Rally®
by Broadcom

AGILE ANALYTICS FOR DIGITAL TRANSFORMATION HANDBOOK

Mastering Agile Analytics to
Unite the Enterprise

Rally®
by Broadcom

Guides included in this handbook

1. How to interpret data from burnup and burndown charts
2. How to interpret data from cumulative flow diagram charts
3. Using data to improve and prepare for Big Room Planning and mastering capacity planning

Rally®
by Broadcom

MASTERING AGILE ANALYTICS
TO UNITE THE ENTERPRISE

HOW TO INTERPRET DATA FROM BURNUP/ BURNDOWN CHARTS

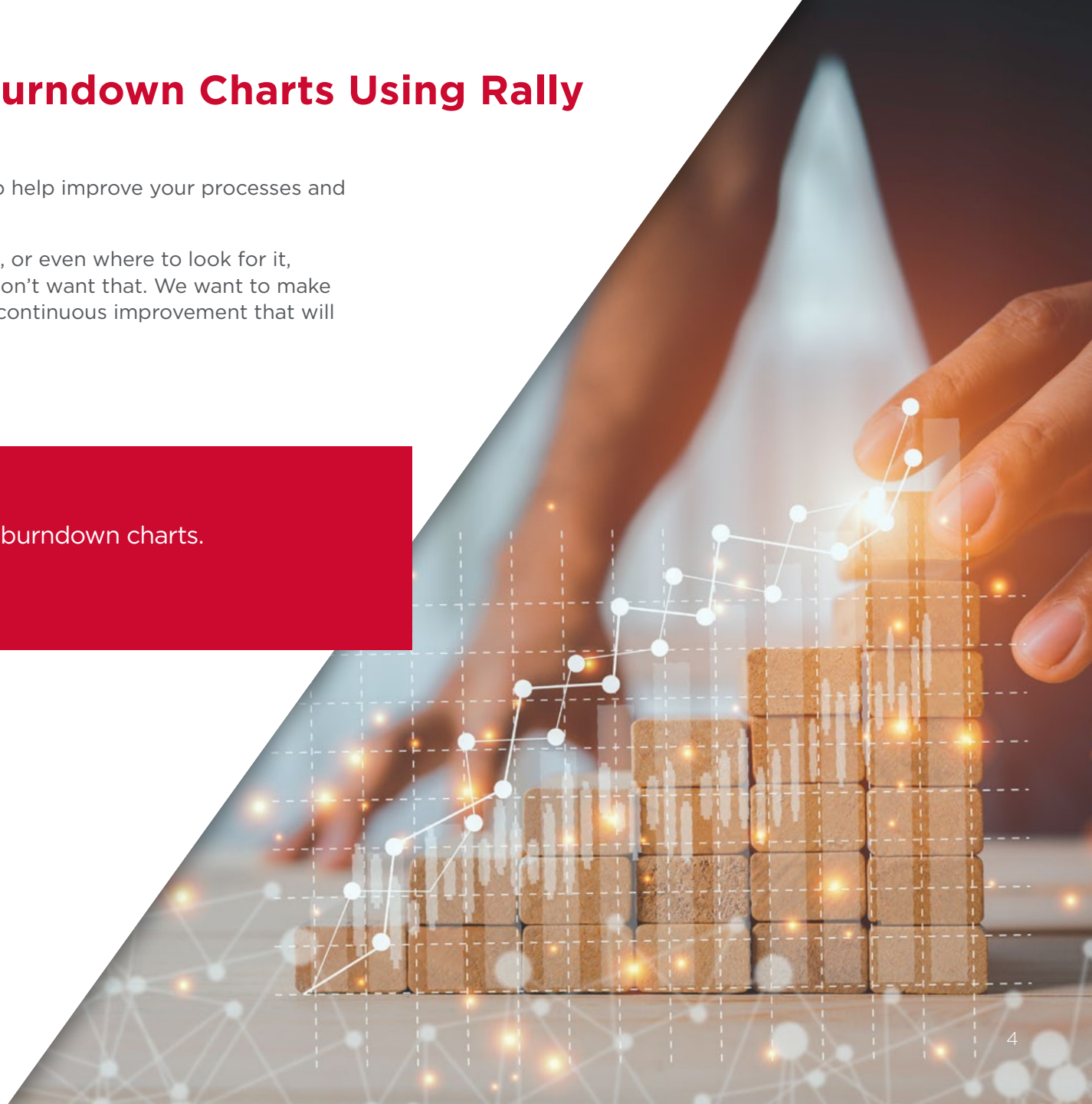
Demystify the Data That You Have Available in Rally and Learn How You Can Use Those Insights to Drive Continuous Improvement

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

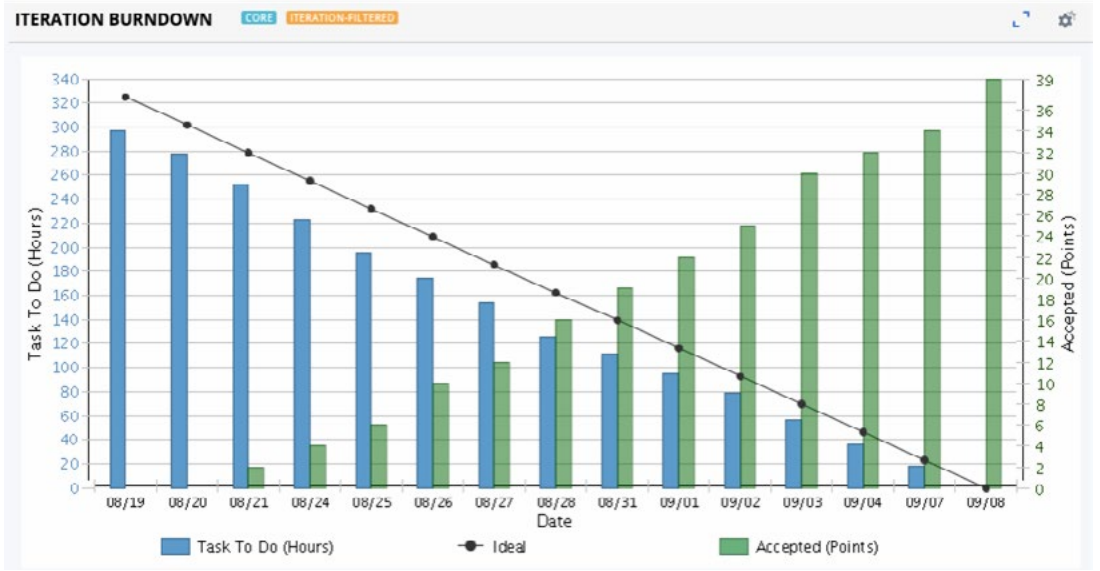
The iteration burndown app displays work remaining and completed in the iteration to proactively anticipate whether the committed work will be delivered by the iteration end date. They look like the example below.

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.

