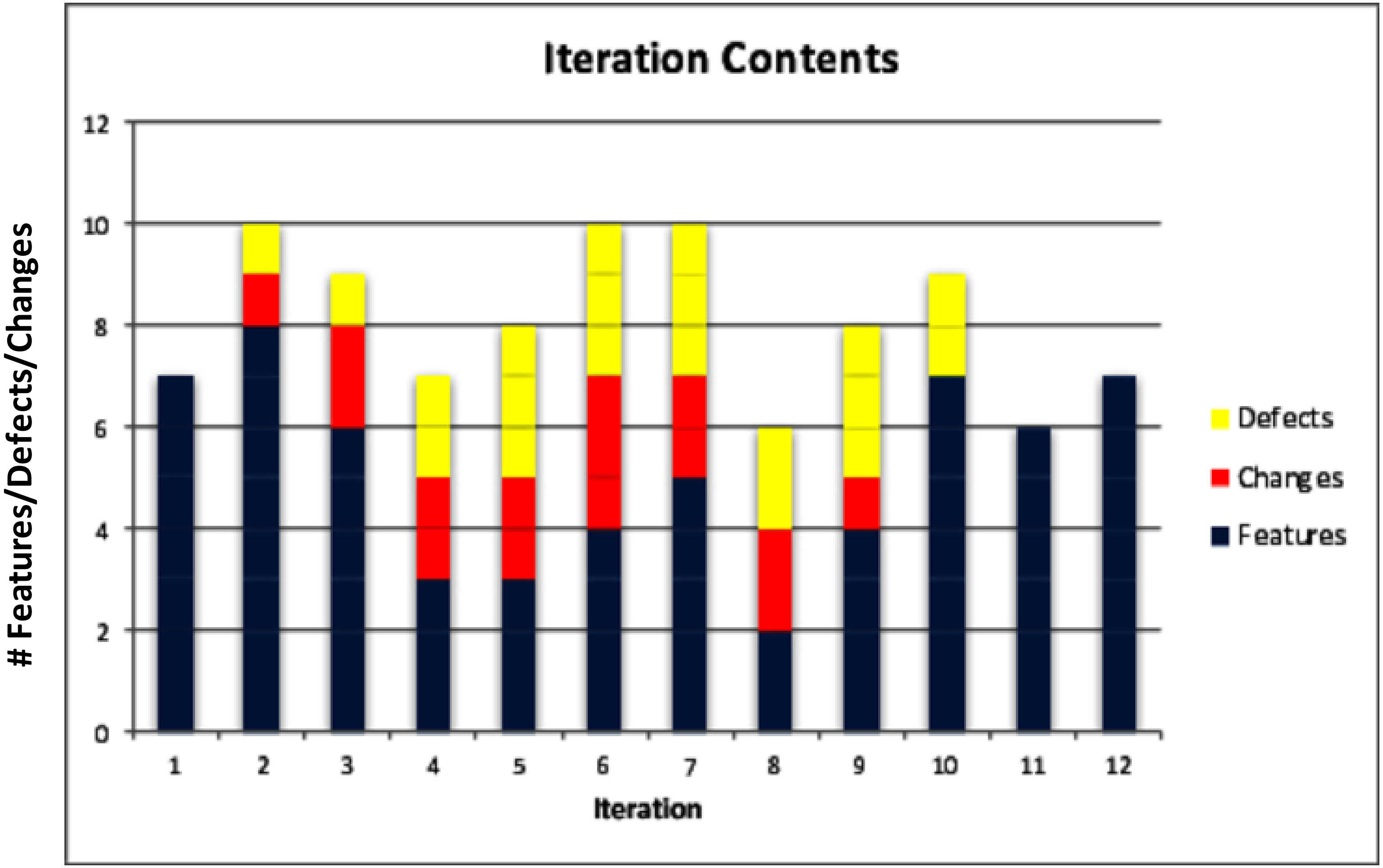
Cumulative Flow for Release



Use cumulative flow charts to see WIP

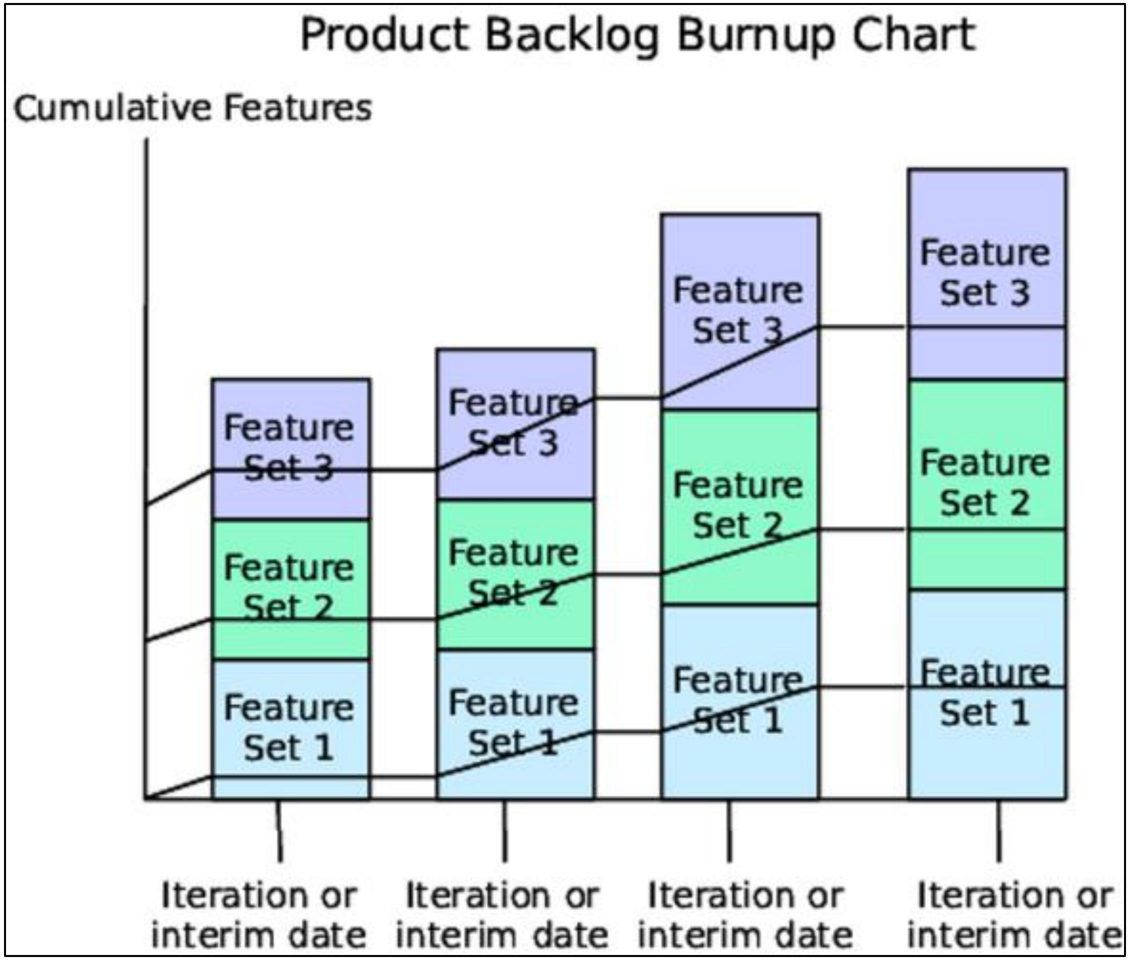
- In this example, WIP includes Work in Analysis, Work in Dev/Unit test, and Work in Testing
- Each band represents a work stage, and has an arrival line and a departure line
  - If the arrival and the departure lines progress in parallel, that indicates stable process
  - If the distance between the arrival and departure lines widens, that stage is often a bottleneck
