BROADCOM® | ValueOps™

AGILE ANALYTICS FOR DIGITAL TRANSFORMATION HANDBOOK

Mastering Agile Analytics to Unite the Enterprise

BROADCOM® | ValueOps™

Guides included in this handbook

1. How to interpret data from burnup and burndown
2. How to interpret data from cumulative flow diagram charts

BROADCOM® | ValueOps™

MASTERING AGILE ANALYTICS
TO UNITE THE ENTERPRISE

HOW TO INTERPRET DATA FROM BURNUP/ BURNDOWN CHARTS

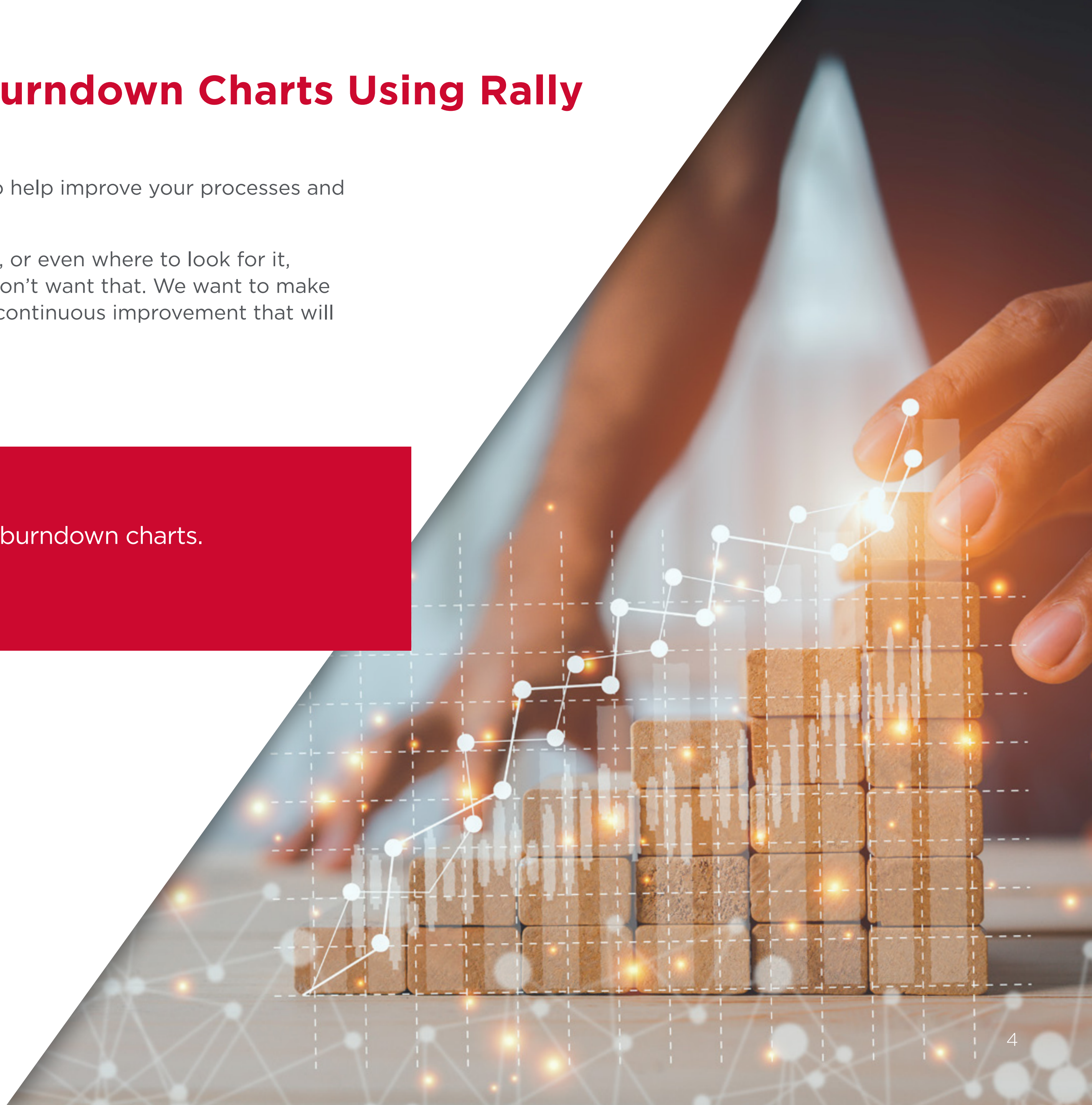
Demystify the Data That You Have Available in Rally and Learn How You Can Use Those Insights to Drive Continuous Improvement

How To Interpret Data from Burnup/Burndown Charts Using Rally

The great thing about Rally is that it produces a ton of data that you can use to help improve your processes and outcomes (among other things) while you're working.

The challenge, though, is that if you don't know how to make sense of the data, or even where to look for it, you're at risk of not making the most of your teams, processes, and Rally. We don't want that. We want to make sure that you're able to get the most out of everything and create a system of continuous improvement that will help you thrive.

The first of the series was on interpreting burnup and burndown charts.
You can watch that educational webinar here.



A Look at the Data Rally Can Produce

One of the things that can prevent people from getting deep into the data is the volume of charts and reports that Rally provides. We will divide that data into several categories. The first category being burndown and burnup charts. Here are just a handful of popular burndown and burnup graphs that are useful to identify patterns and correlations to improve on:

- Iteration Burndown / Burnup
- Tagged Story Burnup
- Portfolio Item Burnup
- Story Burndown / Burnup
- Release Burnup
- Milestone Burnup

Understanding classics such as the Iteration Burndown chart and Release Burnup chart will empower you to make use of any of the burndown and burnup charts.

Burndown Charts

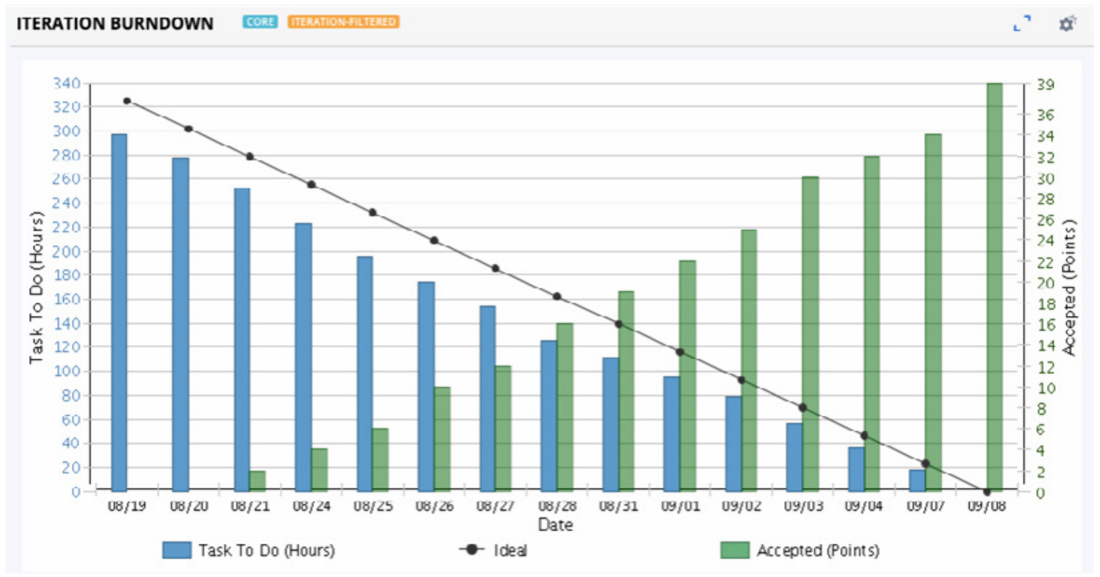
The iteration burndown app displays work remaining and completed in the iteration to proactively anticipate whether the committed work will be delivered by the iteration end date. They look like the example below.