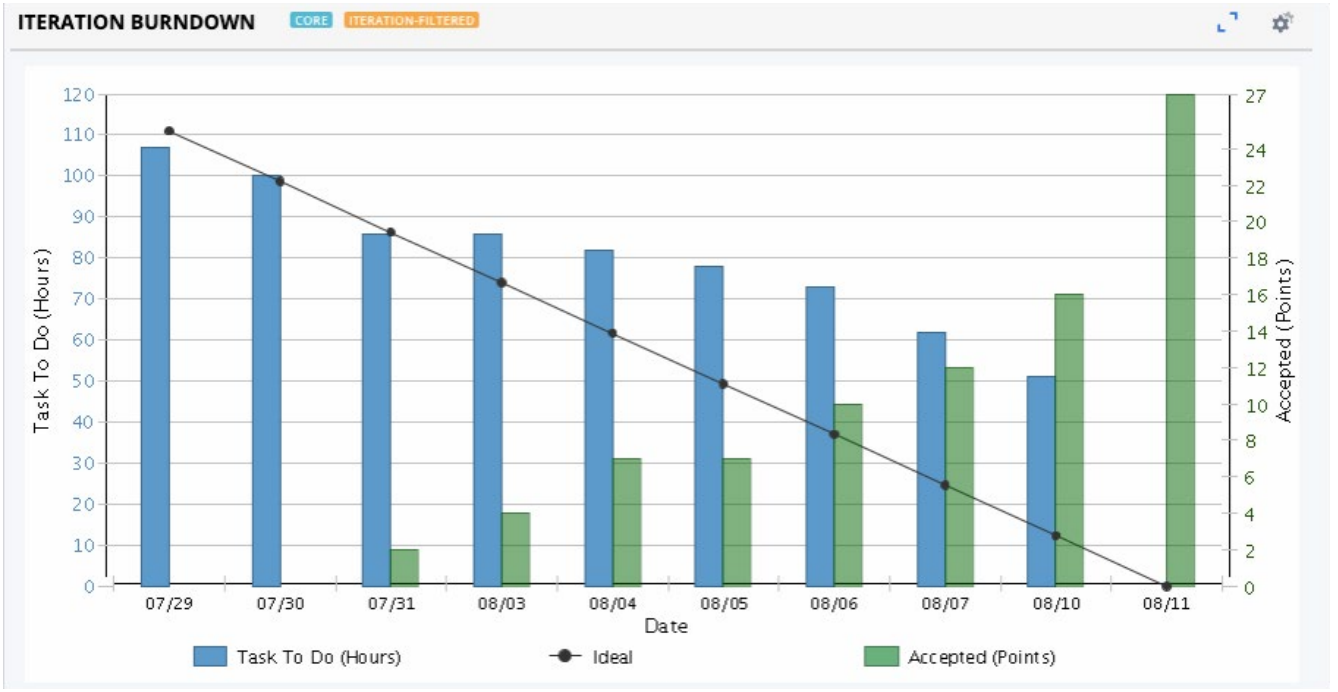


Burndown Examples

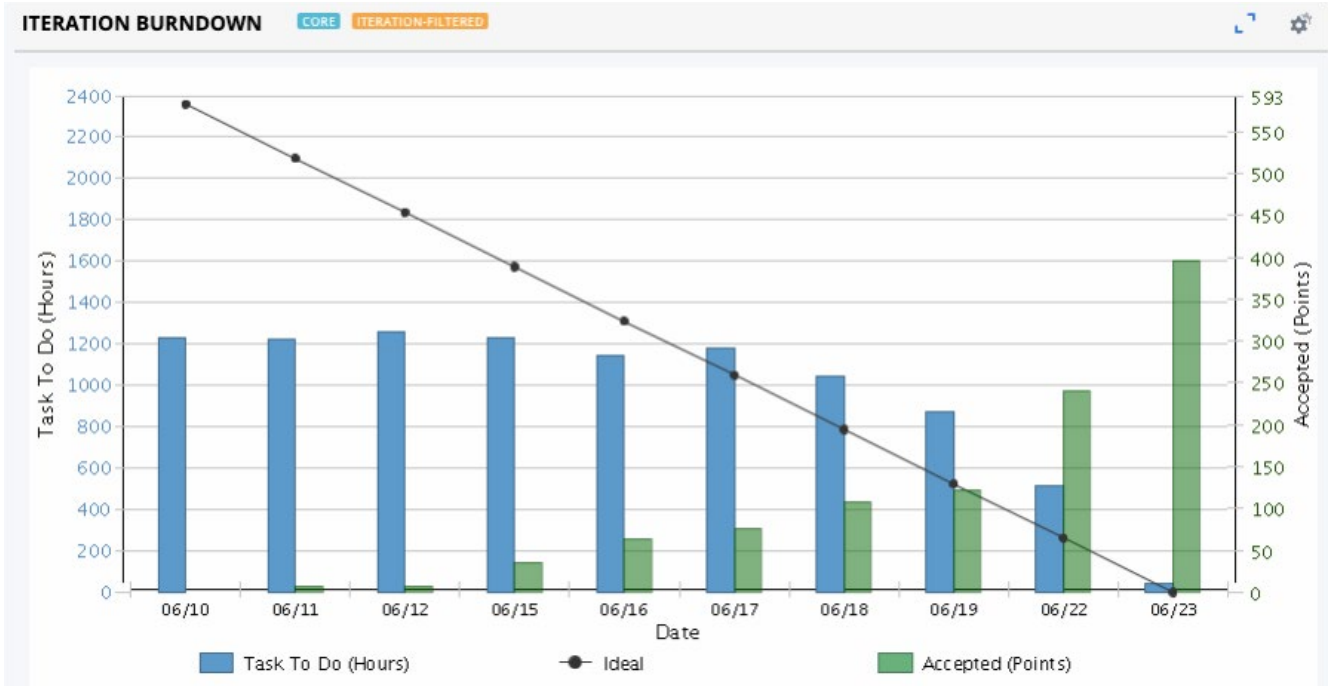
There is only so much learning that can happen from a “happy path” example. To help, we’ve got some real-life examples of burndown chart scenarios. We’ll walk you through what you’re looking at and why the chart looks the way it does.

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')



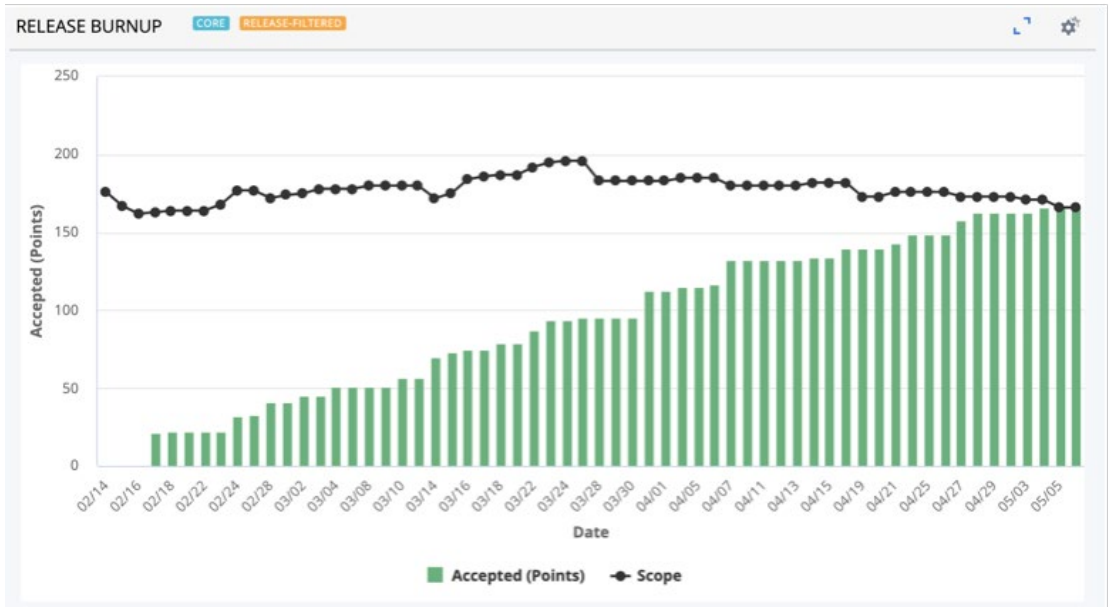
Watch the webinar to learn more related to burndown and burnup charts.

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.

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

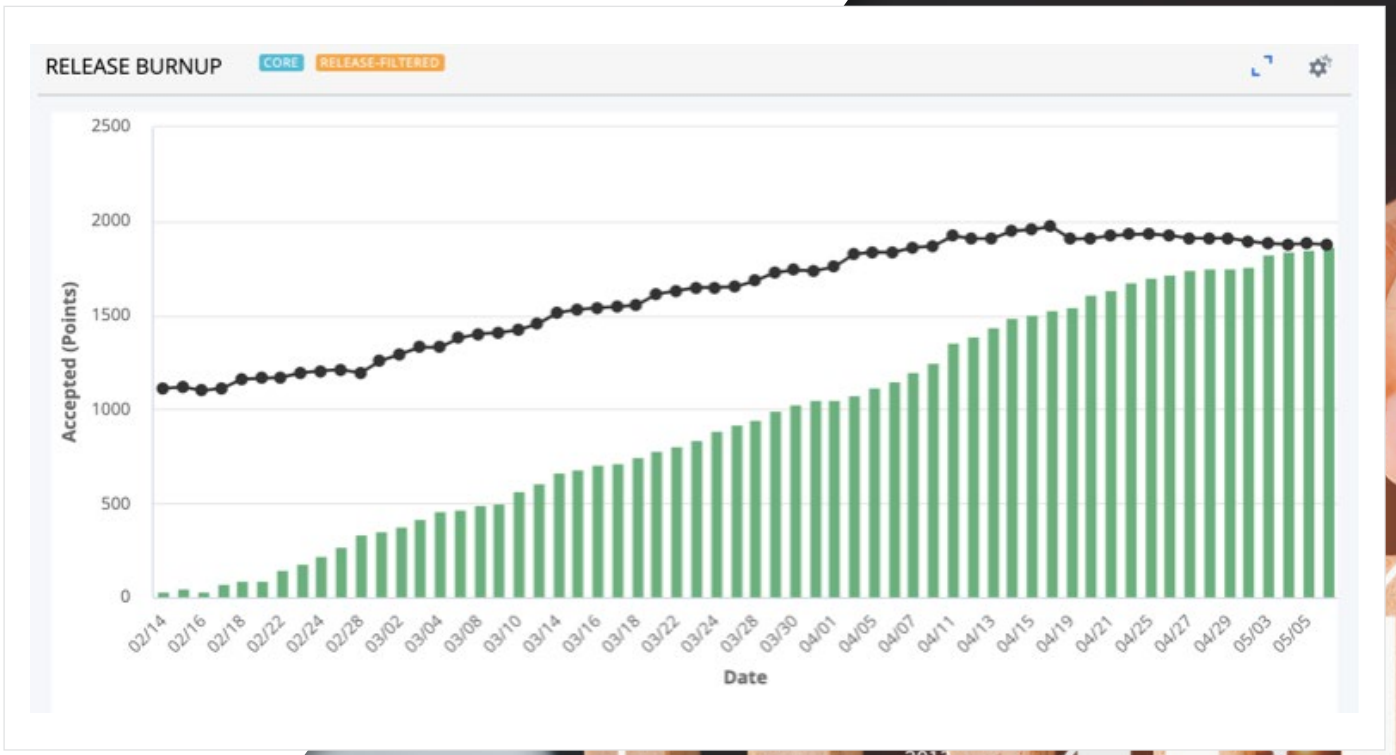
Ideally, on the last day, the scheduled Work Items (black line) and Accepted Work Items (green bars) converge. A gap in the two values on the last day indicates work that wasn't accepted during the Release.

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.

