Why Your Agile Isn’t

presented by: Travis Alexander | travisa@empyra.com

Why Your Agile Isn’t

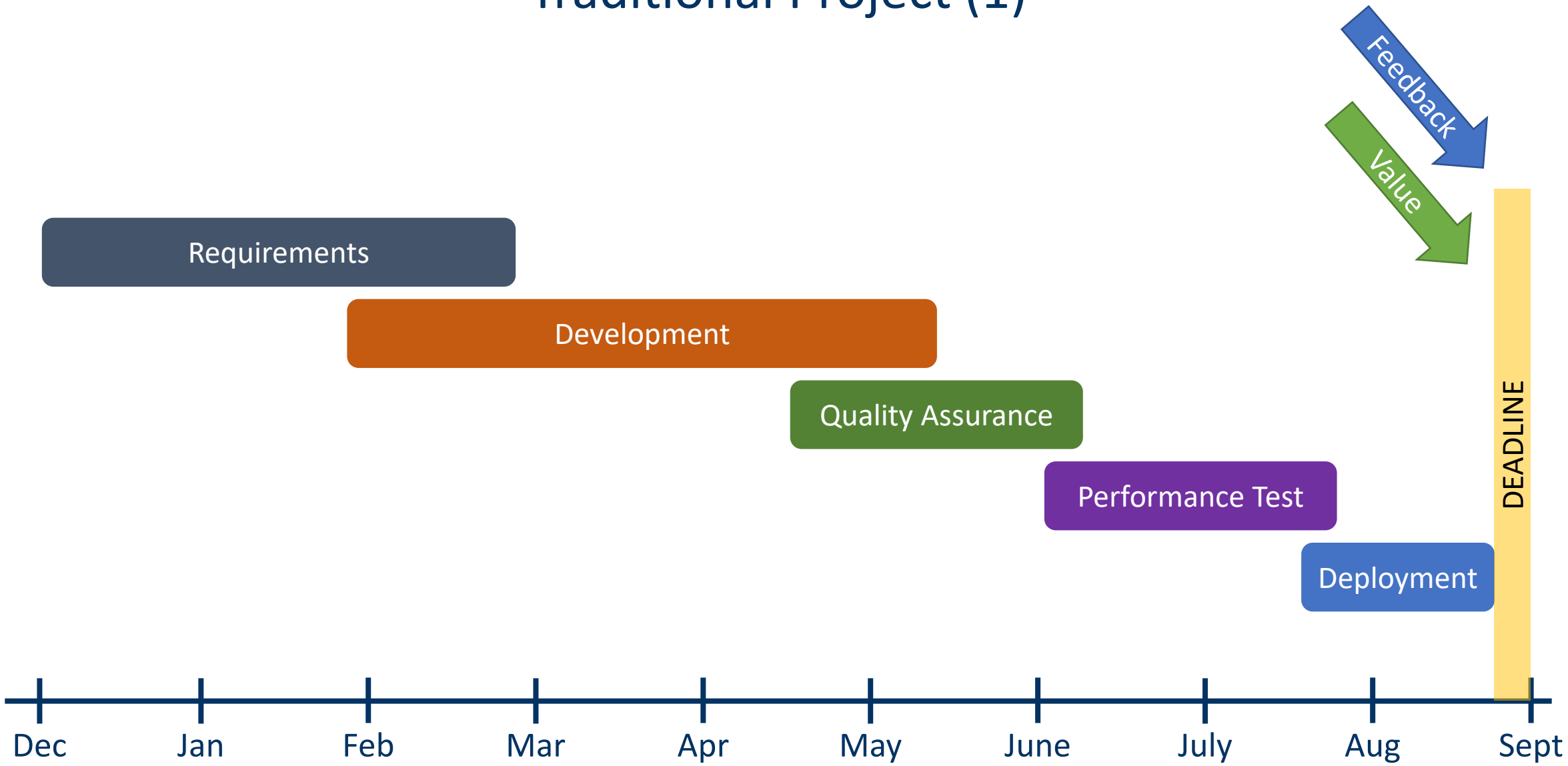
- Your theory game is weaksauce
- Your team operates like a RedBox
- Your backlog got no sex appeal
- Your walls are covered in spaghetti
- Honestly.. your sprints are kind of slutty
- You’re confusing this with AA
- You didn’t fire Nick

Your theory game is weaksauce

Perhaps you've taken it for granted
that the whole team understands the real goals
and benefits of agile theory.

Understanding the goals and benefits of agile naturally drives a proper, better execution of the individual components of the framework.