The iteration burndown chart helps identify whether the team is on track to meet their commitments. This chart requires that the team is tasking out their stories and updating the remaining “To Do” hours on each task daily.

As you can see, each y-axis represents a different unit. The left y-axis is in hours and is used for both blue bars representing the Task To Do values and the black line representing the Ideal burndown rate. The right y-axis is the Accepted value in points which corresponds to the green bars.

As the delivery team members work on their Tasks, the Task To Do (Hours) are decreasing each day and remain below the Ideal line (black line). Even if a team is keeping their blue bars below the ideal line, if the green bars do not appear until later in the Iteration, that could mean the team is at risk of a QA bottleneck and late feedback.

What you want to see, is the green bars steadily increase as this means the team is getting work completed and accepted by the PO.

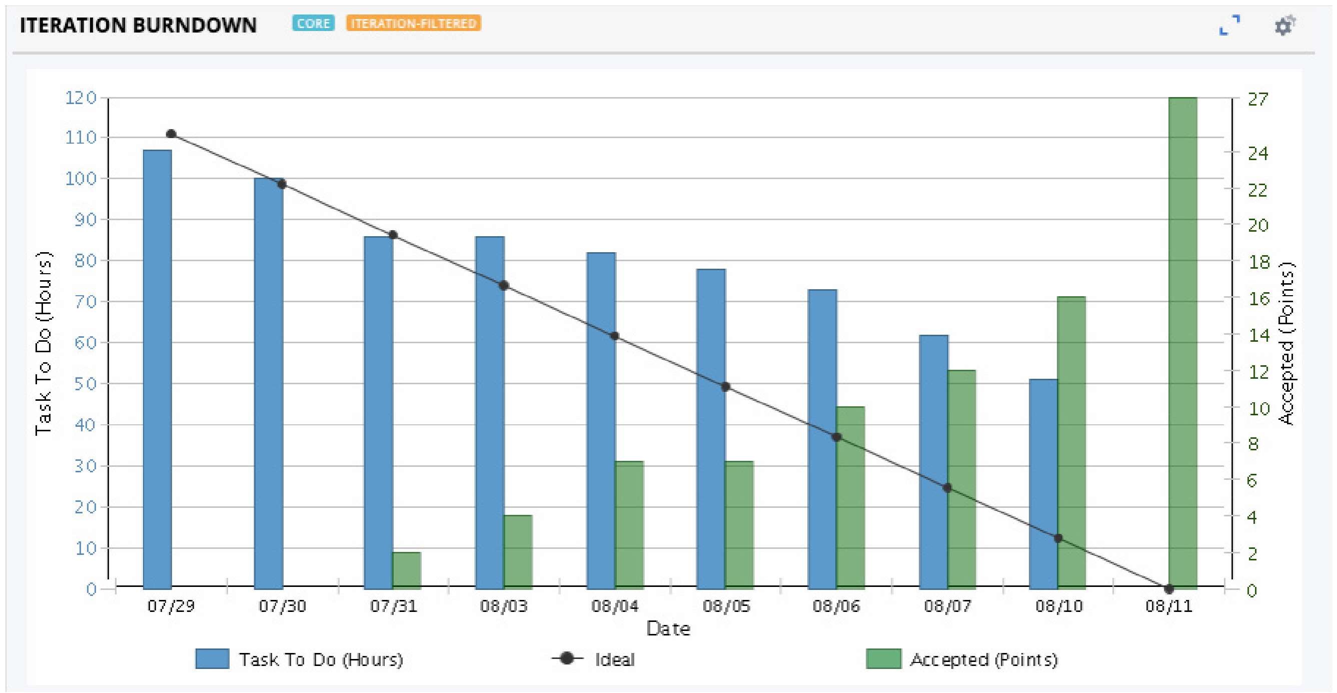


Burndown Examples

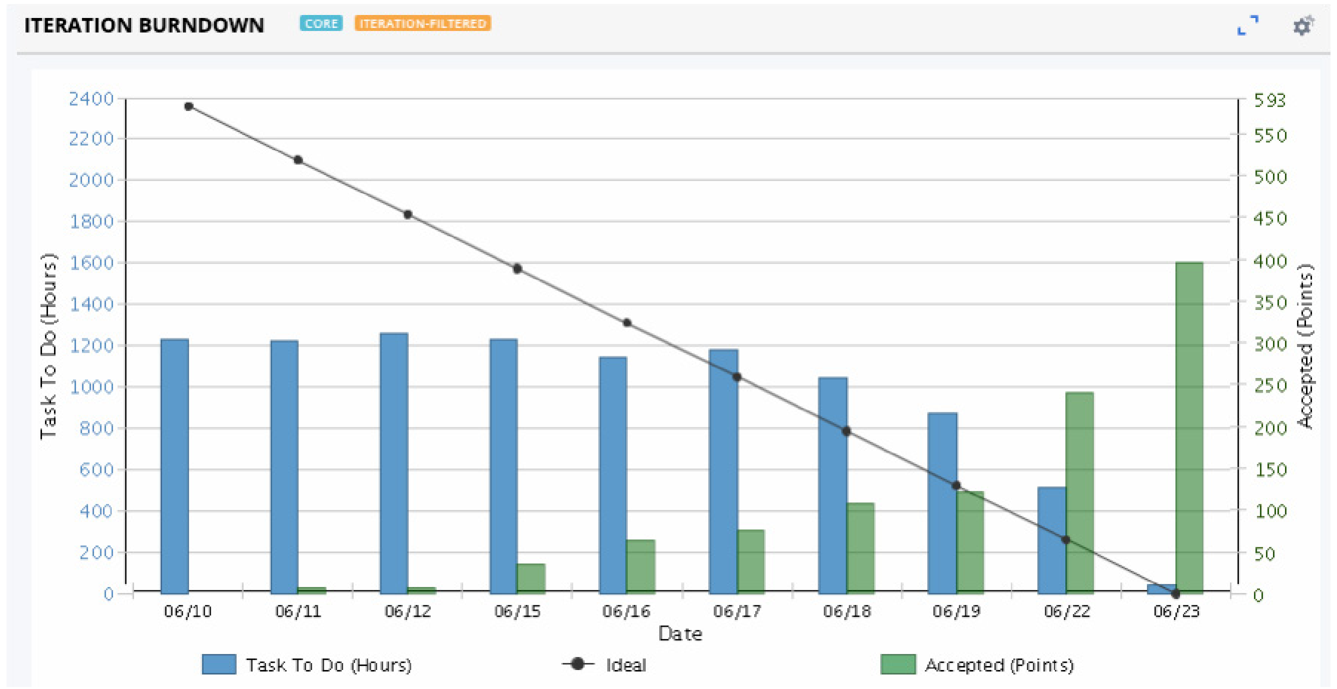
There is only so much learning that can happen from a "happy path" example. To help, we've got some real-life examples of burndown chart scenarios. We'll walk you through what you're looking at and why the chart looks the way it does.