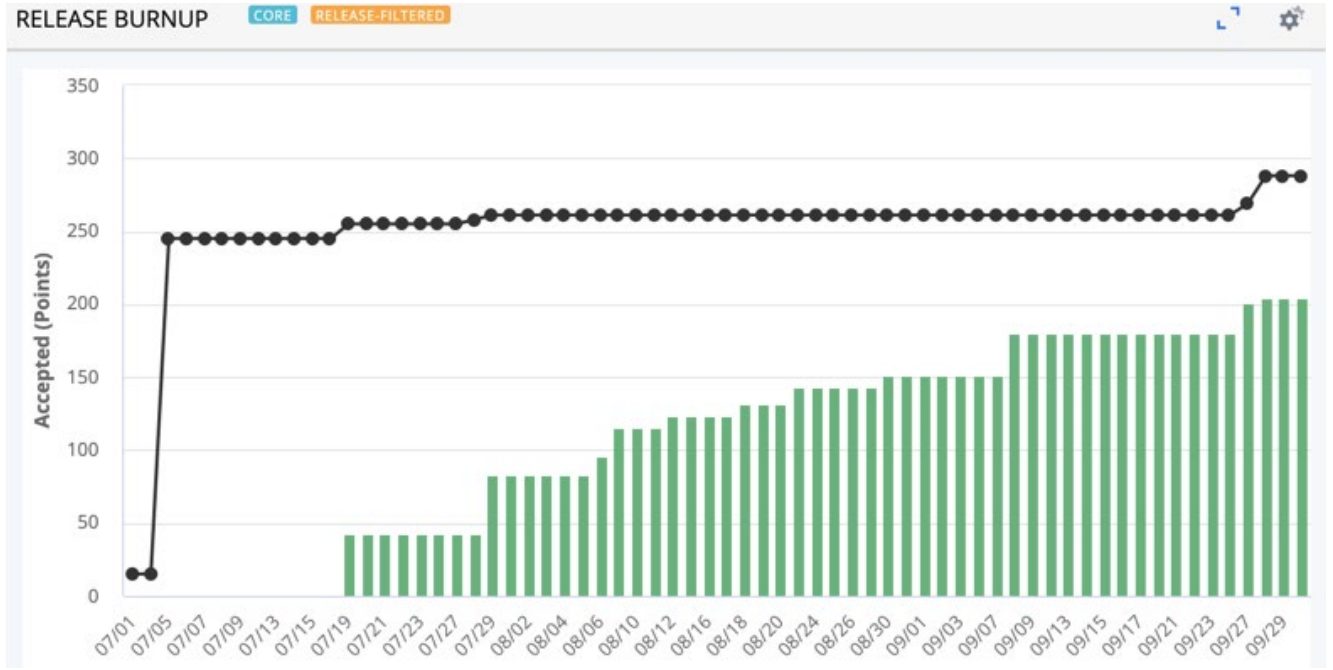
Example #1

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.



Rally®
by Broadcom

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

Rally®
by Broadcom

MASTERING AGILE ANALYTICS
TO UNITE THE ENTERPRISE

HOW TO INTERPRET DATA FROM CUMULATIVE FLOW DIAGRAM CHARTS

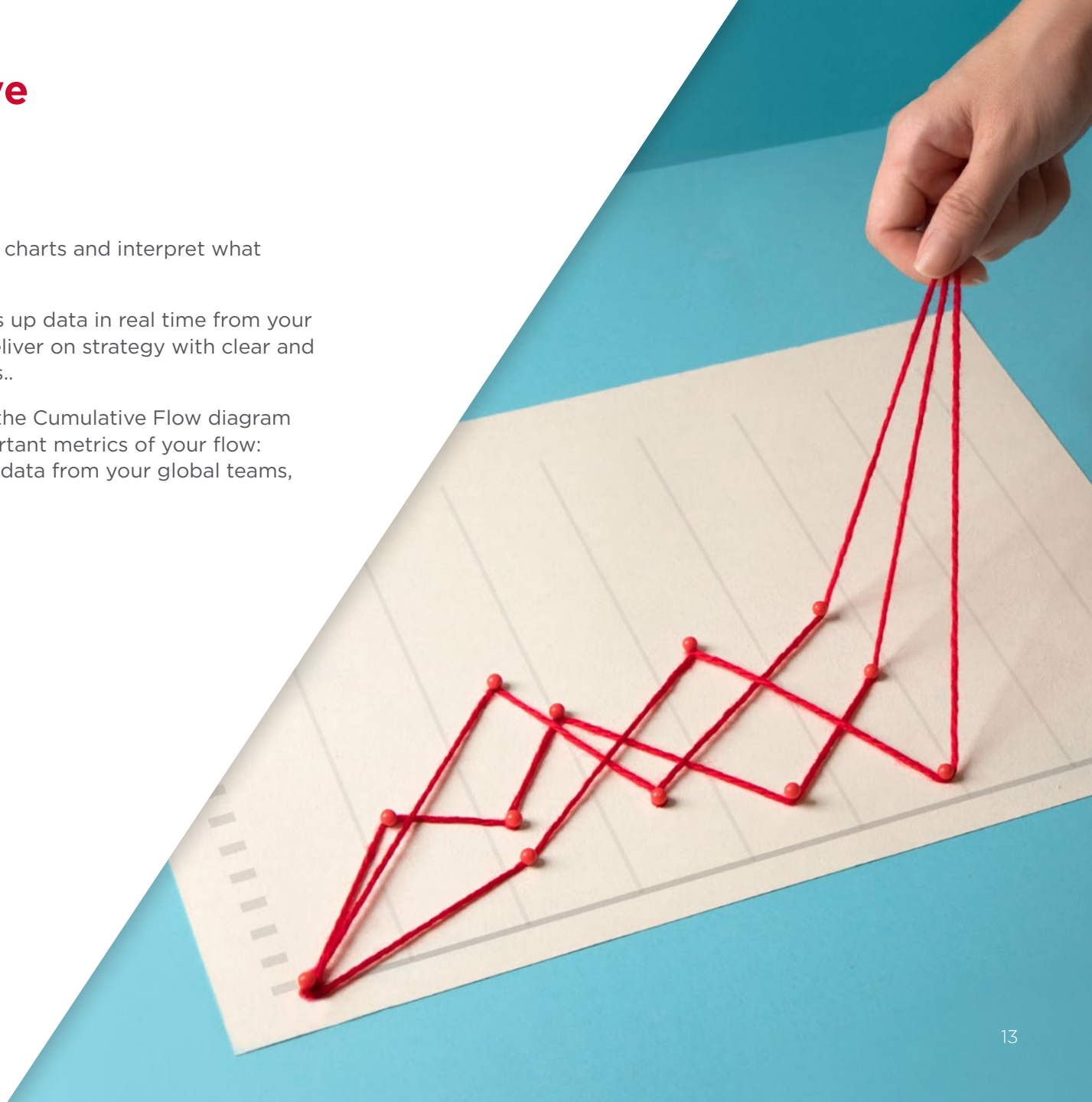
Learn how to get the most of the data for traceability and visibility across the development lifecycle

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.

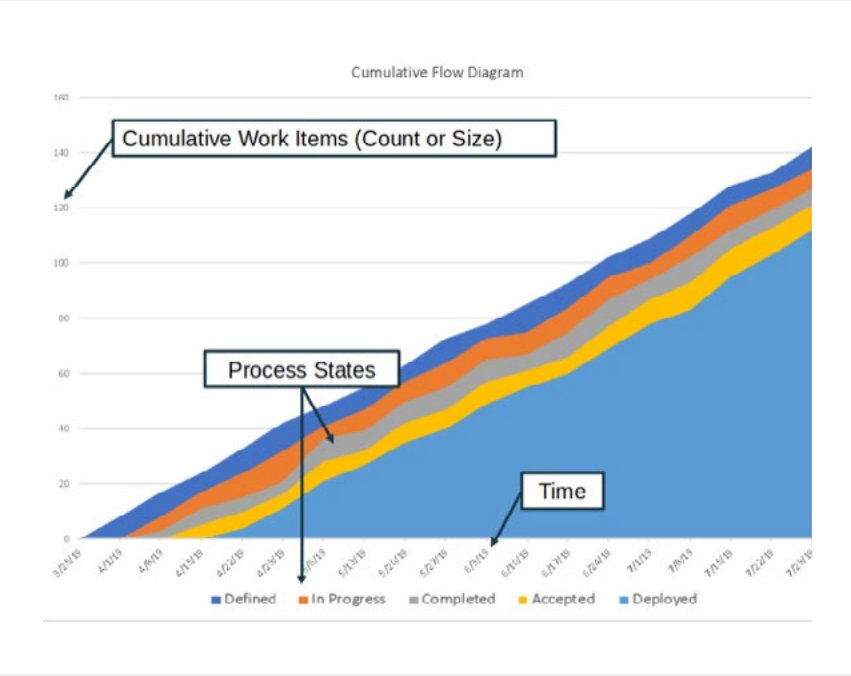


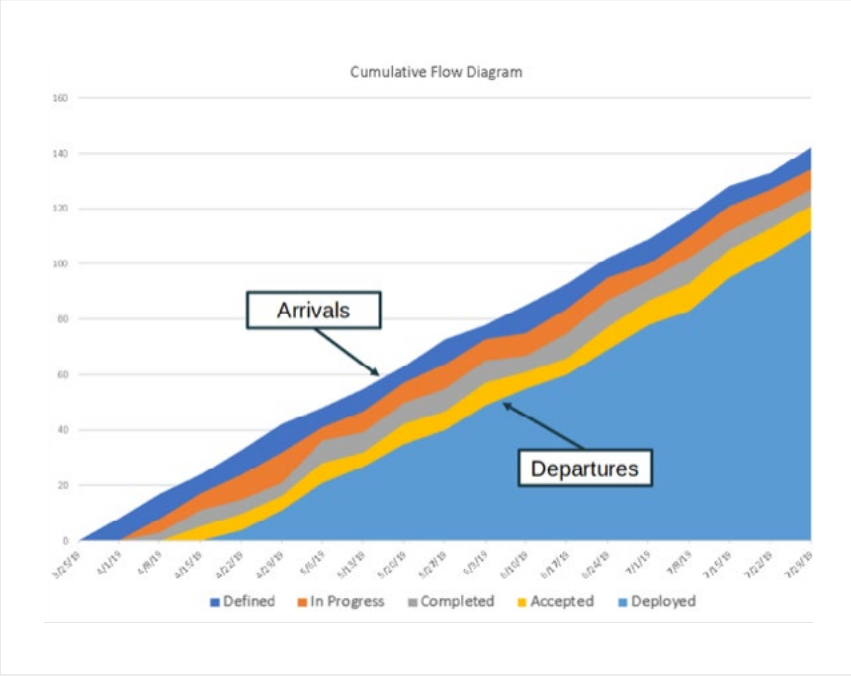
What Is a Cumulative Flow Diagram Chart?