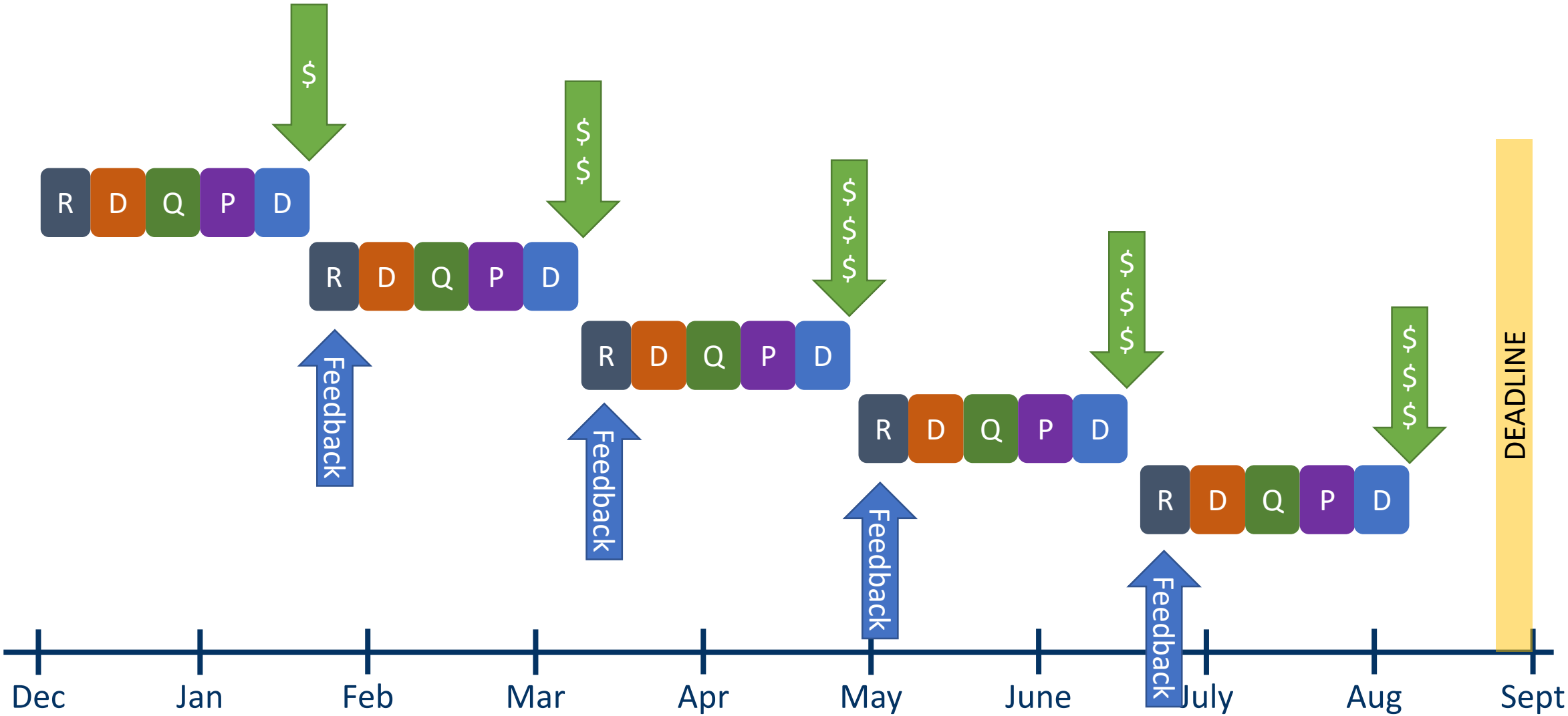
Traditional Project (1)



What is Agile?

In it’s simplest form, it is taking a project and splitting it into smaller projects that collectively accomplish the goal of the original project.

Agile Project



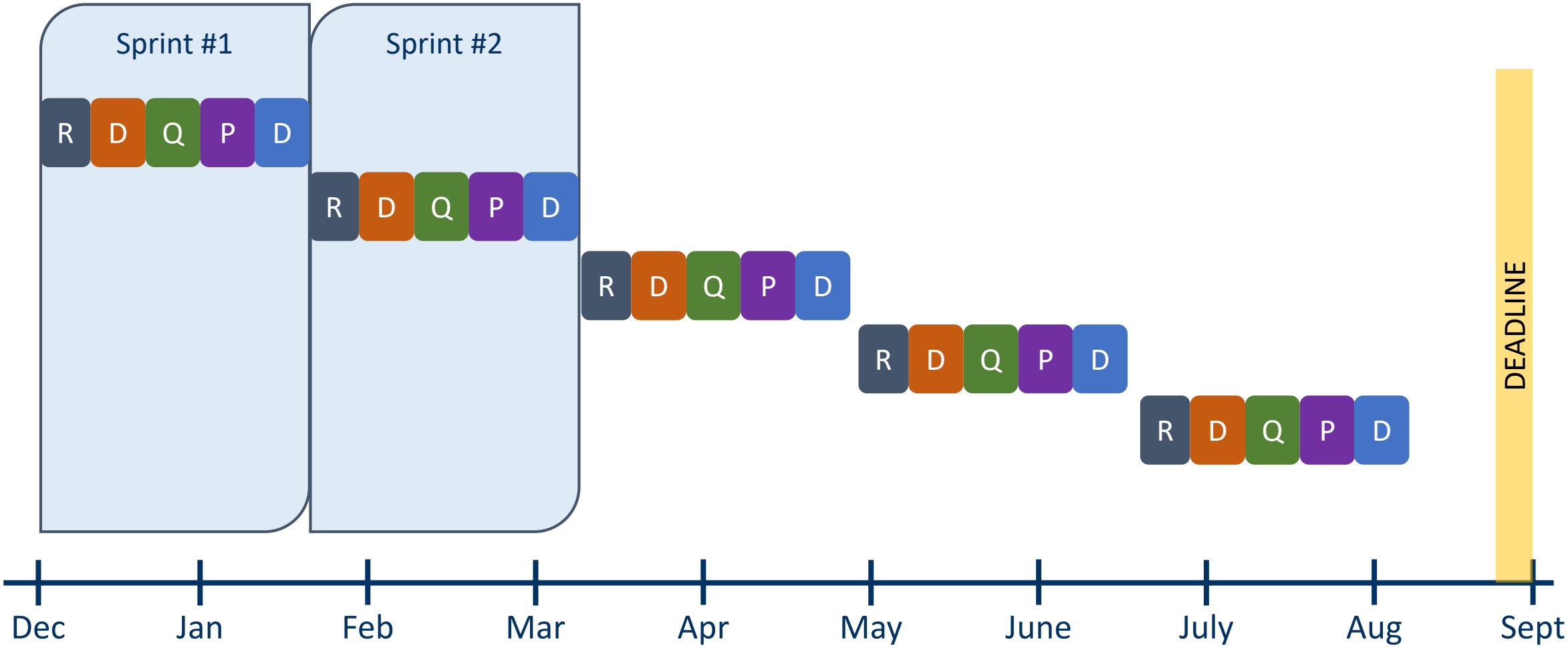
What is Scrum?

Scrum is a framework within which people can address problems, while productively and delivering high-value products.

Scrum is **not** a process or a technique for building products rather, it's a **framework** within which you can employ various processes and techniques.

Therefore, in essence, virtually every team who practices scrum will follow a different process within the framework of scrum.

Scrum Project



Scrum Essentials

- Devoted, cross-functional teams
- Product/project backlogs
- Regular sprint/iteration schedule(s)
- Cross-resource estimates
- Sprint Planning and Sprint Kickoff Meetings
- Standup/Daily Scrum Meetings
- Retrospective Meetings
- A tool to facilitate the process

#AgileGoals

1. Less code, do things once
2. Less dependencies, more agility
3. Less risk
4. Less bugs, better quality
4. Less wait times, more value

Your team operates like a RedBox

It appears likely that the team is not currently organizationally conducive to an agile/scrum framework.