- Also shows cycle time

Iteration Contents Chart



Use iteration contents charts to show what the team did not just what they planned. Changes in the above represent urgent work from a support queue.

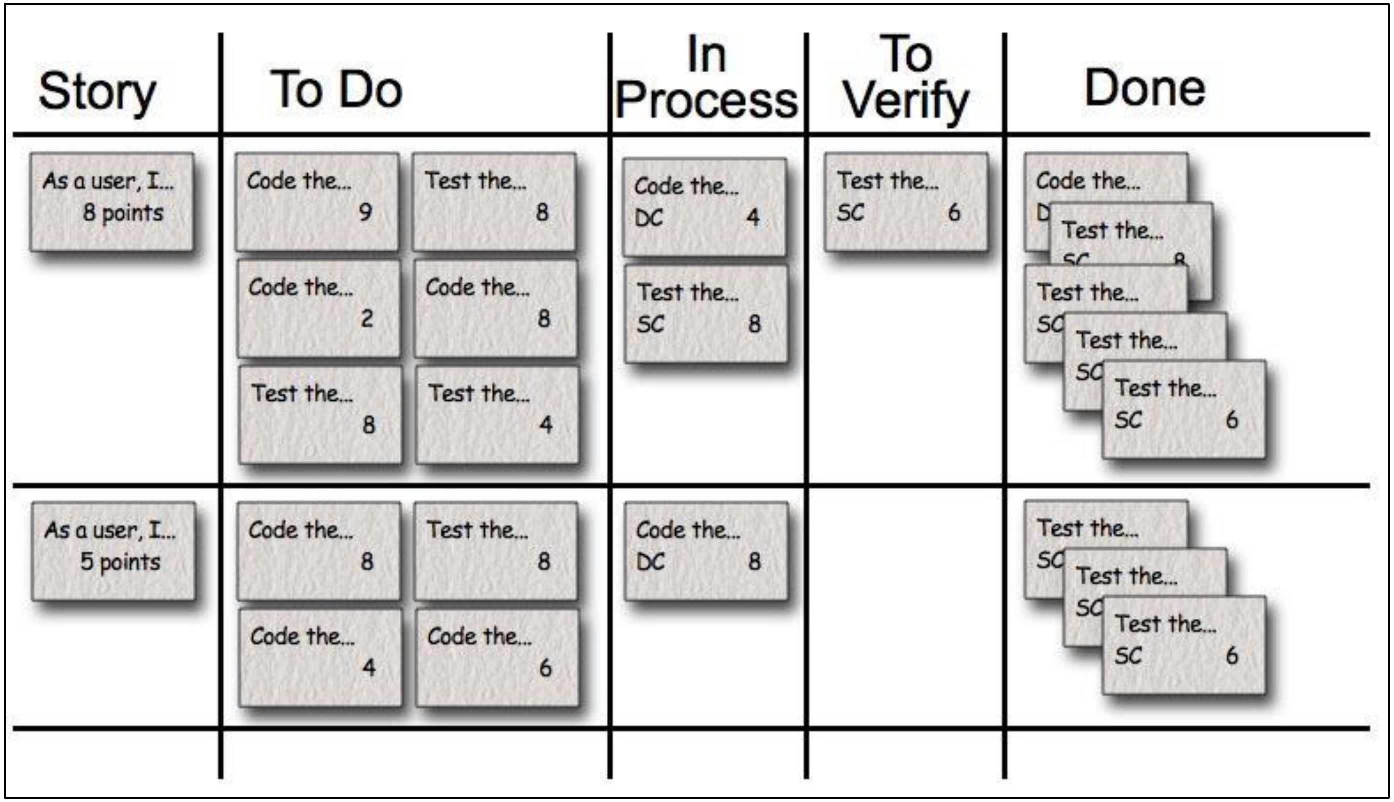
Product Backlog Burnup Chart



Use product backlog burnup chart to show progress against the planned and additional features. For each feature set, the above chart shows the precentage completed for that set.