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:

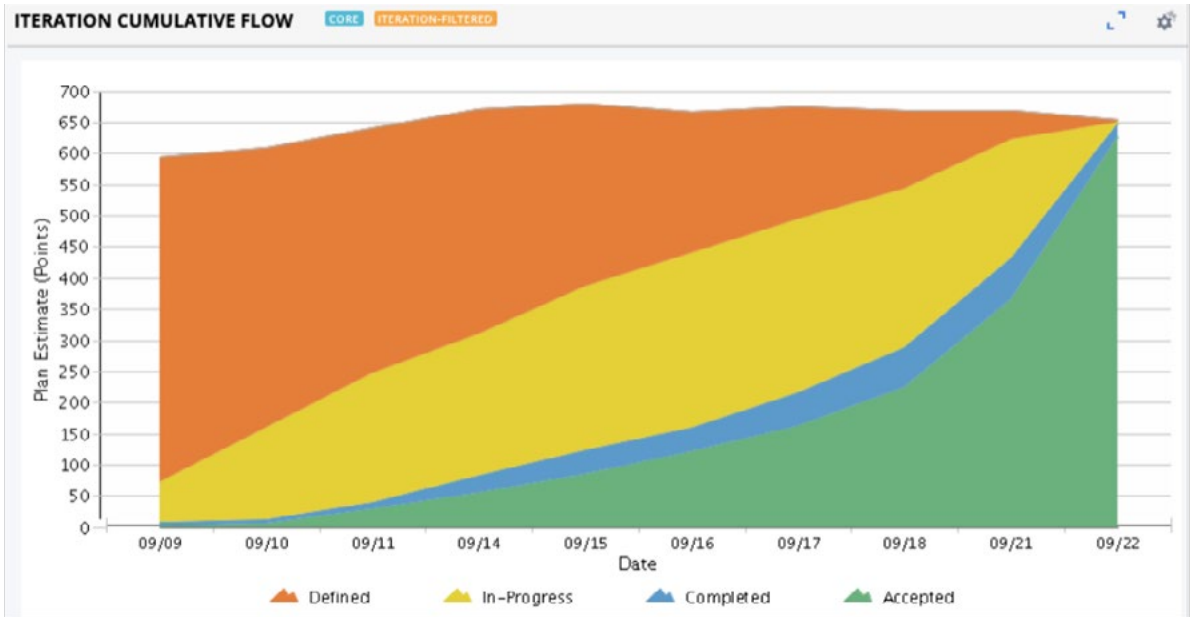
- Iteration CFD
- Release CFD
- Milestone CFD
- Story CFD
- Project CFD
- Portfolio Item CFD
- Team Board — Smart CFD
- And Several custom Apps

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.

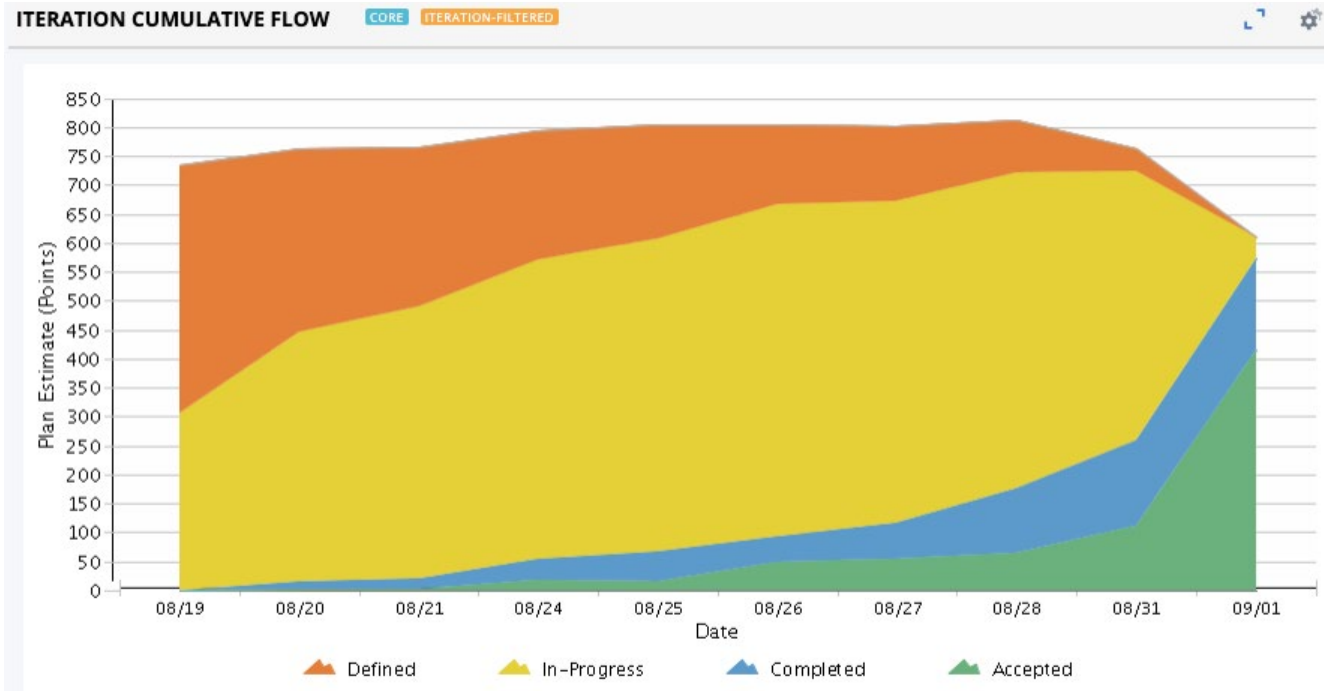
To help ensure that you fully understand what’s going on with these charts, we’ve gathered up some examples of different CFDs. This way you can get a sense of what you’re seeing, why you’re seeing it, and what it means about the way your team is working.

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)

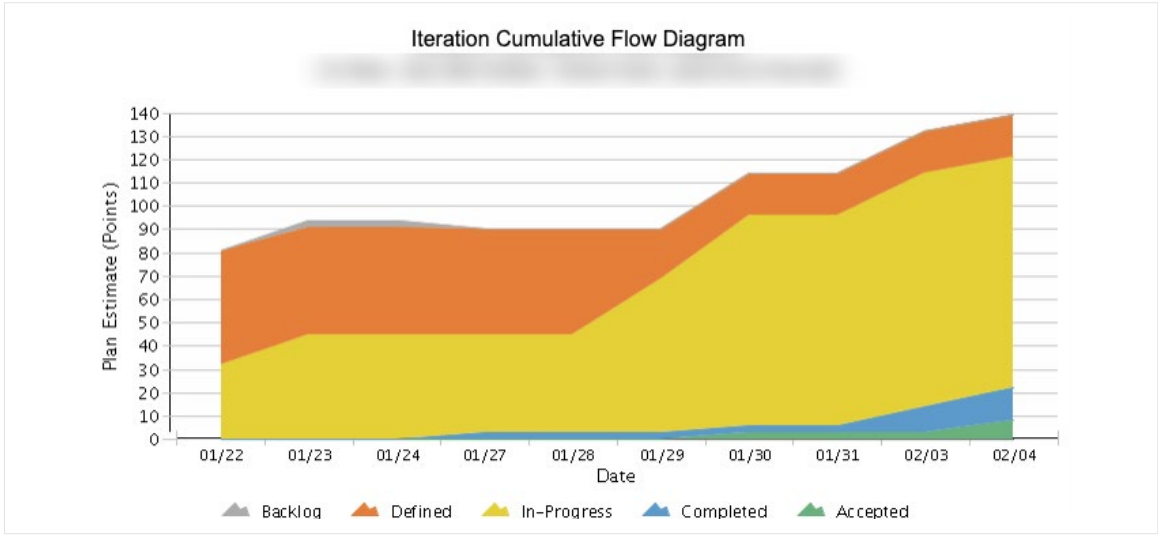
Example #3

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.