Agile doesn’t run itself; teams are literally the foundation that the whole framework operates upon.

Team Structure

- Optimal development team size is small enough to remain nimble and large enough to complete significant work within a sprint
- Fewer than three development team members decrease interaction and results in smaller productivity gains
- Smaller development teams may encounter skill constraints during the Sprint
- Teams should be as cross-functional as possible with regards to development; large specializations within the team are bottle-neck risks

Team Structure (2)

- Ideally, each scrum team consists of
  - Scrum Master
  - Product Manager / Business Analyst
  - Developer(s)
  - QA Resource(s)

- Organizations should do there best to ensure teams have devoted team members and do not share team member’s time across other groups

Think of teams like this..

Finished Product(s)

Without all it's parts, it doesn't work

It's not 'How many developers do you have?'
It's 'How many teams do we have?'

It's not "Who has free time?"
It's "What's on the team's backlog?"

How to Fix It

- If teams are too big, break them into smaller teams, even if it’s the same project
- If teams are shared across projects, change it. If you can’t change it, share the same sprints
- Lower individual available work capacity to account for other work
- Pair-program to breed cross-functional development; value moldable, well-rounded people over niche experts when hiring
- Hire the additional resources you need; full time scrum masters are not a waste of money and downtime is not a bad thing – you’ll pay for it in bottlenecks if you don’t
- If you only have one scrum master, use Team Leads that are in sync with Scrum Master
- If management isn’t having it, see about getting an outside person to come in

Your backlog got no sex appeal

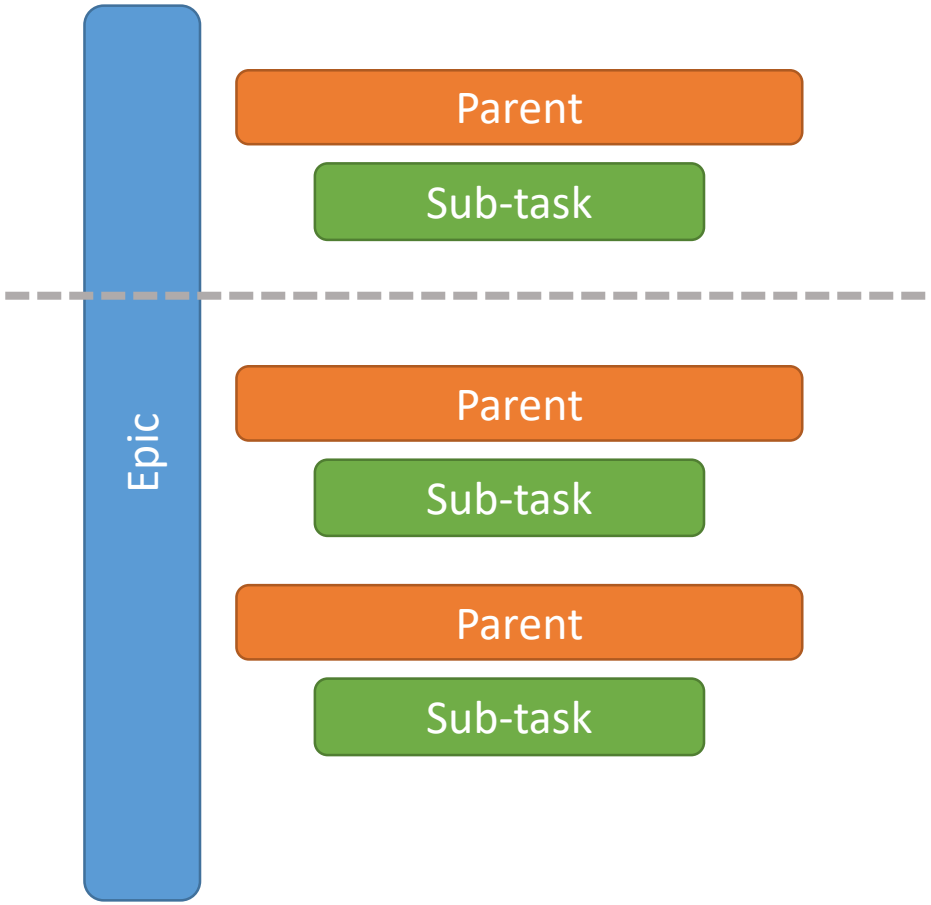
The processes and procedures surrounding how the team breaks down, organizes, and prioritizes their work.. could look better.

And unorganized backlog isn’t really an agile, product, or team backlog.
It’s just a list of things someone should do.