Example #1

In this example, you can see that the blue bars, the task to dos, aren't moving all that much during the middle of the iteration. You can see that work is being accepted early and throughout the iteration because of the rising green bars. One possible explanation for the minimum daily moment in the blue bars during the middle of the iteration, is that the team is new to Rally and is forgetting to update their tasks at the end of the day. If you look closely, you can see many of the To Do task hours drop off on the last day. This could further support the theory that they simply weren't updating the tasks throughout the iteration and made most of the updates to the tasks on the last day. However if that was the case, we probably wouldn't see much growth in the green bars since the tasks have to be completed for the Stories and Defects to be accepted. More likely, the team overcommitted and removed many of the Stories and Defects on the last day which would explain the sudden drop in blue bar on the last day. In either case, ideally the team would have noticed this during the iteration and made the necessary adjustments.



Burndown Examples (cont')



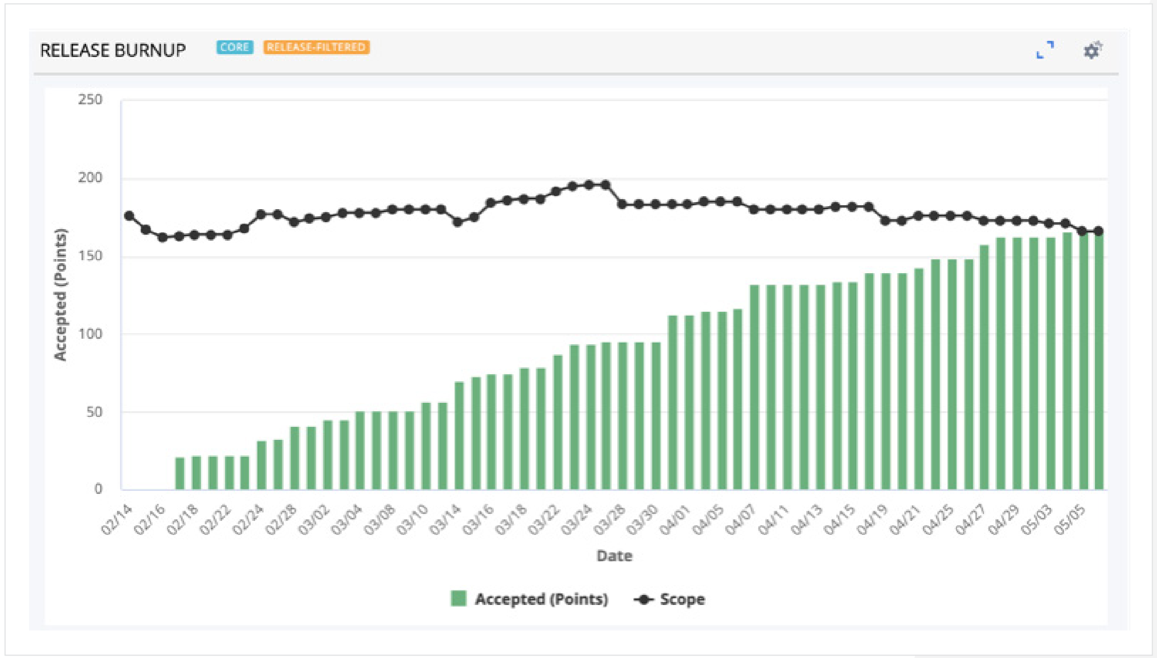
Example #2

The first thing that jumps out when looking at this chart is the large gap between the ideal burndown value (black dot) and the to do value (blue bar) on the first day. Typically, this gap (if any) is usually small on the first day since the team hasn't had time yet to work on the tasks. A large gap like this is seen on a team that is adding tasks (or task hours) throughout the iteration. Adding tasks during the iteration will cause the ideal line to recalculate to accommodate for the new task hours. But adding work throughout the iteration will not affect the blue bars on days prior to the work being added hence the creation of a large gap. This particular chart is for a Scrum of Scrums (or collection of teams working together with a shared iteration cadence) so the values in the y-axis are expectedly large. Another pattern supporting the hypothesis that the teams are adding work throughout the iteration is the fact the blue bars are remaining fairly flat. This would happen if each day the teams are adding a similar number of task hours as they are completing for that day. As for the accepted work (green bars), it is great to see some work being accepted early in the iteration, however, on the last day, it appears as approx. 400 of the 590 points were accepted which means the team didn't meet their commitment. This set of teams can use this information to help improve planning for the next iteration. Improving planning leads to greater predictability and removes waste.

Watch the webinar to learn more related to burndown and burnup charts.

Burnup Charts

The Release Burnup app shows you the work delivered so far in the release to proactively anticipate whether the release scope will be delivered. The chart helps teams identify whether they are on track to meet their Release commitments.



On a burnup chart, the black line represents the total number of Story and Defect points that are scheduled in the Release on each day and changes in the black line (total work points) indicate scope change. We expect some scope change throughout the Release, however, large changes could indicate the team(s) didn't finish planning the Release and/or there was a change in priorities.

The y-axis unit is points. Teams need to size their work to take make use of this chart.

Ideally, team members swarm on the highest priority work and get work completed and accepted early in the Release. This is evident by seeing green bars (representing accepted work) early in the Release. Throughout the Release, we want to see the green bars steadily increase as this indicates the teams are getting work completed and accepted regularly throughout the release and getting the feedback loops started for each item

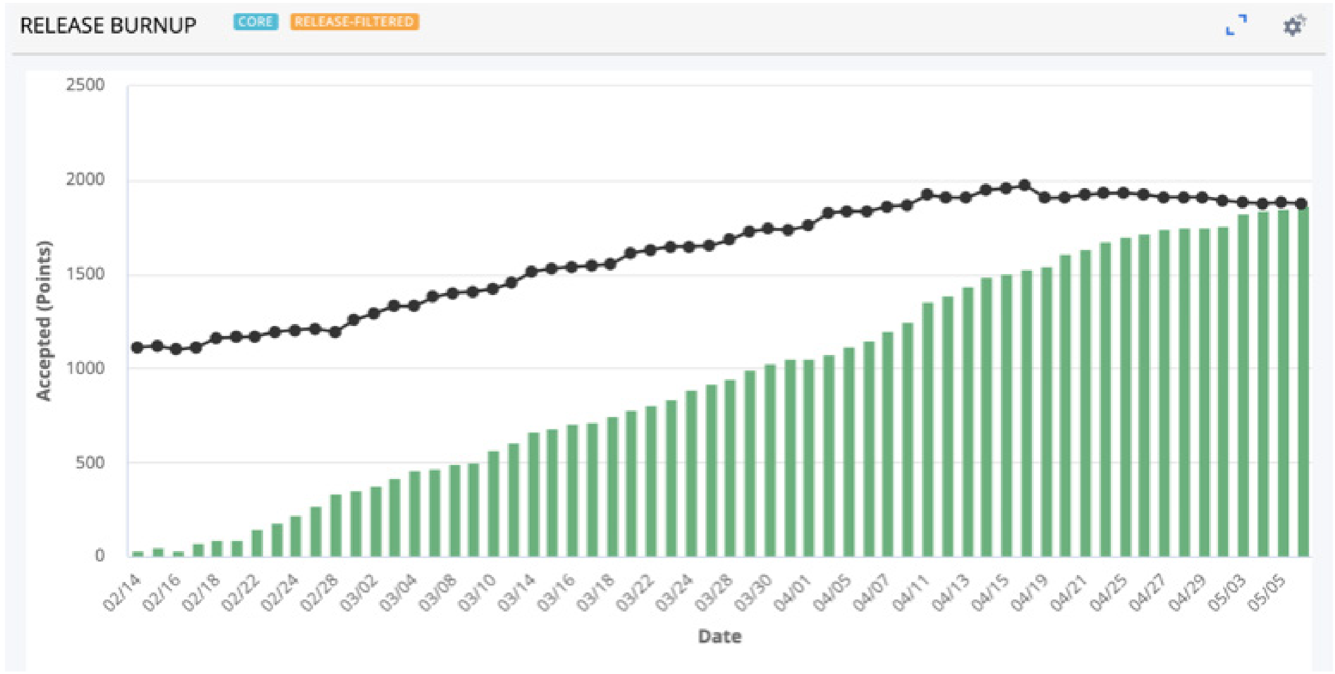
Ideally, on the last day, the scheduled Work Items (black line) and Accepted Work Items (green bars) converge. A gap in the two values on the last day indicates work that wasn't accepted during the Release.