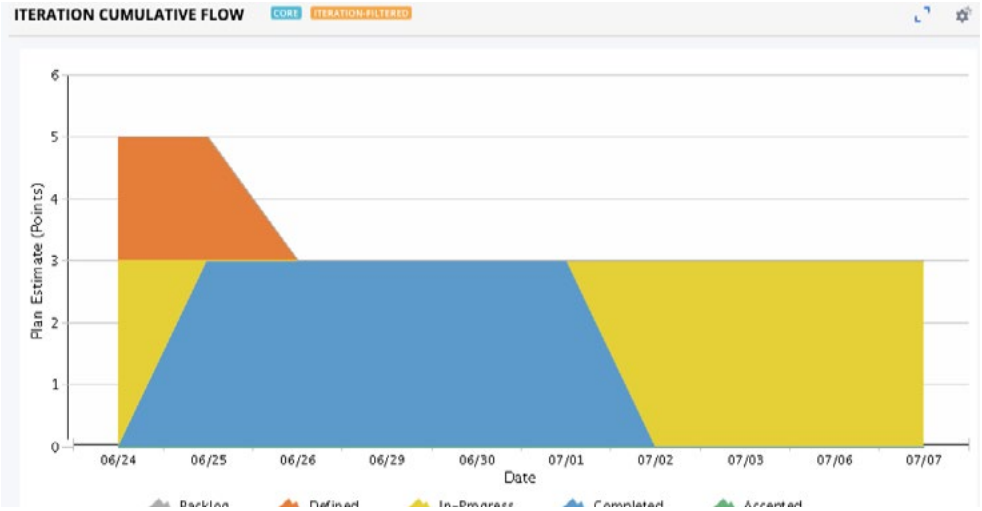
It's not the healthiest-looking CFD, but it gives you an idea of what's going on with your teams and can be a great way to start a conversation about improving. Asking questions can help you get a sense of what's going on.



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

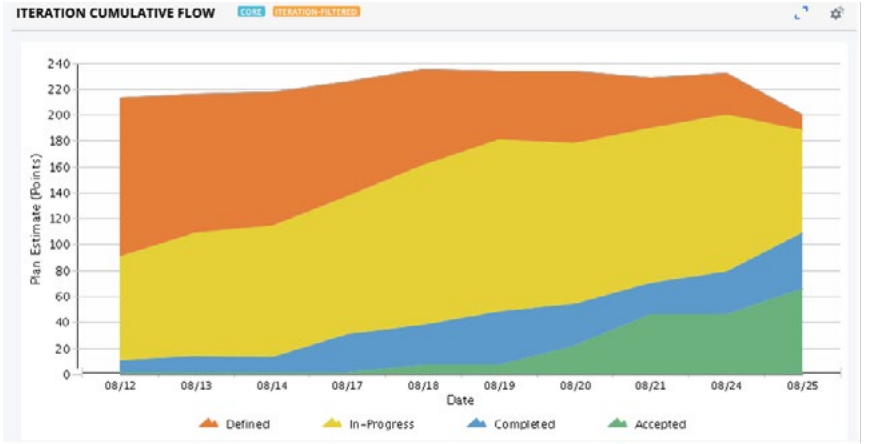
There's too much work happening during this iteration. You can see they're removing things on the last day, there's a lot of work that's still in progress, and not nearly enough work is being completed.

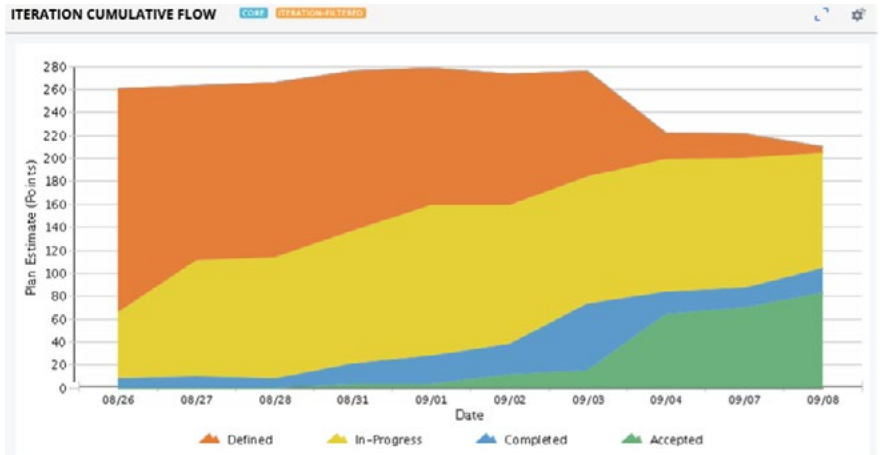
Ask yourself this question about this chart:

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.

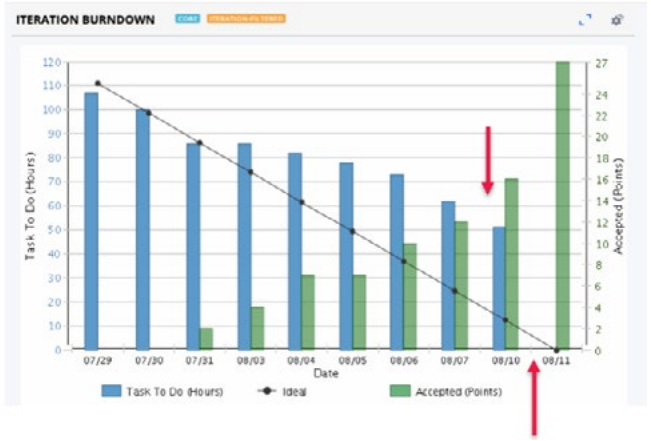
Ask why it's happening and make adjustments based on the answers you're getting. Focus on in-progress work first, then add more.

Cumulative Charts Examples (cont'd)

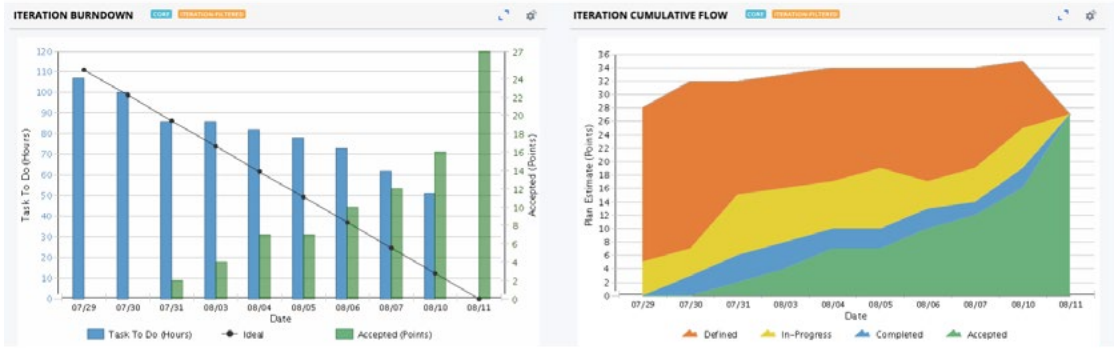
Example #6

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.



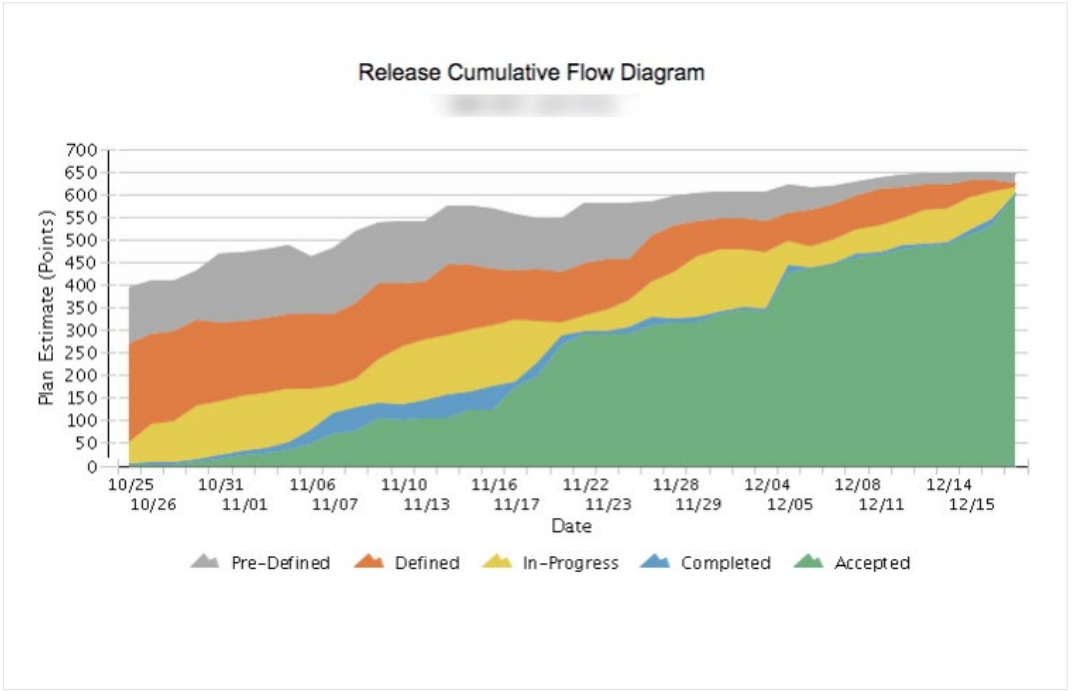
When you combine the two charts, you can see what's going on. They removed a bunch of work on the last day, work they weren't going to get done.



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.



Rally®
by Broadcom

USING DATA TO IMPROVE & PREPARE FOR BIG ROOM PLANNING (BRP)

Gain a stronger and more accurate understanding of capacity using your data

Executive Summary

Big Room Planning (BRP), sometimes referred to as PI Planning, is generally a two-day release train planning event. The purpose of the event is to hold a collaborative planning process, resulting in a committed roadmap for that quarter's Planning Increment (PI).

Agile metrics and data are essential to Big Room Planning so you can answer questions such as:

1. How much capacity do we have for the upcoming release?
2. Are there any known upcoming disruptions?
3. Did we allow for changes to the plan when new, important information emerged?
4. How much time have we historically spent on defects and has our defect debt improved or declined?
5. What is our investment strategy for the upcoming release?

Let's demystify the data so you can improve planning, tracking and forecasting.