Monitor the Iteration Plan
- Task Board

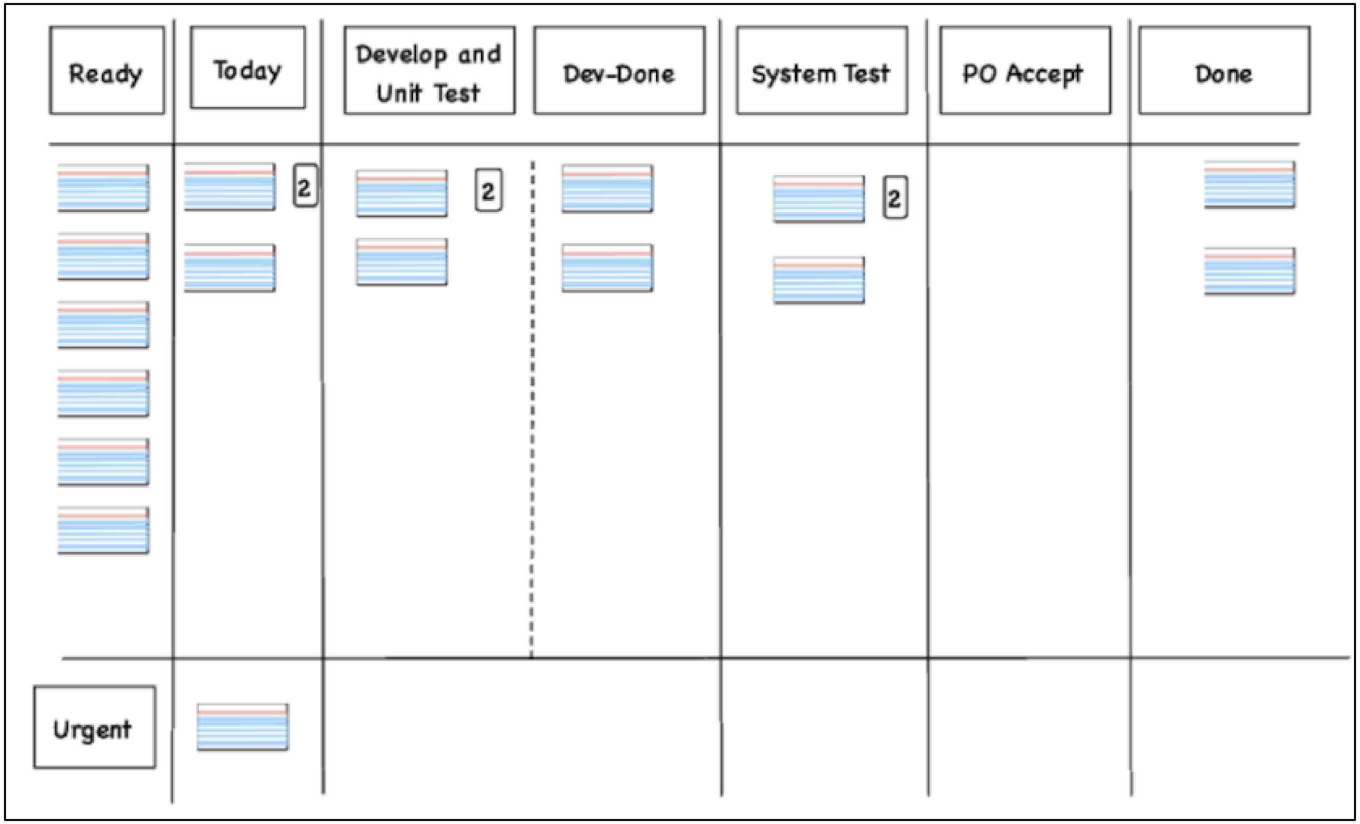
- One row for each user story
- Each card in the first column represents a user story, the first column does not change during the iteration
- Each card in other columns represents a task associated with the user story in that row
- Columns (except the first one) represents stages of tasks, team decide what columns to use
- An example [3]



Story	To Do	In Process	To Verify	Done
As a user, I... 8 points	Code the... 9 Test the... 8 Code the... 2 Code the... 8 Test the... 8 Test the... 4	Code the... DC 4 Test the... SC 8	Test the... SC 6	Code the... D Test the... SC 8 Test the... SC Test the... SC Test the... SC 6
As a user, I... 5 points	Code the... 8 Test the... 8 Code the... 4 Code the... 6	Code the... DC 8		Test the... SC Test the... SC Test the... SC 6