Burnup Chart Examples

As with above, the best way to truly understand the data is by looking at specific, real-world examples of burnup charts. This way you can see exactly what's happening on the chart compared to what's happening with your team.

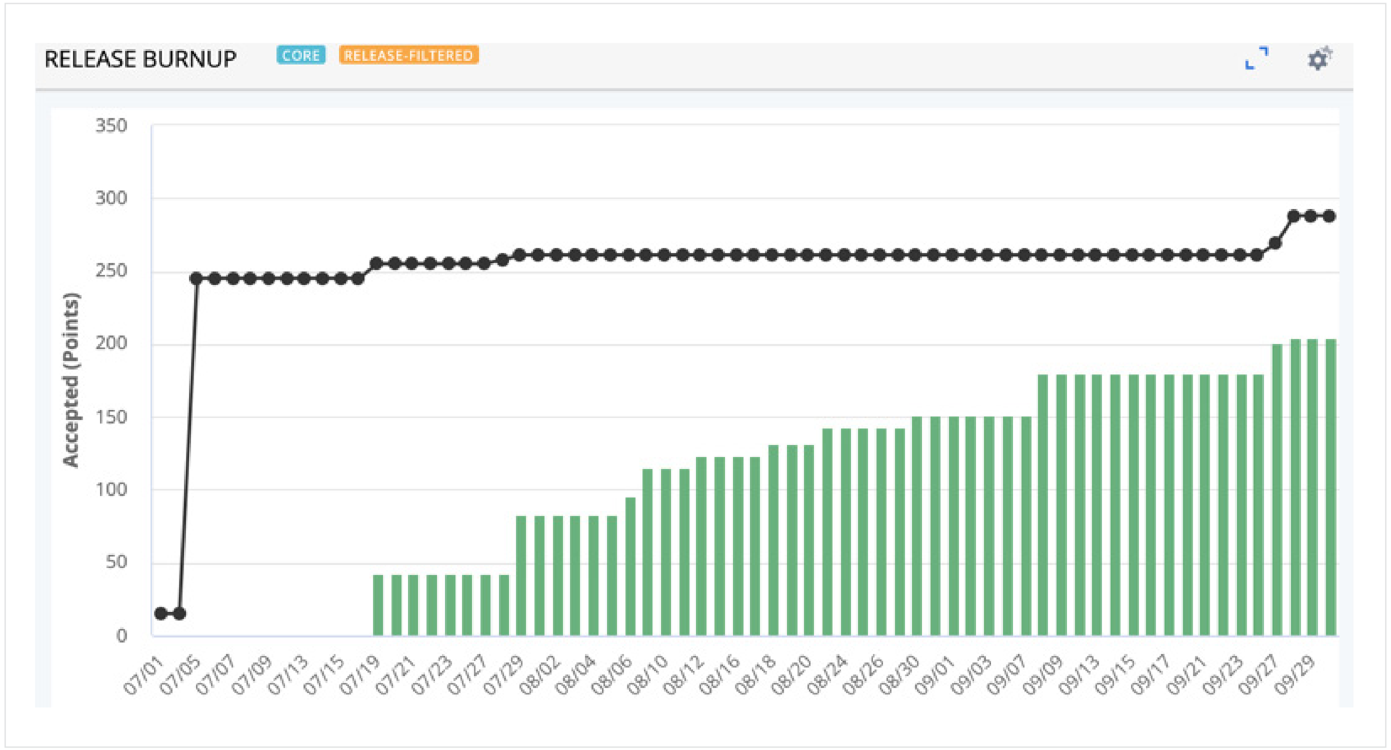
Example #1

Here you can see the scope is steadily increasing throughout most of the release AND all work is accepted by the end of the release. The increasing scope indicates the team members are adding more work to the release as they finish work. This happens often when the features in the release haven't been fully decomposed at the start of the release and the child stories get created during the release. Towards the end of the timebox, they stopped adding work, as you can see by the black line flattening out and the team focused on finishing what was in the plan. You can predict how much work the team is going to be doing based on how consistent the growth of the green bars is which makes planning easier. These are often patterns we see on mature teams. For teams like this that haven't fully decomposed their features into stories at the start of the release, there is great value in using a feature release burnup chart to track scope change on the features as well as progress on the features.



BROADCOM® | ValueOps™

Burnup Chart Examples (cont')



Example #2

This is a less than ideal burnup chart. For starters, there's no work being completed or even entered into the system for the first few days, whether that was because they were busy wrapping up the work from the last iteration or if they were planning, it's not clear. But, as you can see by the black line, they planned for a lot of work to be done during the release and they added to the scope a couple times including towards the end of the release. This is odd in this situation because it was evident they were not on track to finish what was already planned.

This team would have benefited from having reviews throughout the release to steer the contents. Additionally, they can use this information to improve planning for the next release.

Hopefully this team plans less than 200 points for next release. You'll get more work done and increase predictability if you plan closer to what the team is capable of, rather than pushing everyone too hard and asking them to over commit.

Want to Learn More?

If you'd like a deeper dive into burnup/burndown charts, we've got a webinar that expands on this topic and covers more real-world examples. You can find it **here.**

To better understand the data in Rally beyond burnup and burndown charts, you can take part in our extended webinar series to unlock excellence with agile analytics. The purpose is to improve planning, tracking, and forecasting by demystifying common agile metrics and providing practical knowledge so that teams are empowered to identify action for continuous improvement. You can tune into the series found on BrightTALK under Rally Software.