What Is Big Room Planning?

Big room planning (BRP) is exactly what it promises. It's an event that serves as a collaborative planning process, resulting in a committed roadmap for a program increment (PI). At Rally, our PIs are quarterly, organized into six or seven two-week sprints, and all teams are on the same sprint cadence. This, of course, will be different depending on how you run things in your organization, but the general idea remains the same—everyone gets together in a big room to plans things out.

With BRP, you're planning out the next release, which means the first step is to define a release (At Rally, we typically plan out over 12 to 14 weeks).

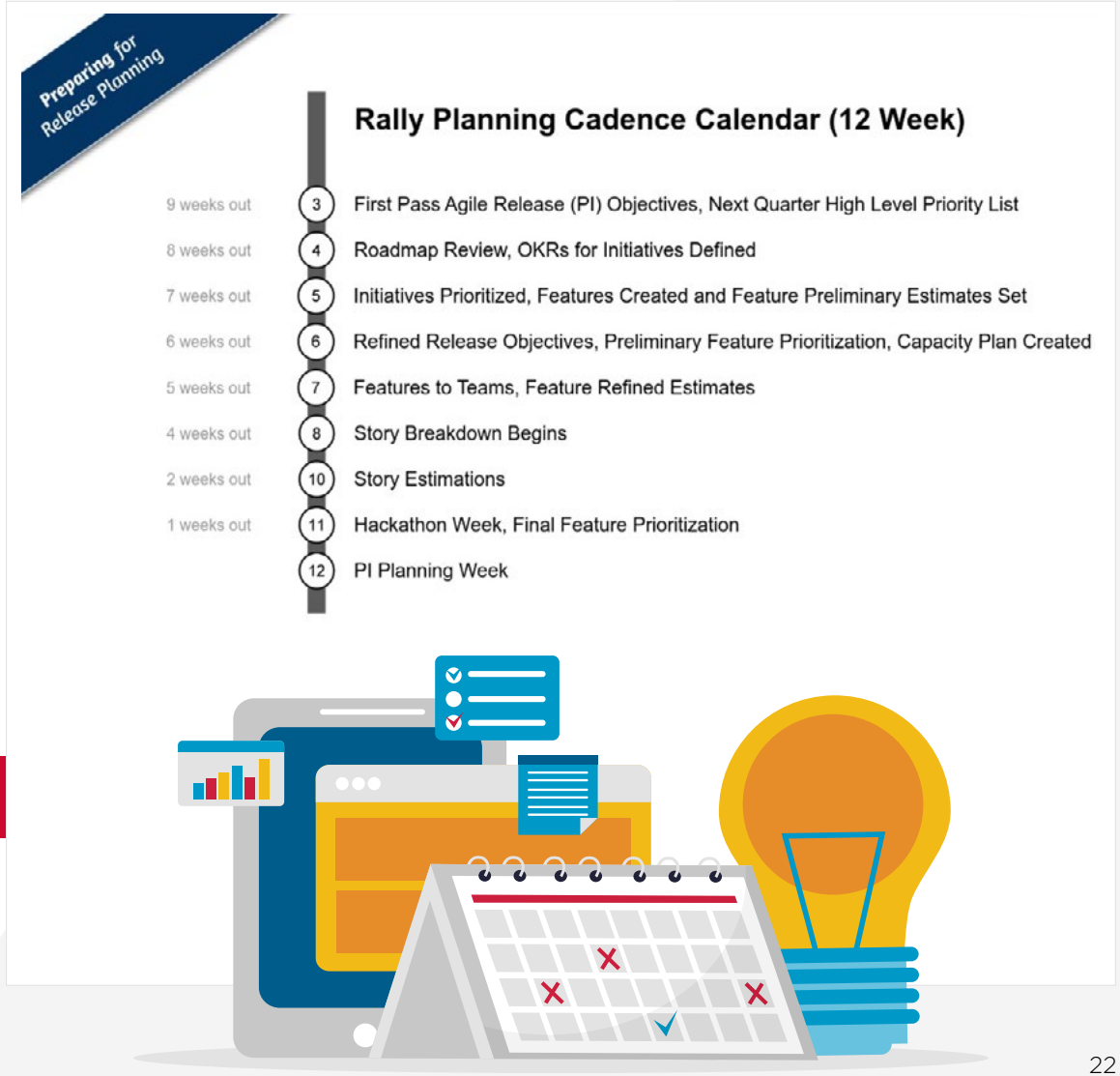
That being said, you can't just show up and start planning. There's a cadence involved in getting ready that starts weeks prior.

The key part of BRP involve details like capacity planning, breaking features down into stories, and sorting out dependencies. Ultimately, you're using past data to inform future decisions.

During PI, it comes down to:

1. Preparing the right work
2. Preparing the right amount of work

Capacity Planning is essential to Program Increment (PI) and Big Room Planning (BRP)



Capacity Planning

We don't always know what we're getting into when we start a project. It can be hard to predict everything that's involved or the unexpected changes that happen along the way. It's expected that scope will change, but what we can't be sure of is how much the scope is going to change and whether or not your team will be able to handle the shift.

The good news is that the data you need to make more informed decisions around capacity planning exists within Rally. And if your global teams are working with different tools, Rally can aggregate that data into a single view. We're going to walk you through what charts and data to look for and where to find that information in Rally so you can better manage capacity.

5 Questions to Determine Capacity

We find it helpful to answer five key questions when thinking about capacity planning. By answeriing these questions, you can make more informed decisions.

1. How much work have we historically accomplished?
2. Do we anticipate additional scope?
3. Are we happy with our defect debt? Do we need to spend more or less time squashing defects?
4. Will the team structure change?
5. Do we have any planned disruptions?

This handbook will help you find the answers to these important questions and show examples of charts and data.

In any organization, there are many work items like features and stories, which can be represented. It comes down to selecting which work items should be selected to plan and prepare for.

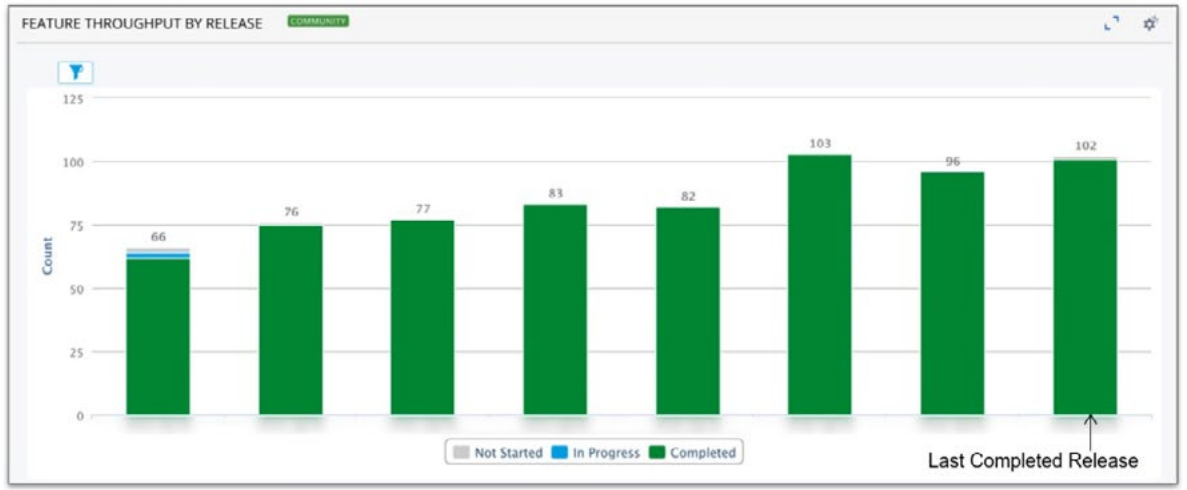
How Much Work Have We Historically Accomplished?

People initially look at burnup or cumulative flow diagrams (CFD) to answer this question. While this data is a good reference, it doesn't show you how many features were completed. Velocity charts and Throughput charts can also be informative, but they also don't tell the complete picture.

The Release Feature Throughput chart is ideal for looking back historically at the work accomplished by teams.

Release Feature Throughput

Pixel Pioneers Agile Release Train (ART)



The Release Feature Throughput chart shows you throughput count per release. The X-axis represents the different quarters of the releases (which is blurred out of this specific example to protect the identify of this company's data).

The last agile release shows 102 items completed.

As you can see in this example, the Pixel Pioneers Agile Release Train is predictable. They manage to complete a steady amount of work through their system.

With this chart, you gain an understanding of how many features your team can deliver and the confidence to predict how much work a team can complete in the future.

In the sample, the Pixel Pioneers Agile Release Train team can confidently deliver 100 features next quarter.

Reminder: It's important to gather multiple data points before making decisions. You can use the same Release Feature Throughput and switch to velocity using Feature size to verify consistency.