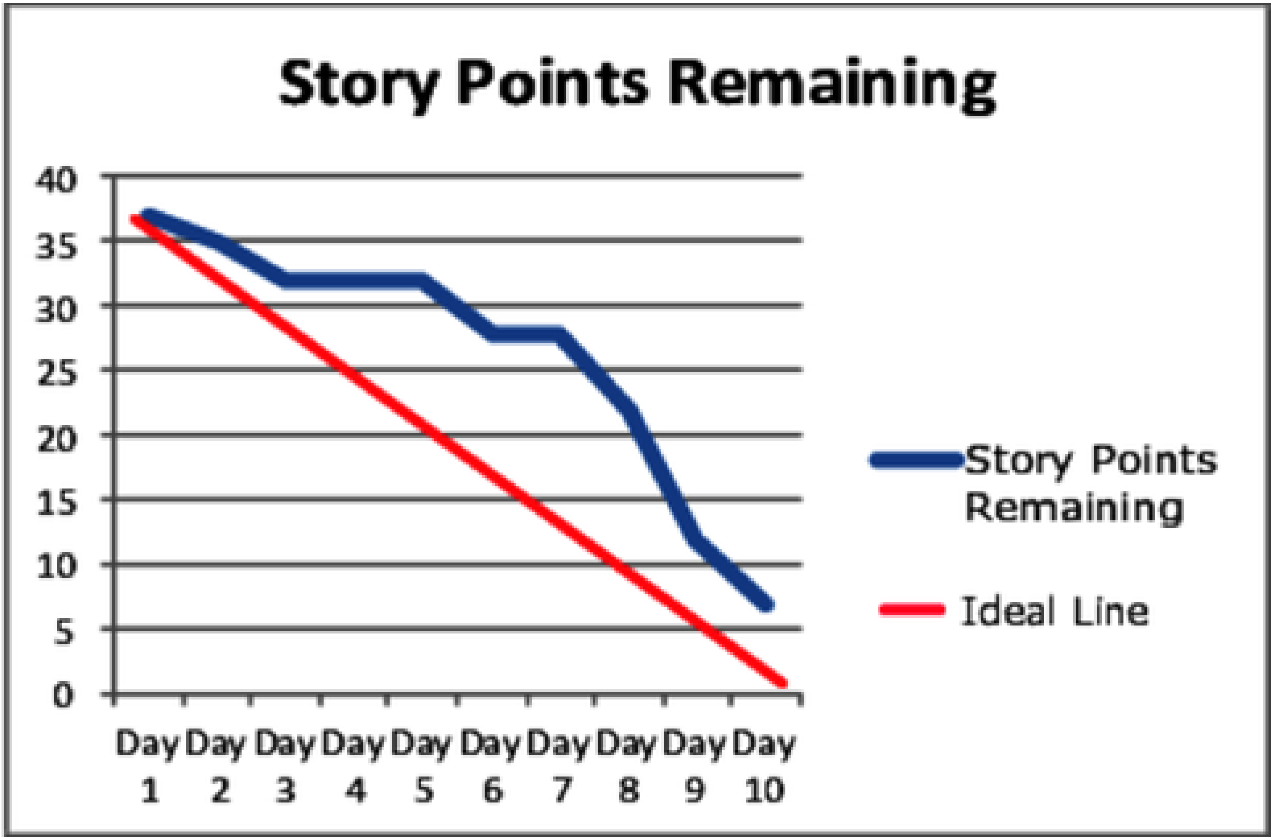
Monitor the Iteration Plan - Flow Board

- Each card represents a user story
- Columns represent flow, which is decided by the team
- Urgent queue can be used for interruptions such as support work
- Example [1]



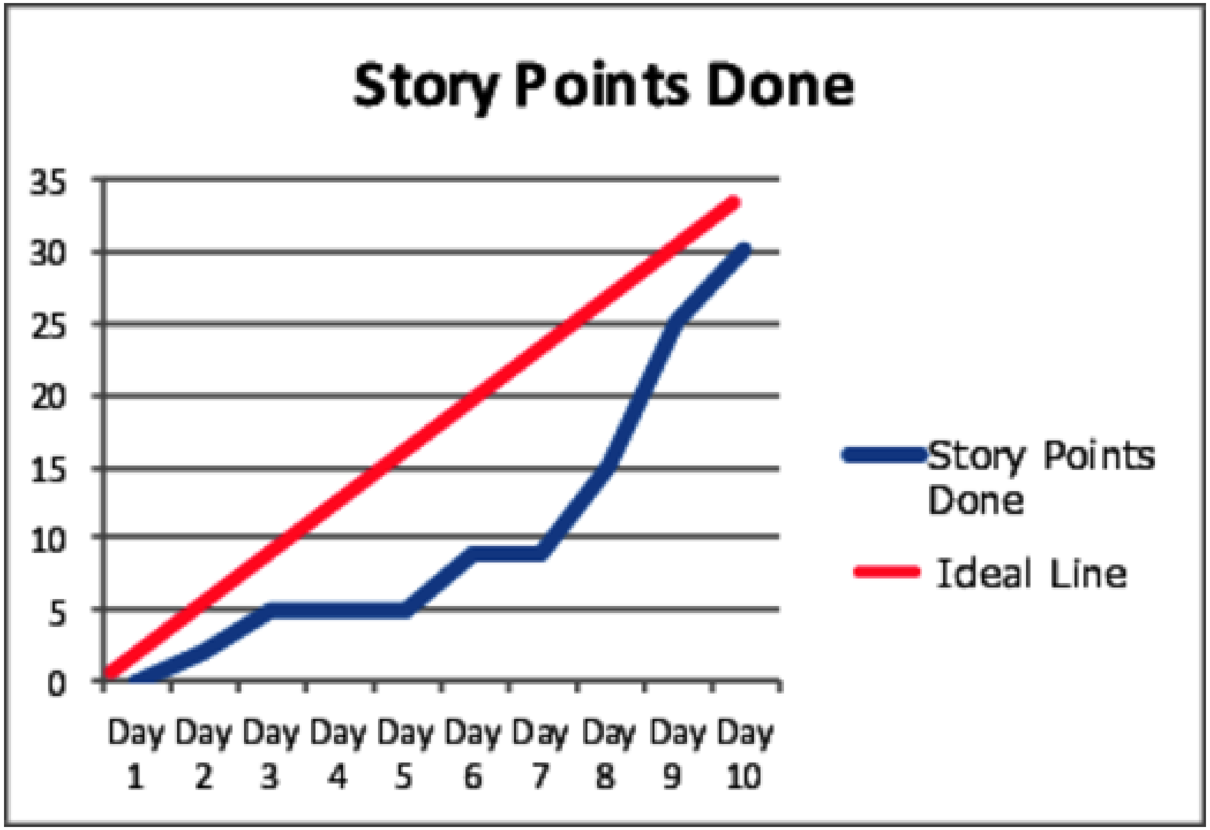
Monitor the Iteration Plan - Iteration Burndown Charts