01

How to Interpret Data for Burnup / Burndown Charts

July 26 @ 1:00 pm ET

02

How to Interpret Data from Cumulative Flow Diagram (CFD) Charts

September 27 @ 1:00 pm ET

03

How to Interpret Data for Throughput and Cycle Time

October 4 @ 1:00 pm ET

04

Advanced Material: Rally Insights Framework

Registration will open soon on BrightTALK

05

Advanced Material: Smart Cumulative Flow Diagram and Cycle Time

Registration will open soon on BrightTALK

06

Dashboarding Roles

Registration will open soon on BrightTALK

07

Dashboarding Rituals

Registration will open soon on BrightTALK

08

Preparing for Big Room Planning (BRP) / Program Increment (PI) Planning with Todd Galloway, RTE for Rally Software

Registration will open soon on BrightTALK

BROADCOM® | ValueOps™

MASTERING AGILE ANALYTICS
TO UNITE THE ENTERPRISE

HOW TO INTERPRET DATA FROM CUMULATIVE FLOW DIAGRAM CHARTS

Learn how to get the most of the data for traceability and visibility across the development lifecycle

How to Interpret Data from Cumulative Flow Diagram Charts

You could have all the data in the world, but if you don't know how to read the charts and interpret what you're looking at, the data is not helpful.

But data is essential to inspire commitment. Rally collects, aggregates and rolls up data in real time from your global teams, empowering enterprises to optimize their business processes, deliver on strategy with clear and decisive commitments, and to quickly adapt to capitalize on new opportunities..

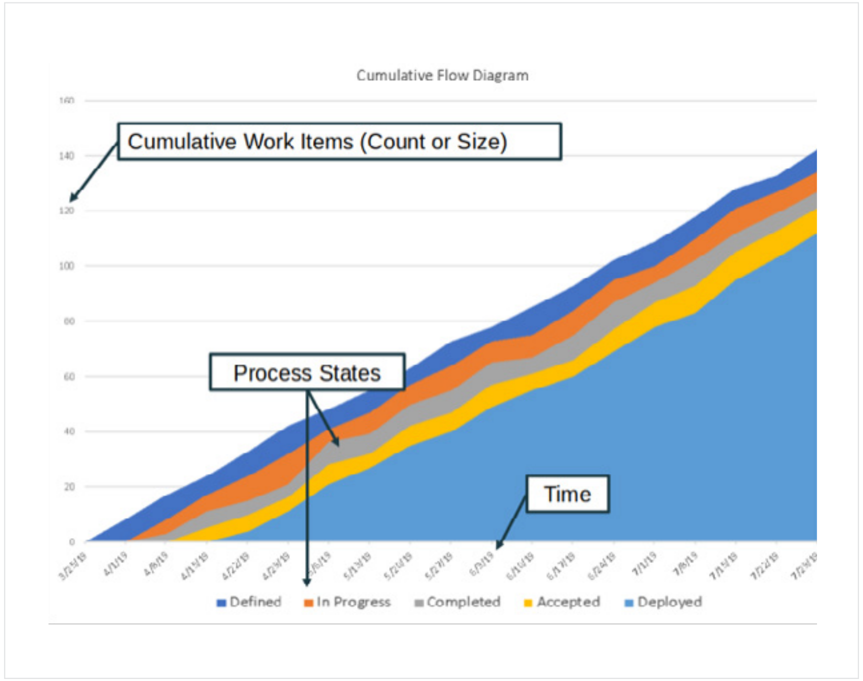
One example of the importance of aggregation and rolling up data is through the Cumulative Flow diagram (CFD) charts. The CFD provides a concise visualization of the three most important metrics of your flow: cycle time, throughput, and work in progress. When your CFD reflects concise data from your global teams, you can plan the ideal amount of work that your teams can handle in a sprint.

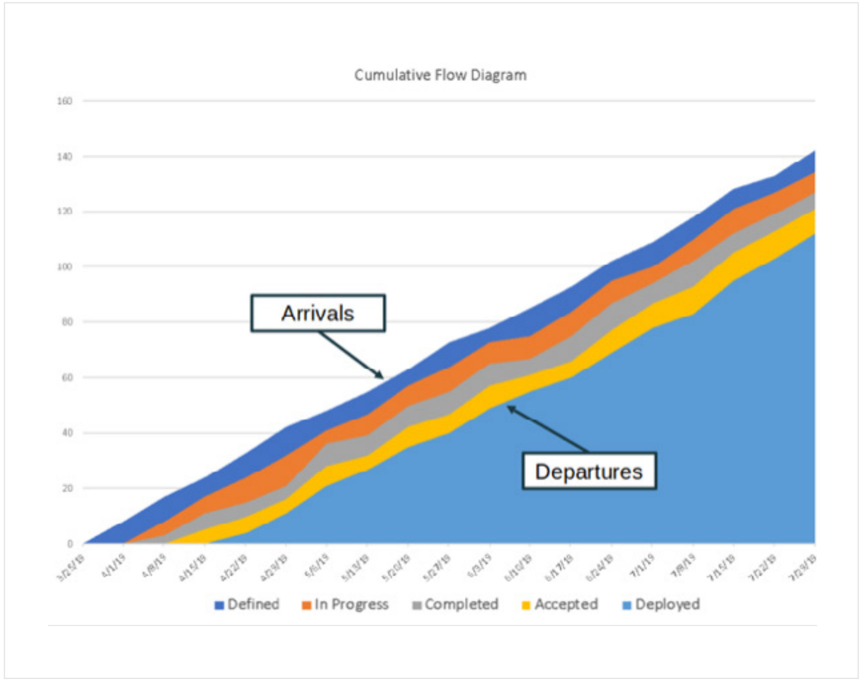
What Is a Cumulative Flow Diagram Chart?

The Cumulative Flow Diagram (CFD) Chart is one of the most important Agile charts. It can tell the team information about their flow process, bottlenecks, arrival and departure rates, throughput, and cycle time.

The y-axis unit is points and represents the total number of Story and Defect points that are scheduled in the Iteration on each day. Teams need to size their work to take advantage of this chart. The X-axis represents the amount of time that's progressed. This could be days, weeks, or months, depending on the kind of diagram you're using. The overall height of the area chart indicates the total scope. Ideally, throughout the Iteration, the scope should not change which would appear as a flat line.

The different colors represent the different Schedule State values.





Ideally, team members swarm on the highest priority work and get work Completed and Accepted early in the Iteration. This is evident by seeing the green area early in the Iteration and seeing the slope of the green area increase steadily throughout the Iteration.

Healthy Agile teams focus on limiting the amount of work In-Progress (WIP) which is represented by the yellow band. Reducing WIP fosters collaboration and focuses the team on the highest priority work getting done early in the Iteration.

Healthy Agile teams have active POs that can Accept the work shortly after it has been Completed by the delivery team members. This is represented by the blue area being a very thin band.