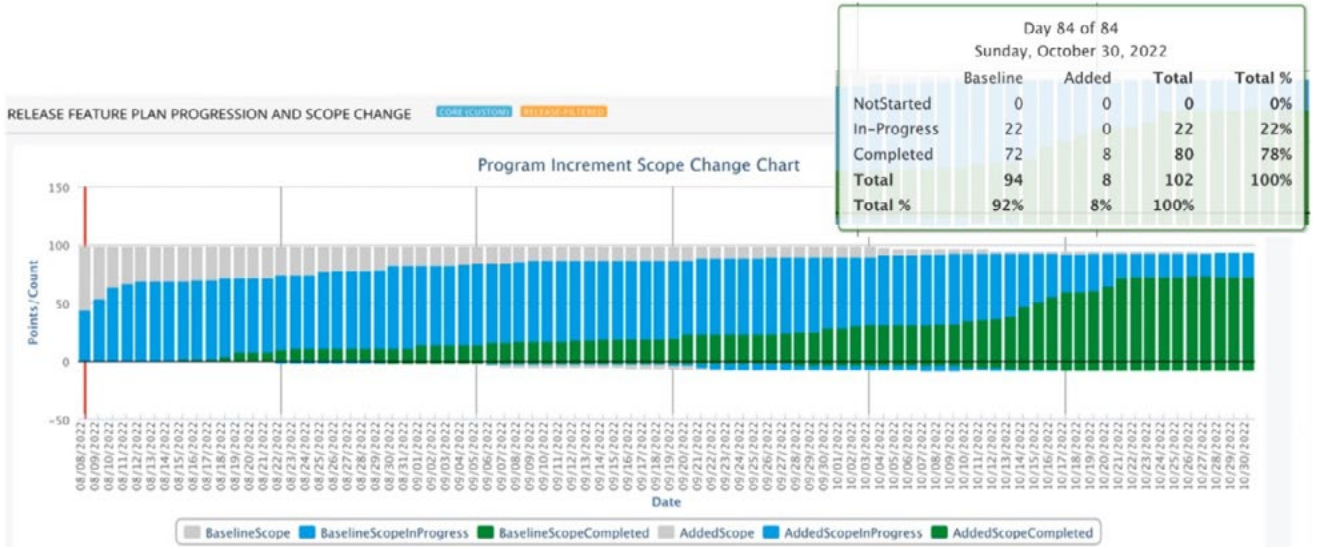
Do We Anticipate Additional Scope?

Scope change is to be expected during most releases. However, too much change can cause problems. How do you find the balance? Burnup charts and Cumulative Flow Diagram charts don't show you if there are any feature scope changes. To estimate scope change, it's important to identify what feature is planned and what feature is unplanned.

To understand how many features were added to the plan, you can use Release Feature Plan Progression and Scope Change chart.

Release Feature Plan Progression and Scope Change

Pixel Pioneers Agile Release Train (ART)



The Release Feature Plan Progression and Scope Change chart identifies what scope change actually looks like in a PI. In this example chart above, we see data that represents the work for the quarter. Everything above the X-axis represents work that was in the plan, before planning was completed. Everything below the X-axis represents work that was added after the team was done planning.

When you click on the "tool tip" in the chart, the box on the right corner pops up showing a summary of insights. In the Total row, you can see the Baseline is 94 and added new features was 8, bringing the total features to be completed at 102. While adding 8 additional features wsn't a huge scope change, the Pixel Pioneers team was agile enough to incorporate these feature asks into their work. Remember that it's important to look at historical releases to see if this team has a similar historical average.

Do We Anticipate Additional Scope? (Continued)

Release Feature Plan Progression and Scope Change

Example 2: What are the causionary signs for this team's program execusion during their quarterly plan?

Day 13 of 84

	Baseline	Added	Total	Total %
NotStarted	216	0	216	66%
In-Progress	93	0	93	28%
Completed	19	0	19	6%
Total	328	0	328	100%
Total %	100%	0%	100%	

Day 82 of 84

	Baseline	Added	Total	Total %
NotStarted	13	25	38	12%
In-Progress	47	26	73	22%
Completed	151	63	214	66%
Total	211	114	325	100%
Total %	65%	35%	100%	

You have to get predictable before you can get fast.

This is another example of the Release Feature Plan Progression and Scope Change chart representing data from another team.

Causionary Signs:

1. The team started work before planning was completed, represented by the red virtical line at Day 13
2. The team steered a lot of work over the PI
3. The team steered a lot of work in the end
4. Only completed about 50% of work initially planned
5. The team received 114 additional work (scope change) during the cycle
6. There is no scope change process

Tip for Managing Scope Creep

Bring the team together 2-3 times during the Program Increment (PI) to check-in

Scope changes require a conversation about how that might reflect against current capacity: Is another feature coming out? Was another feature scoped smaller than was originally thought?

If you pick the right work at the beginning of the PI, that limits scope creep and will reduce churn.

Are We Happy with Our Defect Debt?
Do We Need to Spend More or Less Time Squashing Defects?

It's important for organizations to look at defect debt during PI and work to reduce these. Before your PI, you can ask:

1. Have we been able to keep our defect debt under control?
2. Will we need more or less capacity in the upcoming release to address active defects?
3. If we need more or less capacity, what impact will that have on our feature capacity?

The Defect Trend chart is a great place to start to look at your defects. For the Pixel Pioneers Team chart to the right, look at the red line and the green line. The red line represents the Cumulative Activated and the green line represents Cumulative Terminated. This data was set to pull over a 12 month span. The green and red lines will always start at zero. The Y-axis on the left represents the Cumulative Defect Count and on the right, Active Defect Count.

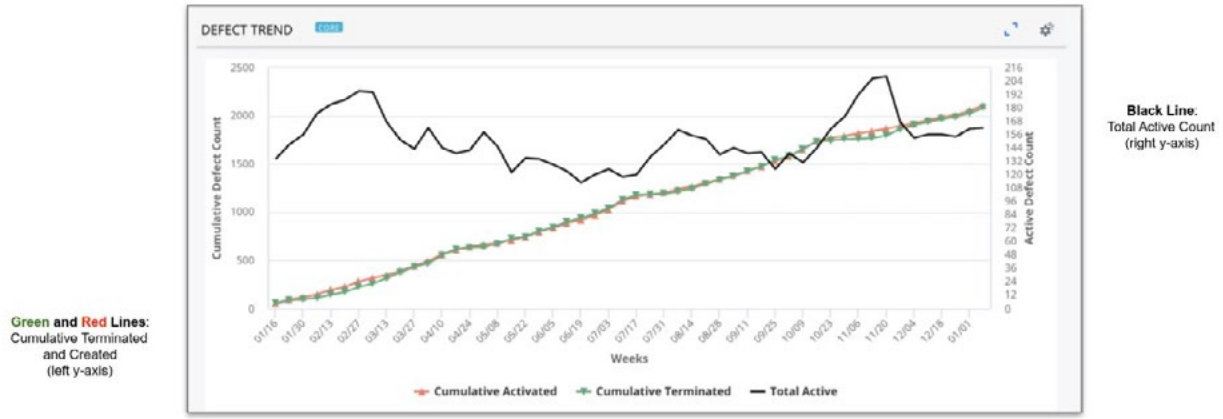
When you see the red and green lines on top of each other, it tells a story that the Pixel Pioneers team was closing defects at the same rate that the defects were being opened.

The black line represents Total Active. While it moved up and down over this period of time, the Pixel Pioneers Team ended the period with 156 active defects.

Recommendation:
Reserve an additional 5% to accelerate defect closures to avoid costly impacts in the future.

Defect Trend

Pixel Pioneers Agile Release Train (ART)

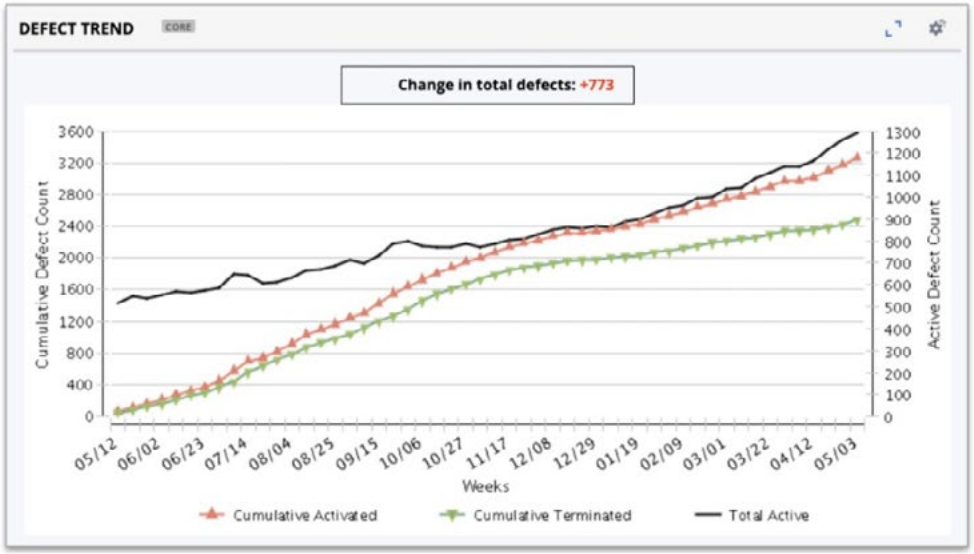


Rally® by Broadcom

Are We Happy with Our Defect Debt?
Do We Need to Spend More or Less Time Squashing Defects? (Continued)

Defect Trend

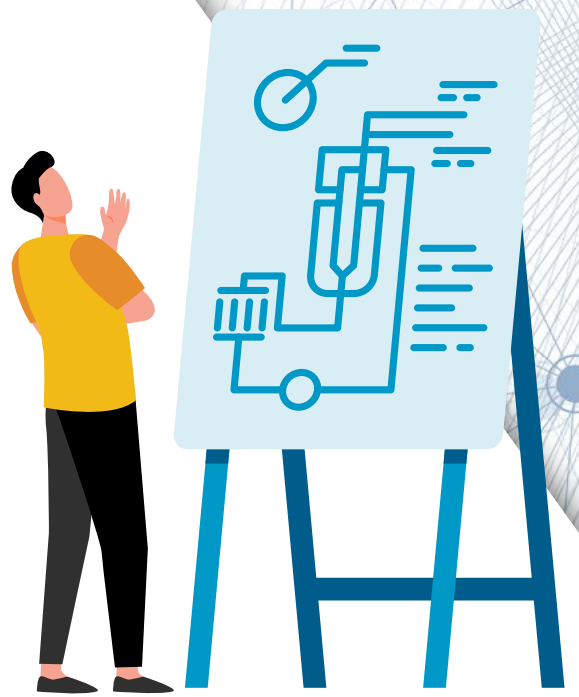
Example 2 Represents Another Team's Defect Trend chart for different observation.