- Shows work remaining over time
- The vertical axis shows the number of story points (idea days/hours, or stories)
- The horizontal axis shows days
- Useful when iteration length is 2-week or longer

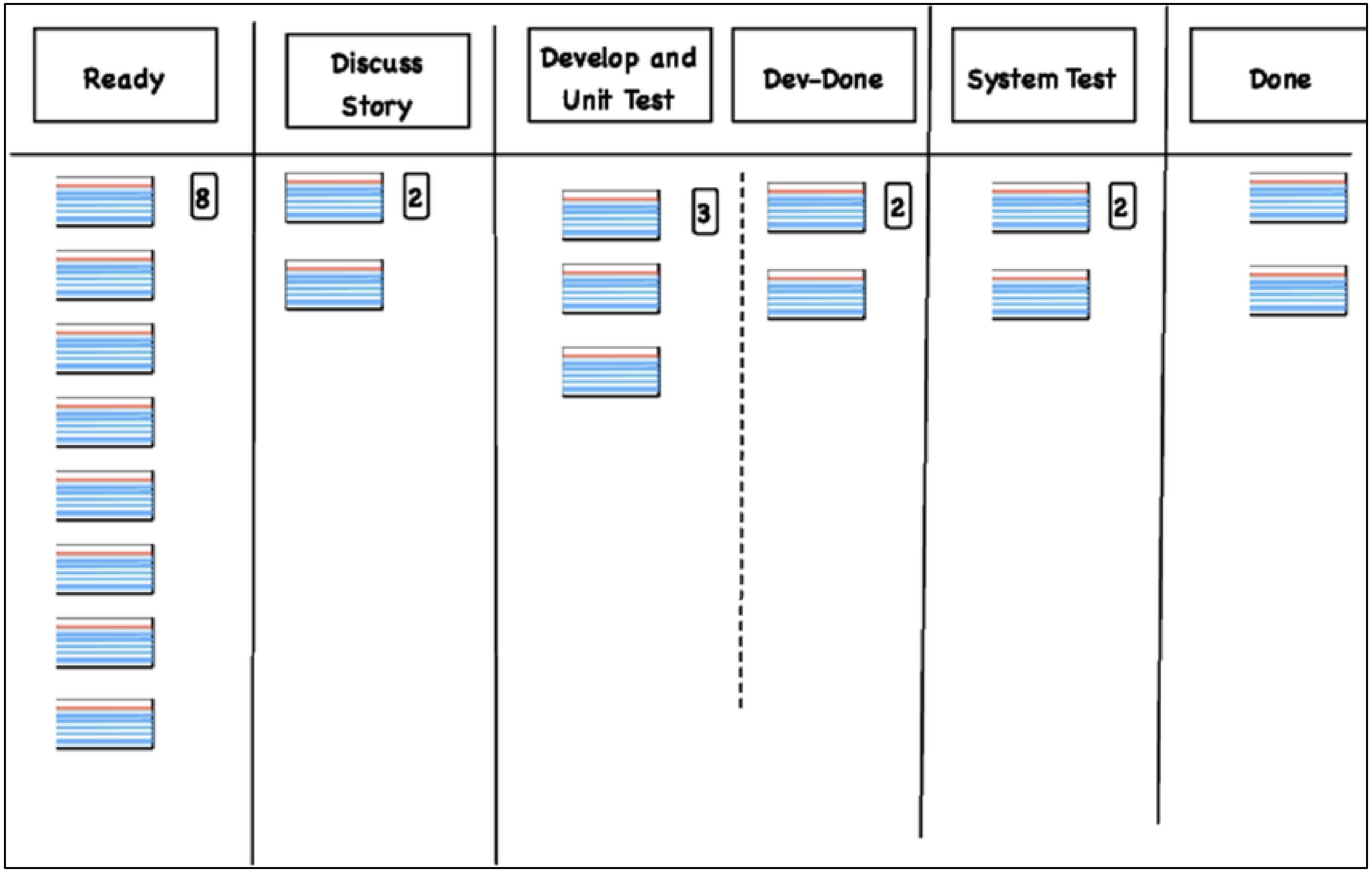


Monitor the Iteration Plan
- Iteration Burnup Chart

- Shows work done over time
- The vertical axis shows the number of story points (idea days/hours, or stories)
- The horizontal axis shows days
- Useful when iteration length is 2-week or longer