How to Fix It

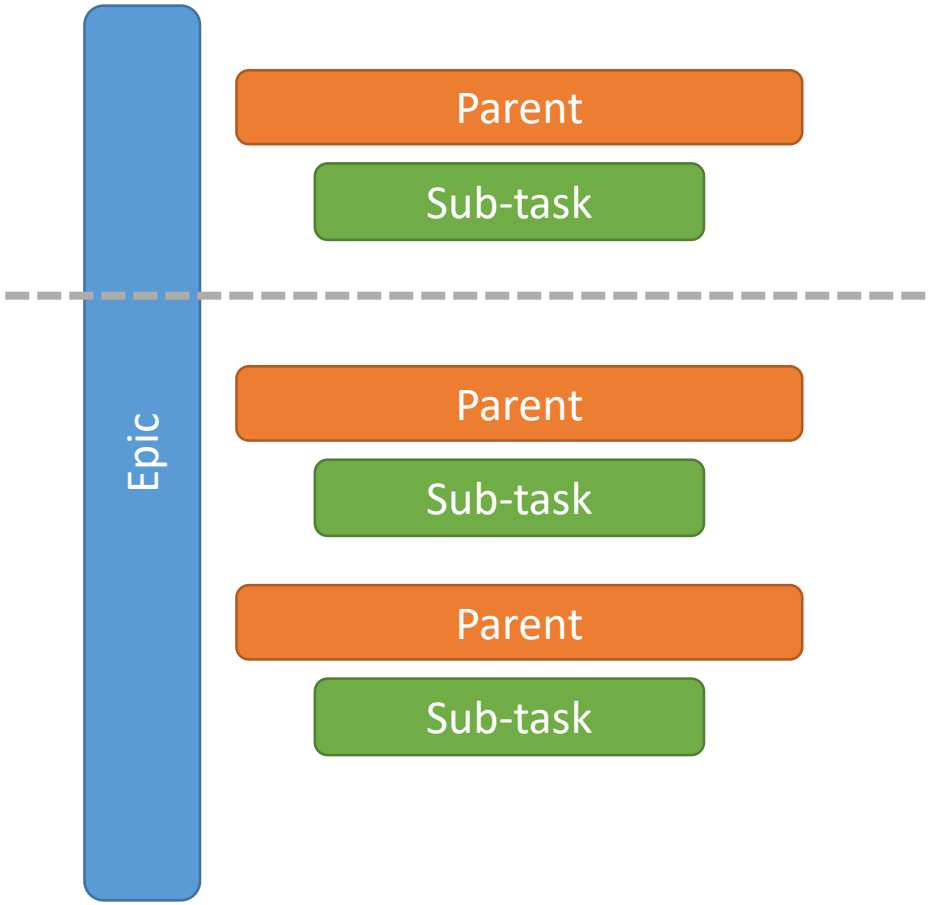
- Get a shared understanding of the terminology
- Standardize your processes and setup within the tool you're using
- Standardize a way to build the backlog and ensure a shared understanding

Understanding Epics



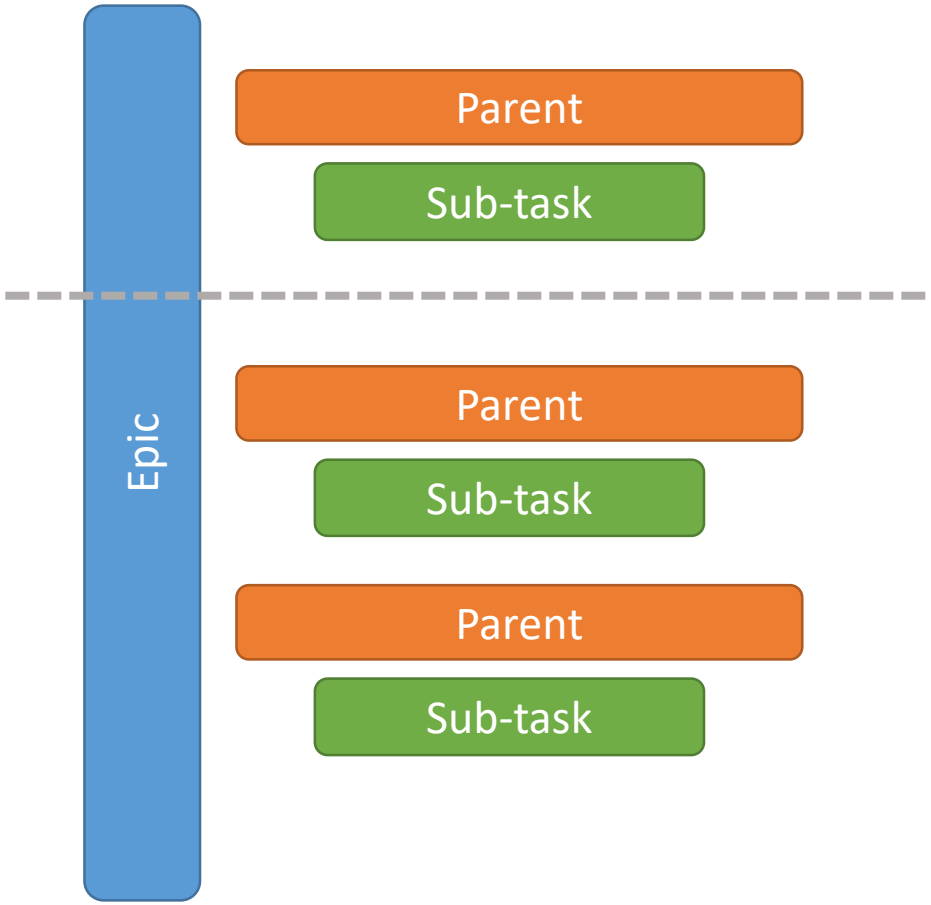
- Epics represent buckets of related work that can be completed; not simply an eternally-existing container of issues.
- Generally, Epics become necessary when a group of related stories are not accomplishable within a single sprint.
- Epics themselves are not added to a sprint, rather, the stories/issues within the epic are what get added to the sprint.
- Epics themselves are not estimated; they receive their total estimate from the sum of the issues within them

Understanding Parent Tasks



- Parent tasks can be configured to be whatever is preferred. Ex. Story, Task, Etc.
- Parent issues may have sub-tasks
- A Parent issue's estimate is always the sum of it's Sub-tasks, if they exist
- Parent issues are the only type (of the three) that receive story-point estimates

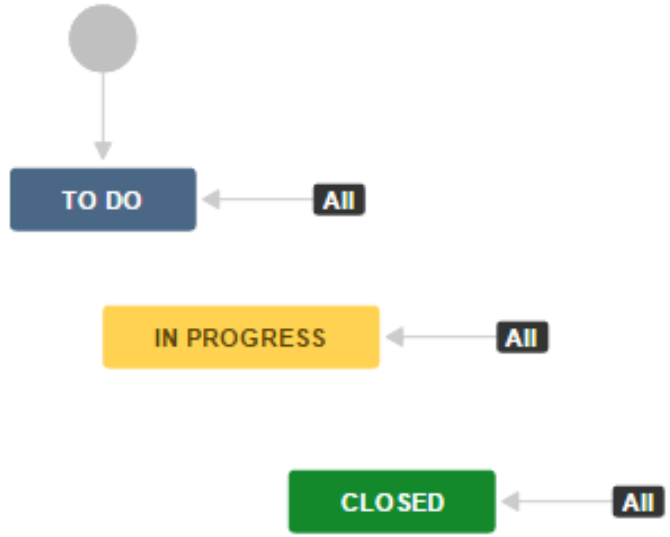
Understanding Sub-tasks



- Sub-tasks types can be configured to be whatever is preferred
- Generally, Sub-tasks represent a check-list of what is necessary to accomplish the parent
- Sub-tasks never not receive story point estimates (the Parent issues does).