The Cumulative Active (red line) slope is greater than the slope of the Cumulative Terminated (green line). You can interpret this to mean the team is opening defects faster than they are terminating defects.

The Total Active (black line) represents the active defects which is also increasing. In 1 year, this team's defects jumpted to 773 which is doubled the number from the beginning of the year.

Recommendation:
This team must account for defects in their capacity planning to avoid doubling in size year over year.

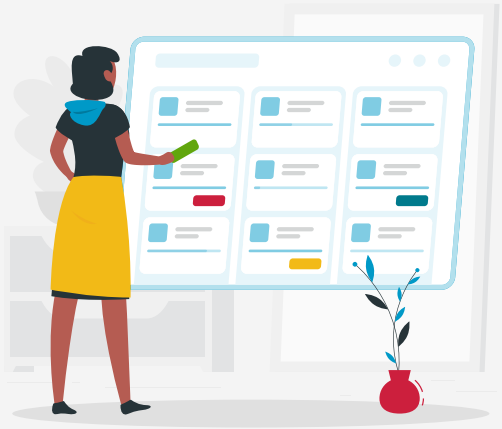


How Rally Manages Defects

Rally focuses on CV Defects, which are Customer Voice Defects. These are defects reported by our customers which are Rally's major focus.

In this chart, Rally has defects broken down by quarter and by state. In 2020 Q1, Rally had 116 CV Defects. In 2022 Q3, Rally reduced the CV Defects to 28, of which some are in a state of submitted and some are closed.

You can also leverage the Cumulative Flow Diagram Chart to see the same result and drill into specific data, like the Open Defects only.



Customer Voice* (CV) Defects

This chart represents Rally's internal team data.

** Customer Voice (CV) defects are defects reported by customers.*

Rally®
by Broadcom

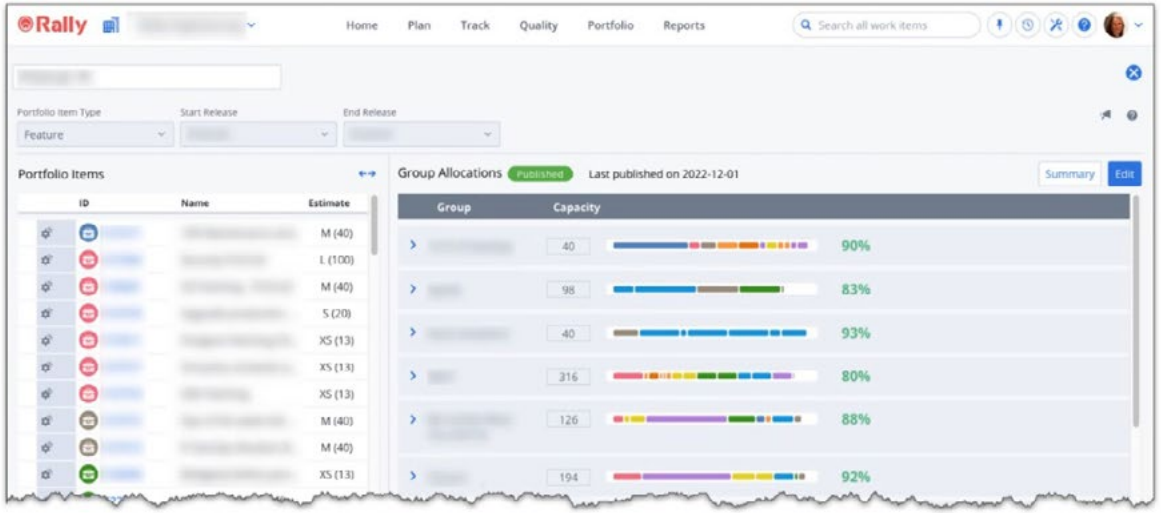
It’s Easy to Set Capacity using Rally

This is Rally’s Capacity Planning page. Each team would use this page to set capacity. Notice that each team is never at 100%, which leaves room for the unknown.

Capacity planning allows leaders to develop realistic plans for a PI in support of priority business initiatives. Capacity is never a single value but multiple values that are used to determine progress.

- “What-if” scenarios that optimize business value (demand) with team capacity (supply).
- Visualization of both demand (features) and supply (teams) in a single view.
- Provides an objective view of too much demand versus not enough capacity.
- Facilitates meaningful above/below-the-line discussions between business and engineering.

Setting Capacity



Want to Learn More?

We covered BRP and capacity planning in-depth in a recent webinar. If you’d like to explore the examples we’ve shown here (and more) in greater detail and gain even more insights from our experts, **check it out today.**

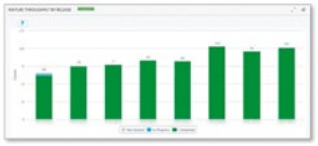
Important Charts to Use to Prepare for Big Room Planning (BRP) or PI (Program Increment) Planning

Release Burnup
Reports Page and App Catalog

Velocity Chart
Reports Page and App Catalog

Release Cumulative Flow Diagram
Reports Page and App Catalog

Throughput
Reports Page and App Catalog



Feature Throughput and Velocity Chart
App Name: PI Throughput/Velocity Chart

App Catalog Source: Community

Configuration:
- Type: Feature (lowest level Portfolio Item)
- Aggregate by: Count (for Throughput), Preliminary Estimate (for Velocity)
- Bucket by: Release

Capacity Planning
Portfolio Tab

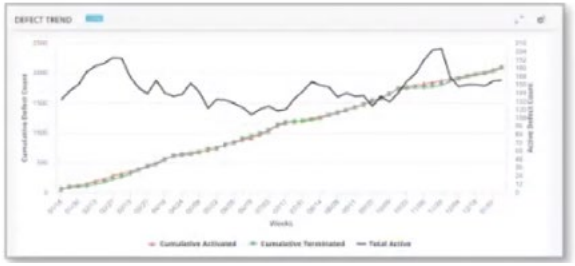
Plan Progression
Portfolio Tab

Feature Release Plan Progression and Scope Change
App Name: Program Increment Scope Change Chart

App Catalog Source: Community

Configuration:
- Aggregate Type: Count (if not using Preliminary Estimate consistently), Preliminary Estimate (for CFD by Feature size)
- Baseline Type: End of first day (if planning is finished by the Release Start Date), End of first Spring (if planning continues during the first sprint of the Release)

Note, this app has to be on a Release Filtered dashboard.



Defect Trend Chart
Reports Page and App Catalog

Note: Additional configuration options are available on the app vs. the chart on the Reports page.

Defect Dashboard
Multiple apps on single dashboard. Includes two custom chart apps.

App Name: Custom Chart

App Catalog Source: Community

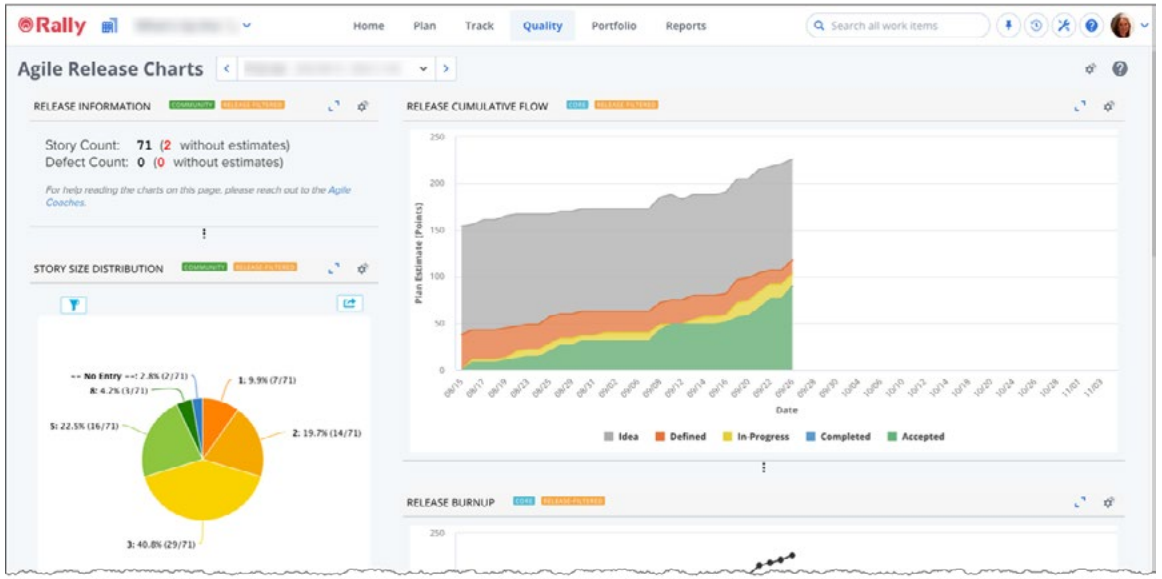
Configuration of Pie Chart:
- Chart type: Pie Chart
- Type: Defect
- Aggregate by: State
- Aggregation type: Count

Configuration of Column Chart:
- Chart type: Column Chart
- Type: Defect
- Aggregate by: Creation Date
- Bucket by: Quarter
- Aggregation type: Count
- Stack by: State

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.



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.**