Monitor Flow – Kanban Board

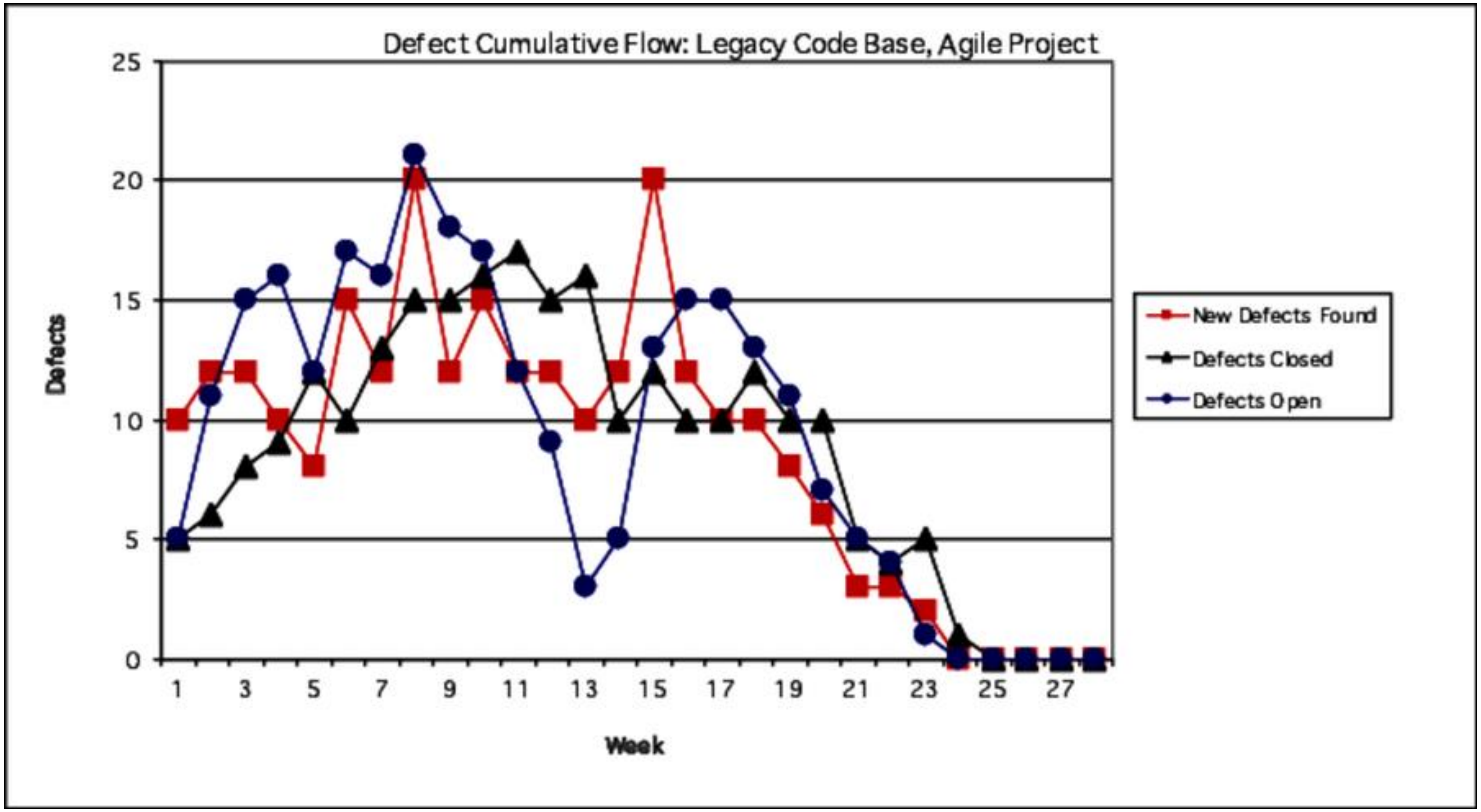


At this point WIP limit is reached at every stage, the team should move items from System Test to Done first before pulling more work.

Measurements

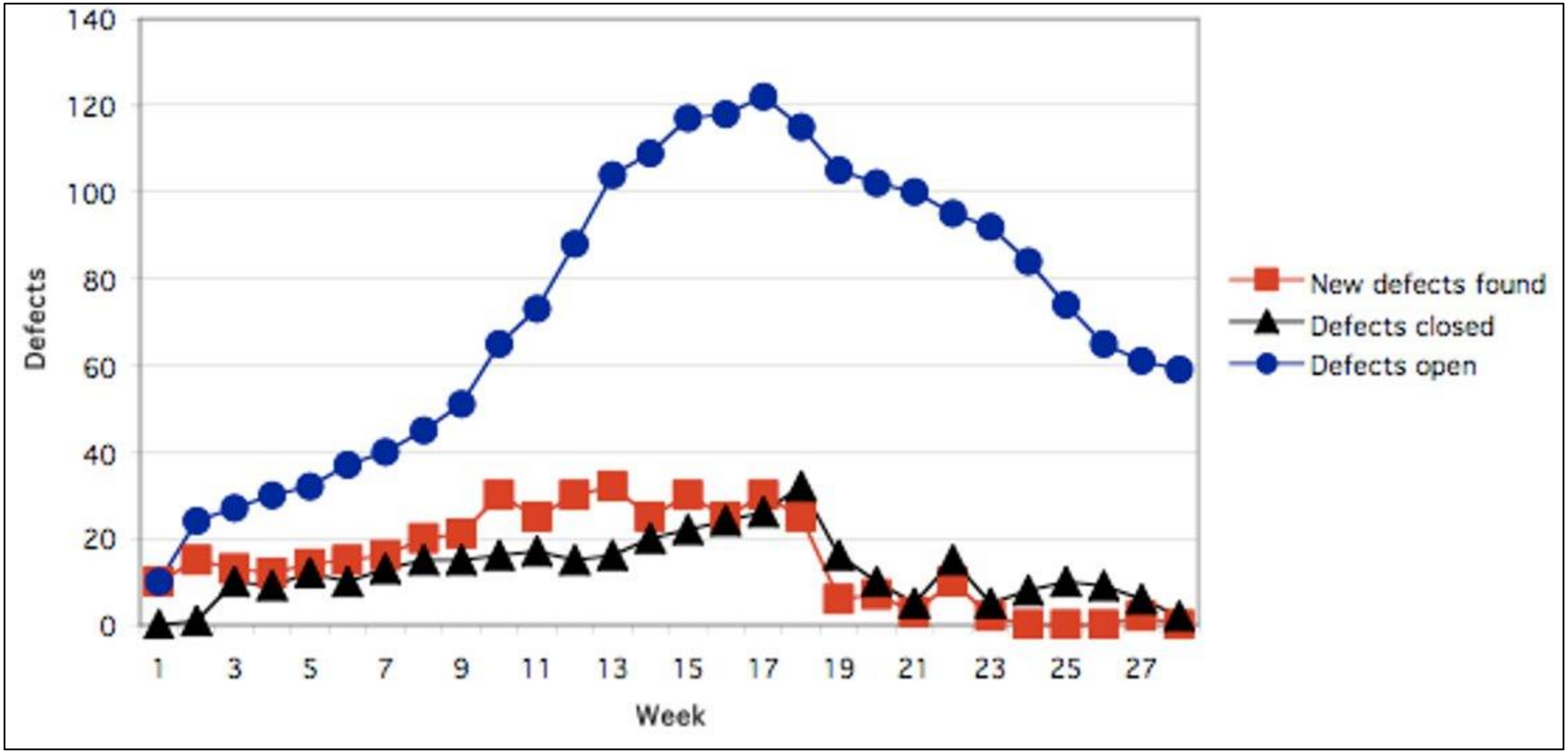
- Flow based agile approach
  - Cycle time
    - The duration of time from when a card is put in the first in-progress column until when it is marked as "done."
  - Lead time
    - The duration of time from when a card goes onto the backlog until it is released to the customer.
- Iteration based agile approach
  - Velocity
    - A team's velocity might vary under these circumstances, the team is learning how to work together or learning a new domain, the stories vary in size, the team changed the iteration duration, etc.
- Defect measurements
  - Defect escape rates
    - Number of defects that escape over time
    - Defect counts only start after the story has been accepted
  - Cost of fix a defect
    - e.g., Cycle time of defect-fixing
    - The later a defect is discovered, the more costly to fix it
  - Fault feedback ratio (FFR)
    - Rejected fixes/total fixes
  - Defect cumulative flow