Workflow Pro-tips

- Ensure workflows facilitate your end-to-end process
- Try to design workflows to be as cross-functional as possible
- Avoid adding complexity when you can do without it – the simpler, the better
- When building workflows, often ask “Will we need to report on this?” if not, special workflow capabilities or fancy task types are generally not necessary
- Do your best as an organization to standardize (at least at a foundational level) the workflows across all teams



Step 1: Create Epics

- **Involve the whole team in this process**
- Ask, “What’s a logical way to break down the project into large bodies of work?” For example, perhaps and epic titled “Home Page” or “Develop API” would be a good start for your first epic
- If the project/initiative is short-term, you may use a single epic
- You don’t need to be perfect, additional epics may become necessary as you breakdown the individual issues

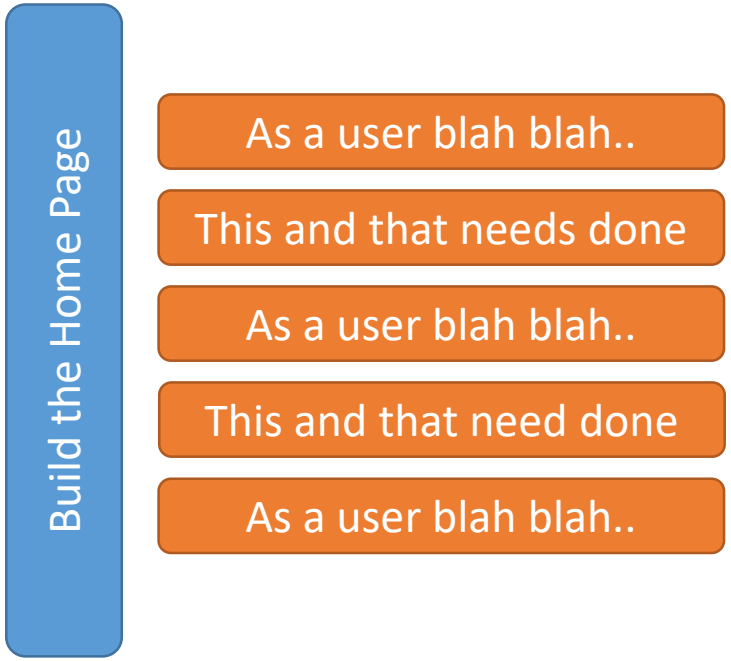
What it looks like… (Epics)

- Build the Home page
- Build the Contact Us page
- Build the About Us page
- Build the API

Step 2: Create Stories / Tasks

- **Involve the whole team in this process**
- Work to break down the new body of work in a logical way that is conducive to development and QA processes (this will take significant amounts of time, but it pays off)
- Typically, if story is going to take longer than (3) days, it needs broken down into smaller, additional issues
- Work overtime to settle on a standardized way of creating stories as a team – master this, you'll master scrum (But, that's a topic for another day)
- At a minimum, have the Story's summaries complete; you can assign the BA's (or others add content, descriptions, specific requirements, and/or acceptance criteria later

What it looks like.. (Stories/Tasks)



Your walls are covered in spaghetti

Your estimations don’t take into account variable change and at the end of the day are just an individuals throwing things at the wall and hoping they stick.

Like Kevin Spacy says in “21”,
“Always account for variable change.”

How to Fix It

- Use Story Points for project estimates
- Use Hours Estimates when sprint planning

Story Points

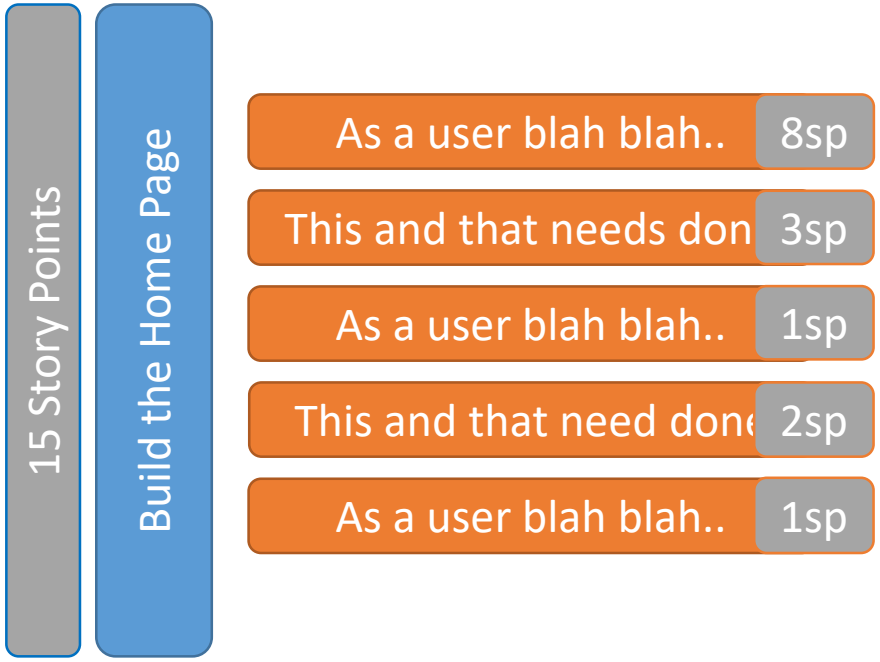
- A Story Point is a relative measurement of complexity given to a Story (in the backlog) and has no relation to time
- Story Points are team estimates, rather than individual estimates – which allows for a **team** velocity to be established (think wood chipper)
- Only when the a Story is entirely complete are the Story Points “counted” towards Sprint Velocity

Step 3: Story Point Your Backlog

- **Involve the whole team**
- Have a sequence defined for all teams (typically Fibonacci) 1, 2, 3, 5, 8, 12, 20…
- Review each story in the backlog together, discussing (in detail) the requirements of each
- When first starting – determine a relatively simple task that everyone has done or is familiar with – call this the “2”. Now, compare all other numbers to the complexity of this task. For example, if adding content to a modal is a 2, then a 5 is a little more than twice as complex as adding content to a modal
- Play rounds of planning poker to determine a story point value, only settling on a story point value when all team members are within two story-cards from each other

Planning Poker Cards

What it looks like.. (Story Points)



If the team has a (2) week sprint cycle and average number of Story Points completed is (5), about how long will it take to build the home page?