Let's Explore the Examples

Rally provides multiple different kinds of CFDs you can use. They are:

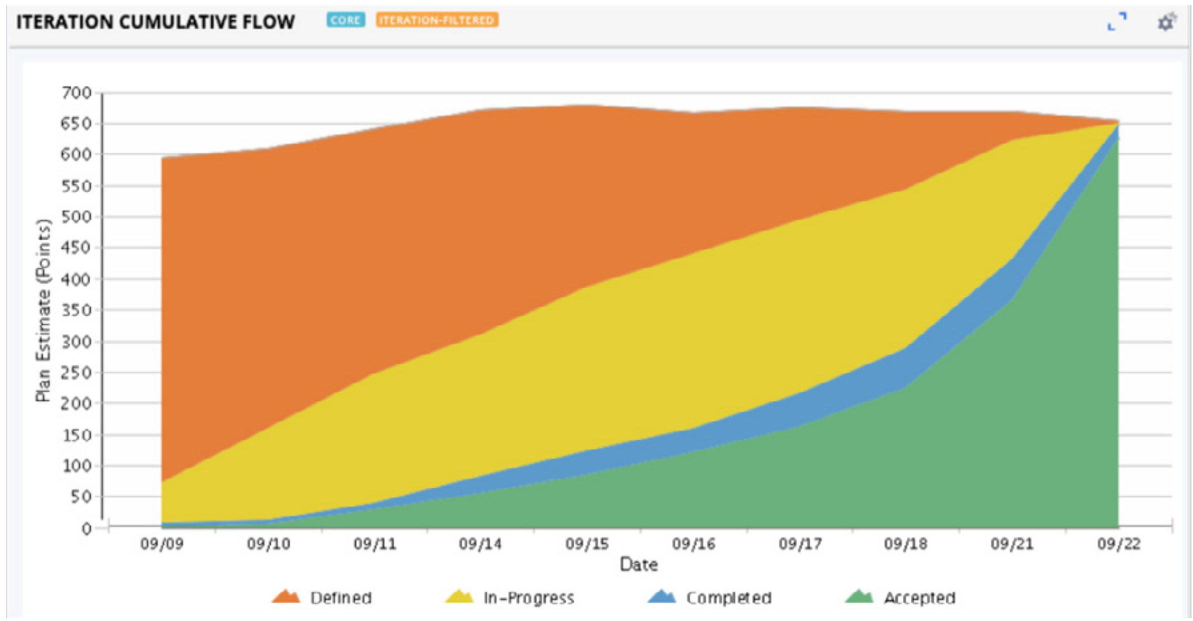
- Iteration CFD
- Release CFD
- Milestone CFD
- Story CFD
- Project CFD
- Portfolio Item CFD
- Team Board — Smart CFD
- And Several custom Apps

For this ebook, we're going to be focused on Iteration CFDs. The Iteration Cumulative Flow Diagram displays the rolled-up states of all scheduled items to help you plan and track your iterations. You can find this chart on the Iteration Status page, on the Reports page, and in the App Catalog.

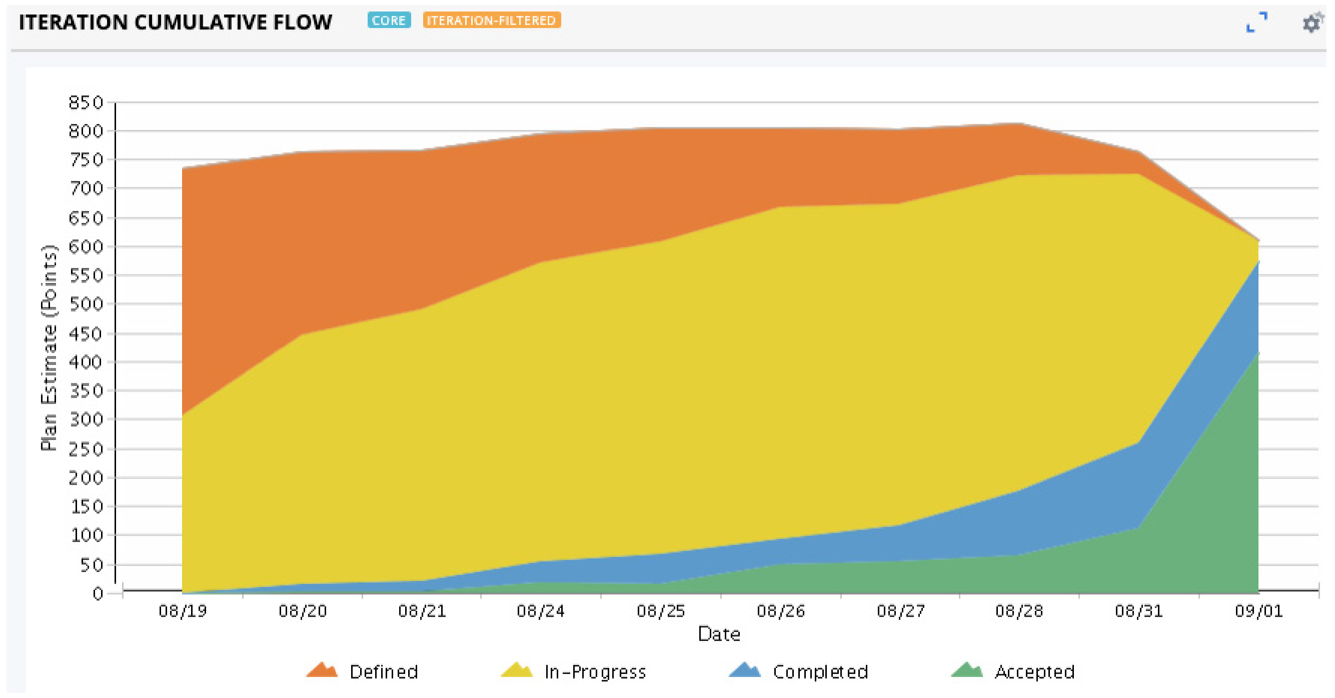
To help ensure that you fully understand what's going on with these charts, we've gathered up some examples of different CFDs. This way you can get a sense of what you're seeing, why you're seeing it, and what it means about the way your team is working.

Example #1

This first diagram is a scrum of scrums. There isn't much going on with this one. It's pretty much what you'd hope to see in a situation like this. Everything converges at the end, which is great. And, everything is green on the last day. Ideally, if you're coordinating a scrum of scrums, this is exactly what you'd want your CFD to look like.



Cumulative Charts Examples (cont'd)



Example #2

The first thing you should notice here is that planned work is disappearing at the end, which shouldn't really happen. What's likely happening here is that the team is probably over-committed. They bump the work that isn't done over to the next iteration, which is kind of why things look odd at the end. This can lead to a bottleneck because everything is wrapping up at the end.

Another thing to notice this that work isn't being accepted as much as it probably should be, you can tell by the larger blue bar. That should be bumped over into the green where accepted work shows up.

We can also see that it looks like they added more work on the second to last day, which is unusual since they're clearly planning more than they can write, so everything is just getting bumped.

You can use the y-axis to understand how much work you can reasonably accomplish during a sprint.

Cumulative Charts Examples (cont'd)

Example #3

There's quite a lot going on here. The first thing to notice is that work isn't being completed. It's not even being moved from in-progress to done. It's just sitting there.

On top of that, they're adding more work and putting it into the in-progress section. You can even see that work is being added on the last day.