Defect Cumulative Flow Example



The above shows the defects found, closed, and remaining as a line chart.
The above example shows that the team had debt from previous projects that it paid off in this project.

Defect Cumulative Flow Example - not a "Successful Agile Project"



Week 28 delivery with 60 open defects

Why Expected Progress is not being made?

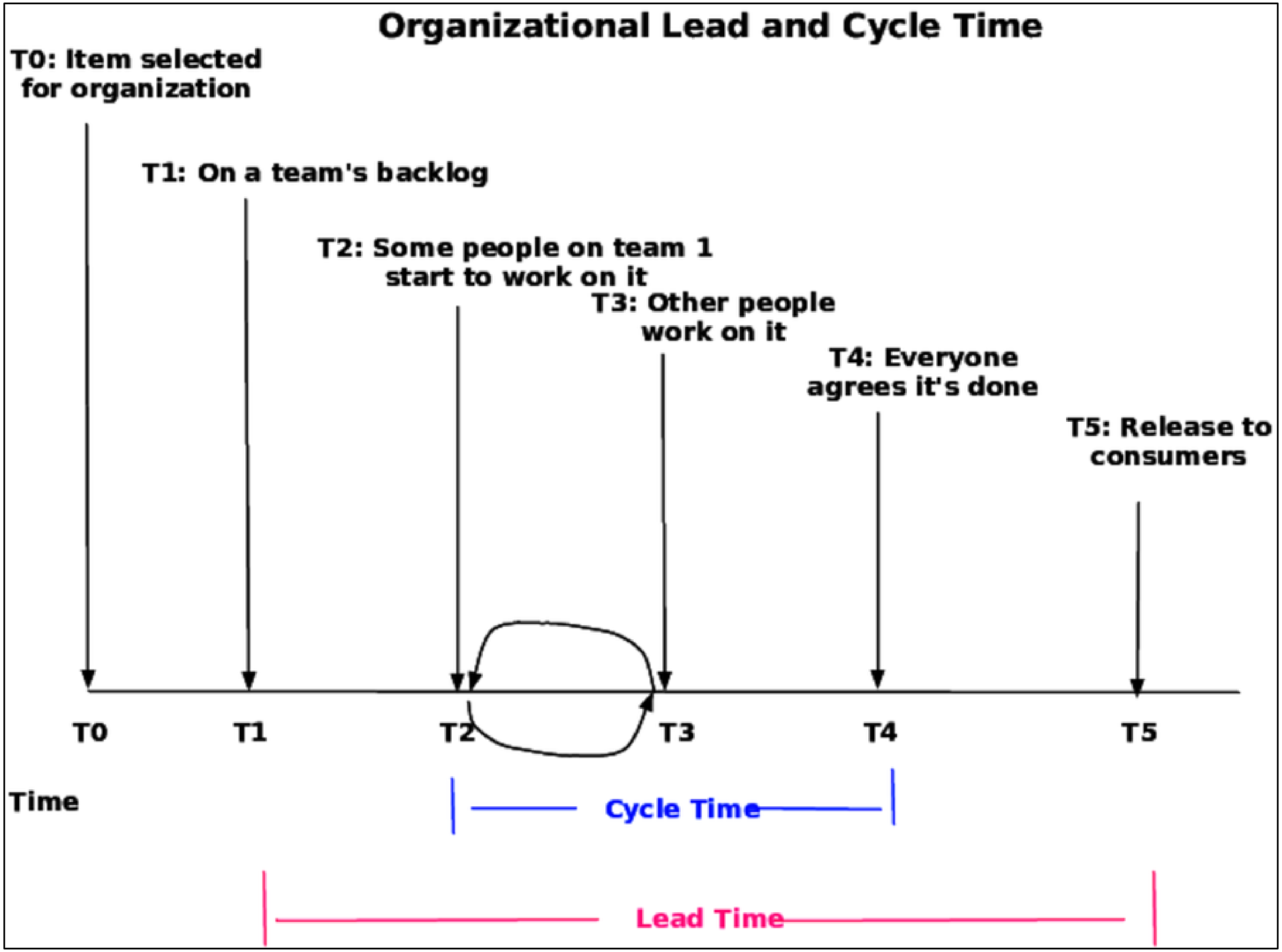
- The story was much more difficult that the team anticipated (and estimated)
- Everyone takes his or her own story at the start of the iteration
- Team members perform asynchronous code reviews… they have no collaborative commitment to review each other’s work
- No WIP limit
- Team members get interrupted by other work that the team did not anticipate
- Delays caused by various reasons
- …