3 Sprints / 6 Weeks

More things about Story Points

- Story Point velocity represents the amount of new development you can complete in a sprint therefore, only Stories (value add development items) are assigned story points
- Production Bugs, Technical Debt, and other overhead project tasks should NOT receive story points

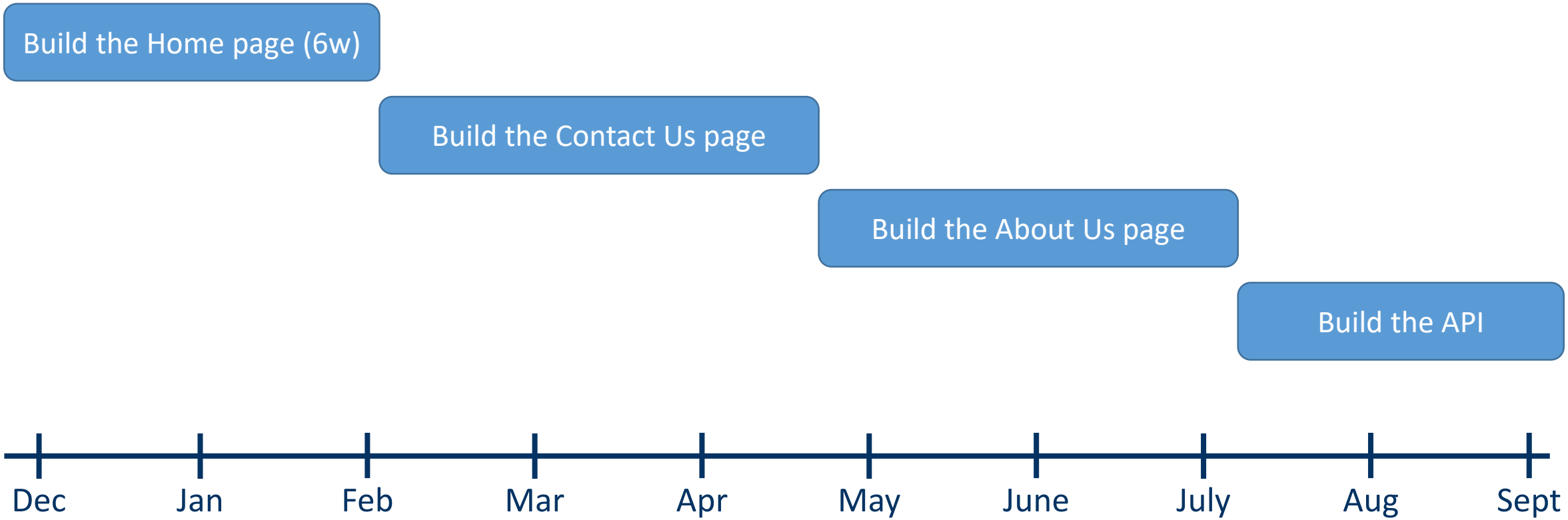
Story Point Velocity

Value Add Products

***Velocity is new development only!**

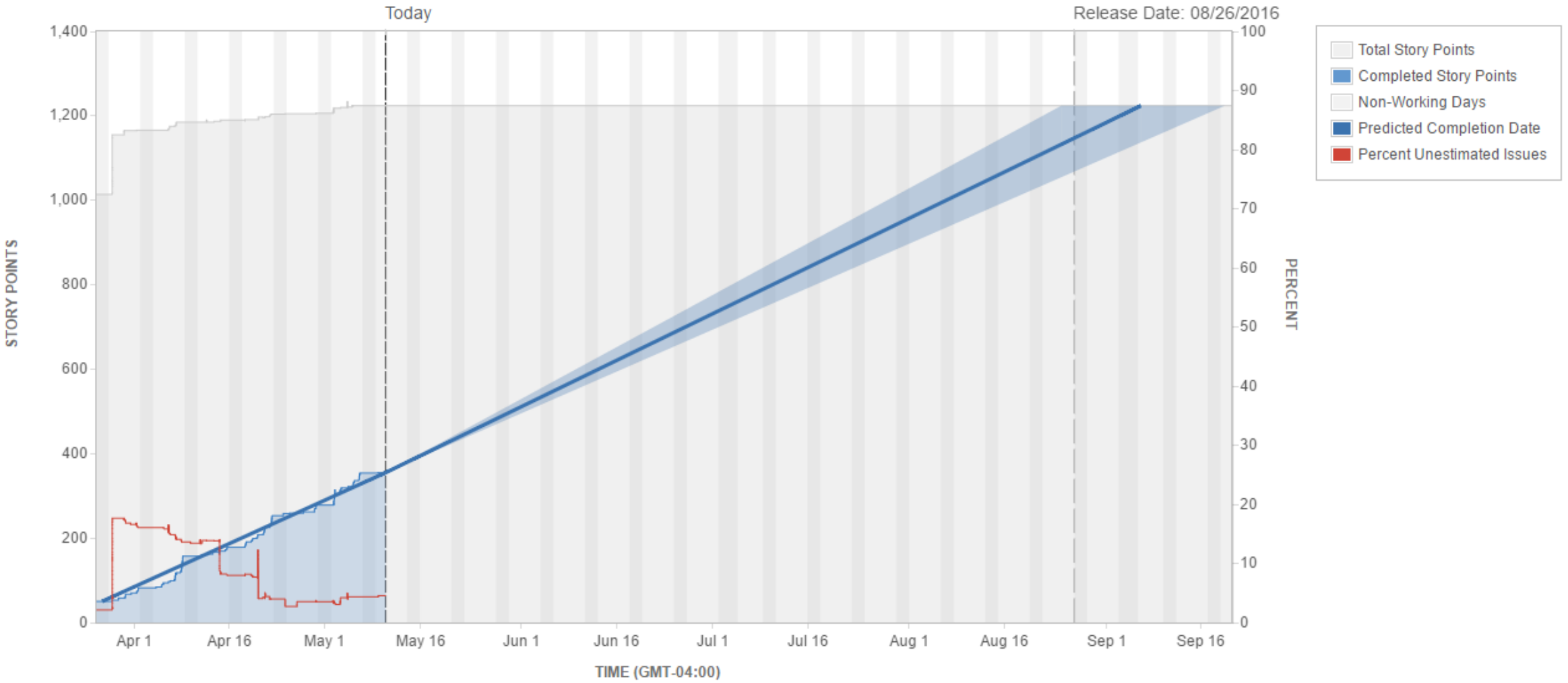
Wood-chipper RPM = Sprint/Team Velocity

Now look what we can do..