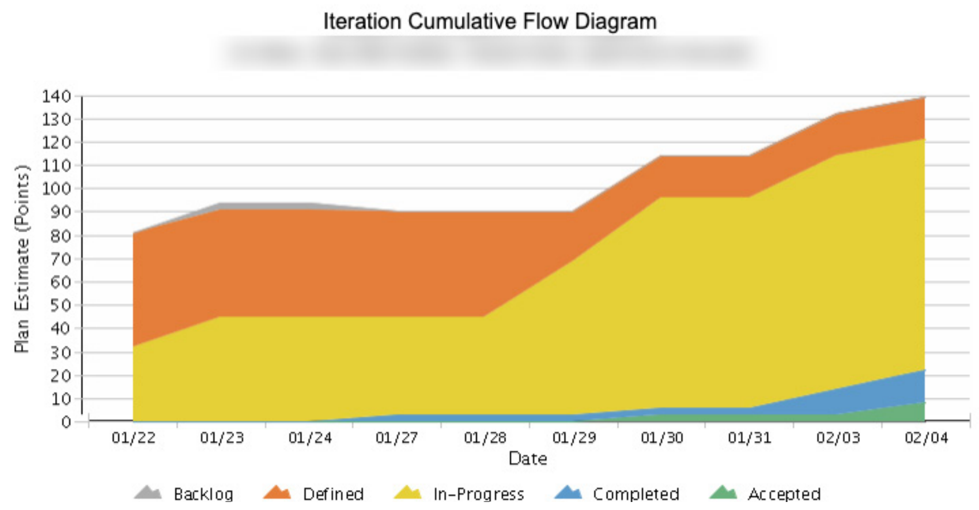
This chart is likely individuals working individually.

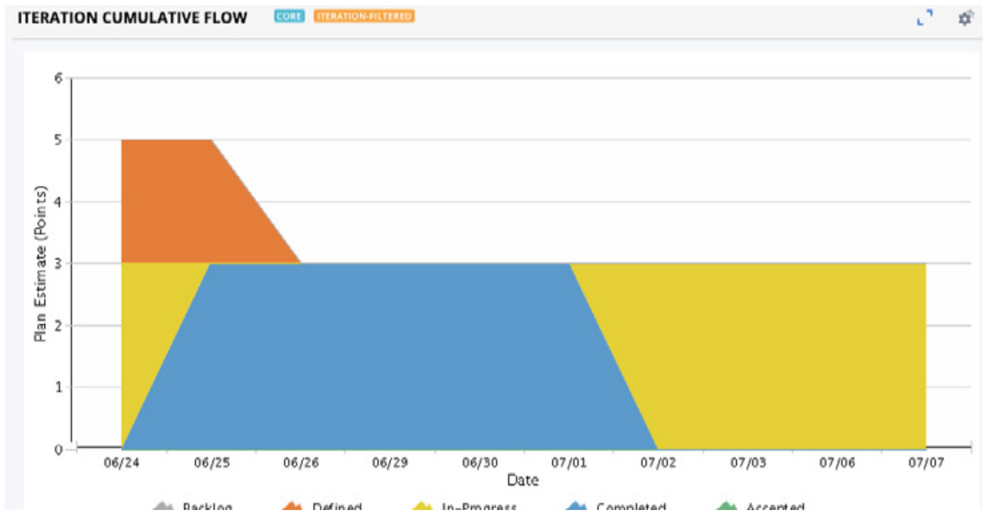
It's not the healthiest-looking CFD, but it gives you an idea of what's going on with your teams and can be a great way to start a conversation about improving. Asking questions can help you get a sense of what's going on.

Example #4

This is a very interesting, but not a super useful graph. Could be a data hygiene issue affecting the graph.

Talk to the team and figure out what's going on.





Cumulative Charts Examples (cont'd)

Example #5

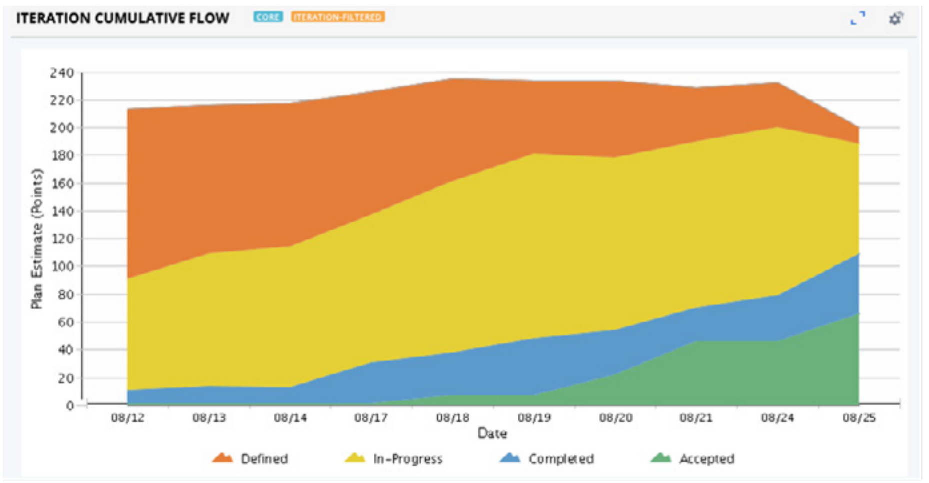
There's too much work happening during this iteration. You can see they're removing things on the last day, there's a lot of work that's still in progress, and not nearly enough work is being completed.

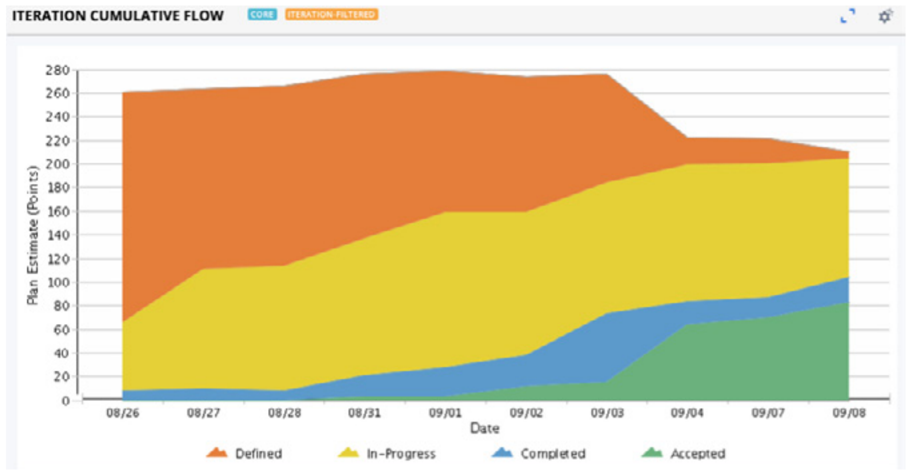
Ask yourself this question about this chart:

Given the Iteration Cumulative Flow diagram to the left, approximately how much work should this team plan for the next iteration?

Should they continue to plan approximately 200 points of work? If not, why?

This next chart is the same group as above, but a different chart.





This next chart is the same group as above, but a different chart. As you can see, they're not learning their lessons. They're overplanning over multiple iterations. This team (or teams) continues to plan more work than they have historically been able to complete. Doing this leads to several unhealthy behaviors and slows down the team.
They would be better off only planning approx. 60 – 80 points of work and focusing on reducing their WiP and meeting their commitments.