Show Multitasking

Day	Number of Requests	Individual or Teams?	Notes
Day 1	2	Individual and team	Sandy for Project B and the team for support
Day 2	1	Individual	Sandy for Project B
Day 5	5	Individual	Each person for a different project
Day 6	2	Individual	Sandy for Project B
Day 8	1	Individual	Sandy for Project B

- The table shows the team got interrupted 5 out of 10 days for that iteration.
- Is Project B more valuable?
- Multitasking makes estimation impossible and reduces throughput.
- Teams are more apt to create defects when they multitask.

Locate & Visualize Delays


Organizational Lead and Cycle Time
T0: Item selected for organization
T1: On a team's backlog
T2: Some people on team 1 start to work on it
T3: Other people work on it
T4: Everyone agrees it's done
T5: Release to consumers
T0
T1
T2
T3
T4
T5
Time
Cycle Time
Lead Time

Earned Value Management

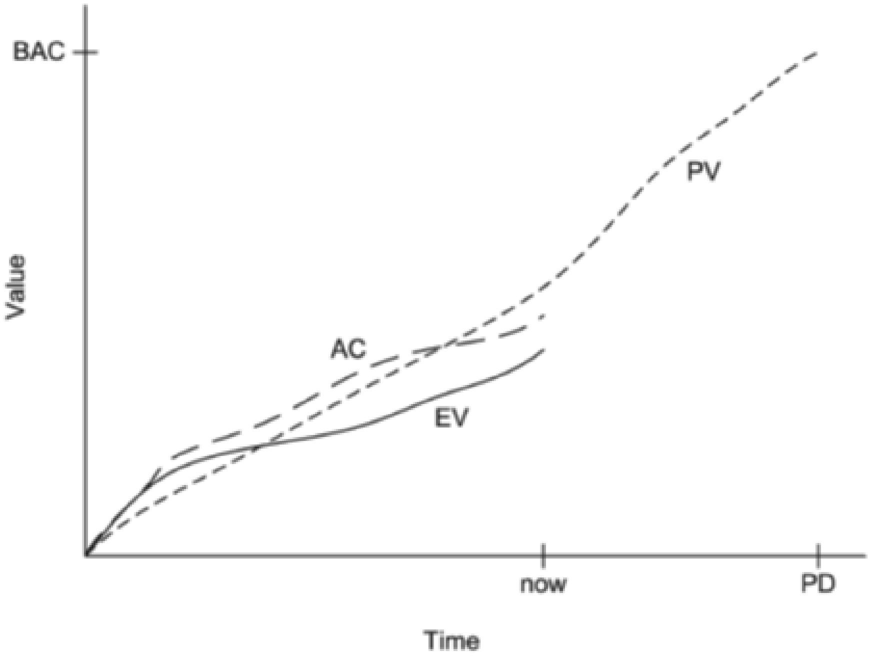
- Often used in traditional development approaches
- Assumption
  - For each task in the project effort and duration have been estimated, and task dependencies are known
- Concept
  - Use a quantifiable measure of progress (called the value)
  - Compare the earned value with the planned or actual value over time
- Possible measure of value
  - Effort of tasks/features (e.g., number of person-months)
  - Cost of tasks/features (e.g., dollar amount)
- The process
  - Before the project starts, use the project plan to calculate the planned value over time
  - While the project is underway, calculate the earned or actual value over time
  - Compare cumulative earned value against planned or actual value over time

Earned Value Management – Cont.

- Planned Duration (PD)
  - Total estimated time for the project
  - Can be calculated by using Gantt charts, CPM, etc.
- Budget at Completion (BAC)
  - Total estimated cost for the project
- Planned Value (PV)
  - Estimated cost of a task
- Earned Value (EV)
  - Planned value of a completed task
- Actual Cost (AC)
  - Cost actually expended on a completed task

Earned Value Management – Cont.

- Schedule Performance Index
  - SPI = EV / PV
  - If SPI == 1, on schedule
  - If SPI < 1, behind schedule
  - IF SPI>1, ahead of schedule
- Cost Performance Index
  - CPI = EV/AC
  - If CPI ==1, on budget
  - If CPI < 1, over budget
  - If CPI>1, under budget
- Forecast Project Duration
  - FPD = PD / SPI
  - E.g., PD=30, SPI=0.9, FPD=33.33
- Estimate At Completion
  - EAC = BAC / CPI
  - E.g., BAC=1140, CPI=0.95, EAC=1200



At the current time, the above project is behind schedule and over budget.

References

- [1] Create Your Successful Agile Project, Johanna Rothman, Pragmatic Programmers LLC, 2017. ISBN:9781680502602
- [2] Agile Estimating and Planning 1st Edition; Author: Mike Cohn; ISBN-13: 978-0131479418; ISBN-10: 9780131479418
- [3] Scrum TaskBoard, https://www.mountaingoatsoftware.com/agile/scrum/scrum-tools/task-boards