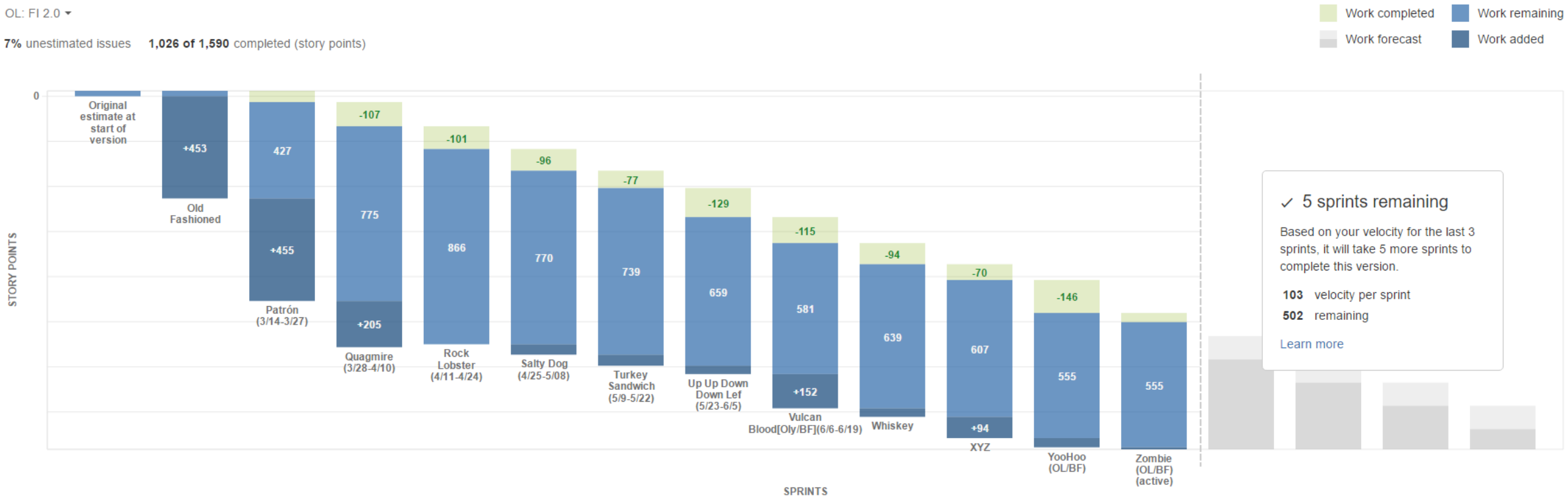


Version Report

Version Report



Sprint Burndown



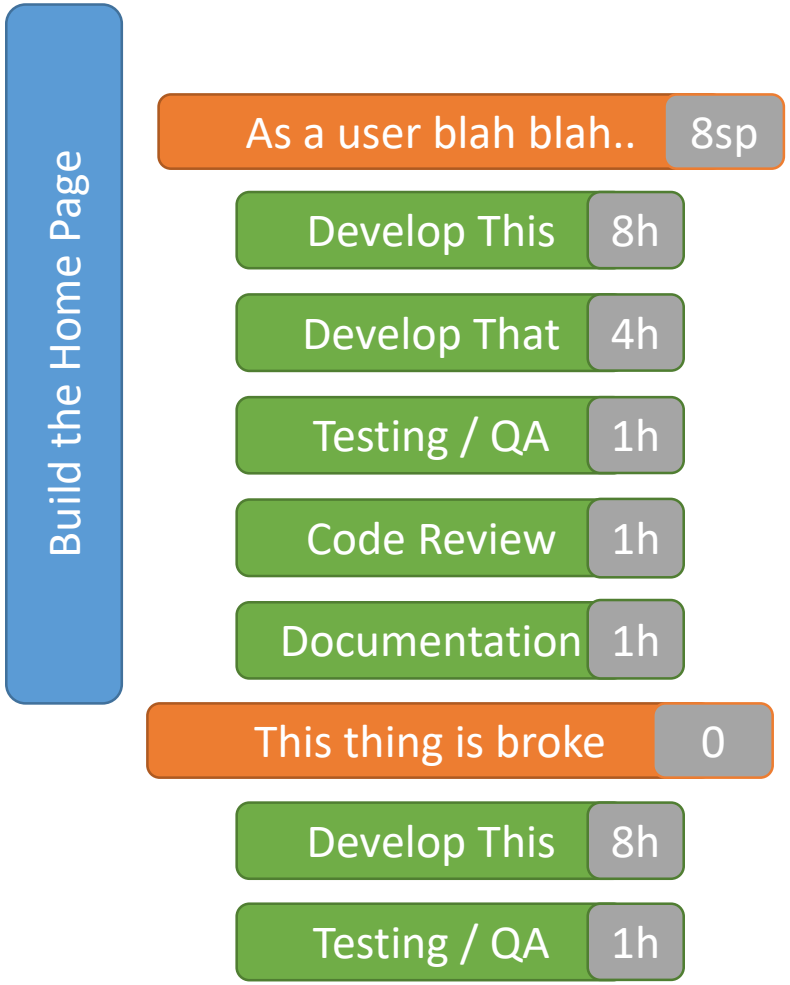
Empiricism

Empiricism asserts that knowledge comes from experience and making decisions based on what is known. Scrum employs an iterative, incremental approach to optimize predictability and control risk.