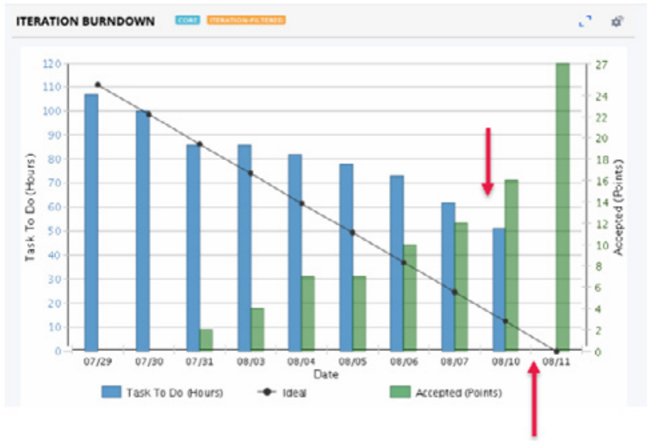
Ask why it's happening and make adjustments based on the answers you're getting. Focus on in-progress work first, then add more.

Cumulative Charts Examples (cont’d)

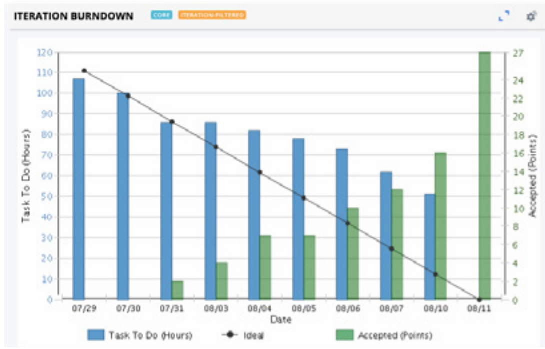
Example #6

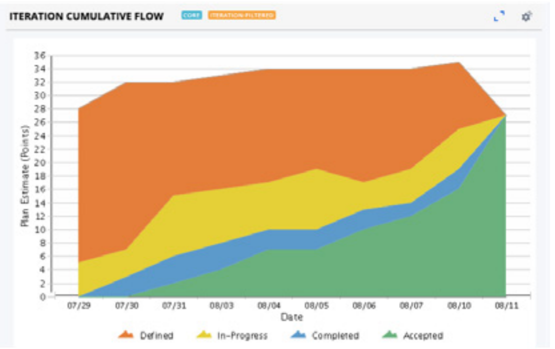
Using CFD with Burndown

What happened on the last day? Without more context, this can be a nearly impossible question to answer. But, when you add in a CFD, you get a better idea of what’s happening with the team.



When you combine the two charts, you can see what’s going on. They removed a bunch of work on the last day, work they weren’t going to get done.

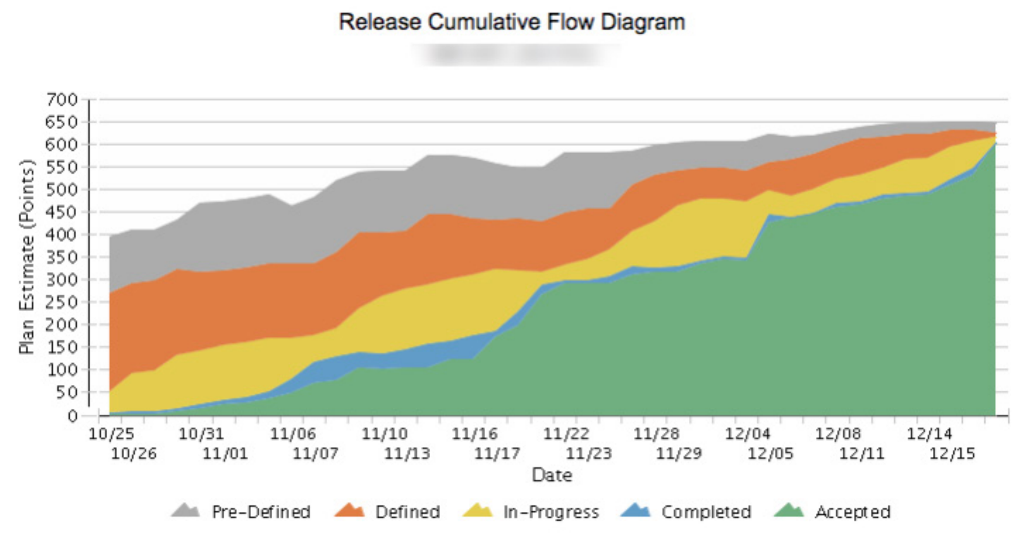




Release CFD

The release CFD shows a quarter’s worth of work. You can see what’s happening from one iteration to the next. There is scope change, but that’s okay and is pretty normal.

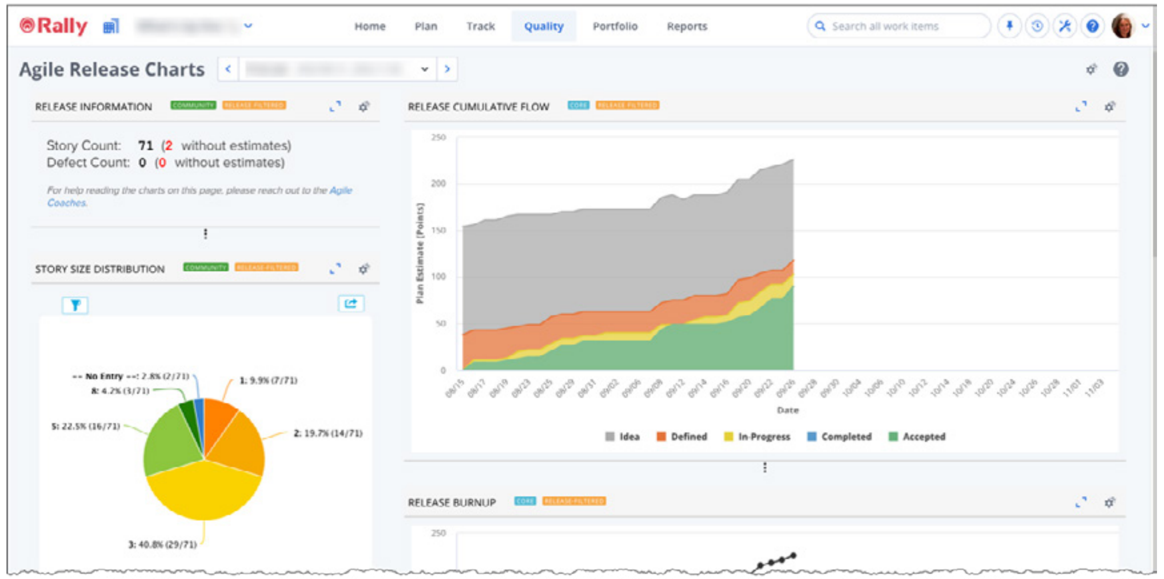
All in all, this is a pretty healthy-looking quarter. Some stair stepping at the end of each iteration, but nothing too bad.



Adding Information to Help Interpret Charts

As you can see, the more information you have, the better you can understand these charts and the information that they're showing you.

One way to ensure that you've got the full picture is to create a dashboard that provides the context you sometimes need to read the charts. Include stories in the dashboard, as they can help you better understand the scope of work being done. All the extra information helps you see the full picture with your CFDs.



Story Count: 71 (2 without estimates)
Defect Count: 0 (0 without estimates)

For help reading the charts on this page, please reach out to the Agile Coaches.

Want to Learn More?

We covered cumulative flow diagrams in a recent webinar. To see the full story and learn more about CFDs, **check it out today.**