Your Sprint Planning Meeting

- Take a whole, it helps keep people focused
- The teams available capacity (in Hours) is calculated
- The team determines what should be worked on
- Issues from the backlog are added to the sprint
- Issues are assigned to individuals and hour estimates are added to each story (or sub-tasks)
- Once each individual has reached his/her hourly capacity, he/she is considered “ready”
- Each team member completes this process one by one with the group until the whole team’s capacity is filled
- Everything in the sprint should receive an hour estimate

What it looks like.. (hours)



Capacity Planning

- Don’t rush to plan sprint capacity using Story Points
- To start, teams need to plan to an hourly capacity (since a SP velocity is not yet determined)
- An individual’s capacity for a sprint is calculated simply by adding the total number of working days in the sprint, minus any days off, multiplied by hours in a workday, multiplied by a working capacity percentage.

For example:

(10 Working Days) x (8 hours per day) = 80 Hours

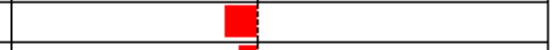
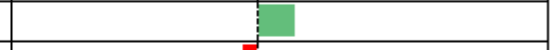
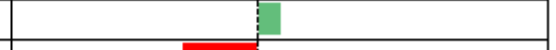
(80 Hours) x (.65) = 52 Hours

Individual available capacity = 52 Hours

Capacity Percentage

- Nobody works on new development tasks 100% of their working time
- 65% for most roles is a good percentage to start with
- The remaining 35% is for
  - Meetings / Planning
  - Other tasks not related to the project
  - Misestimates
  - Bathroom Breaks
  - Trolling Reddit
- This percentage can be changed when necessary – you'll learn to perfect this incrementally as the project continues
- Avoid having individual resources allotted their own percentage – the team should all be using the same capacity when possible

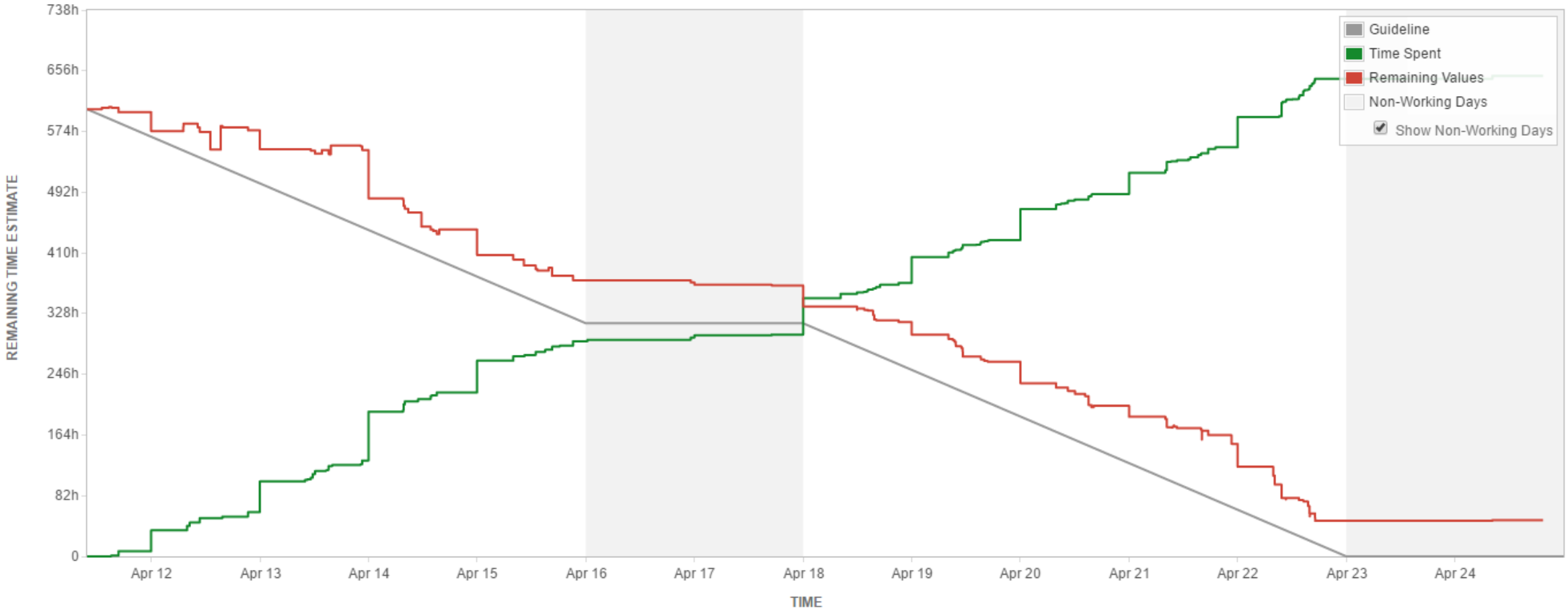
Capacity Planner

	Resource / Assignee	Iteration Capacity	Availability	Usable Capacity	Scheduled Effort	Unused Capacity (Hours)	
Web Team	**Phil**	10 days	65%	52.00 h	0.00 h	52.00 h	
	Adar	9 days	65%	46.80 h	54.00 h	-7.20 h	
	Brian	10 days	65%	52.00 h	56.00 h	-4.00 h	
	Mike	10 days	65%	52.00 h	52.00 h	0.00 h	
	Matt	10 days	65%	52.00 h	44.00 h	8.00 h	
	Davi	10 days	65%	52.00 h	55.00 h	-3.00 h	
	Justi	10 days	65%	52.00 h	48.00 h	4.00 h	
	Devi	10 days	65%	52.00 h	28.00 h	24.00 h	
	~~Ali M~~	~~0 days~~	~~65%~~	~~0.00 h~~	~~54.00 h~~	-54.00 h	
	Phili	10 days	65%	52.00 h	52.00 h	0.00 h	
	Xiu M	4 days	65%	20.80 h	16.00 h	4.80 h	
	Justi	8 days	65%	41.60 h	58.00 h	-16.40 h	
	Jorda	10 days	65%	52.00 h	33.00 h	19.00 h	
	Fred	10 days	65%	52.00 h	36.00 h	16.00 h	

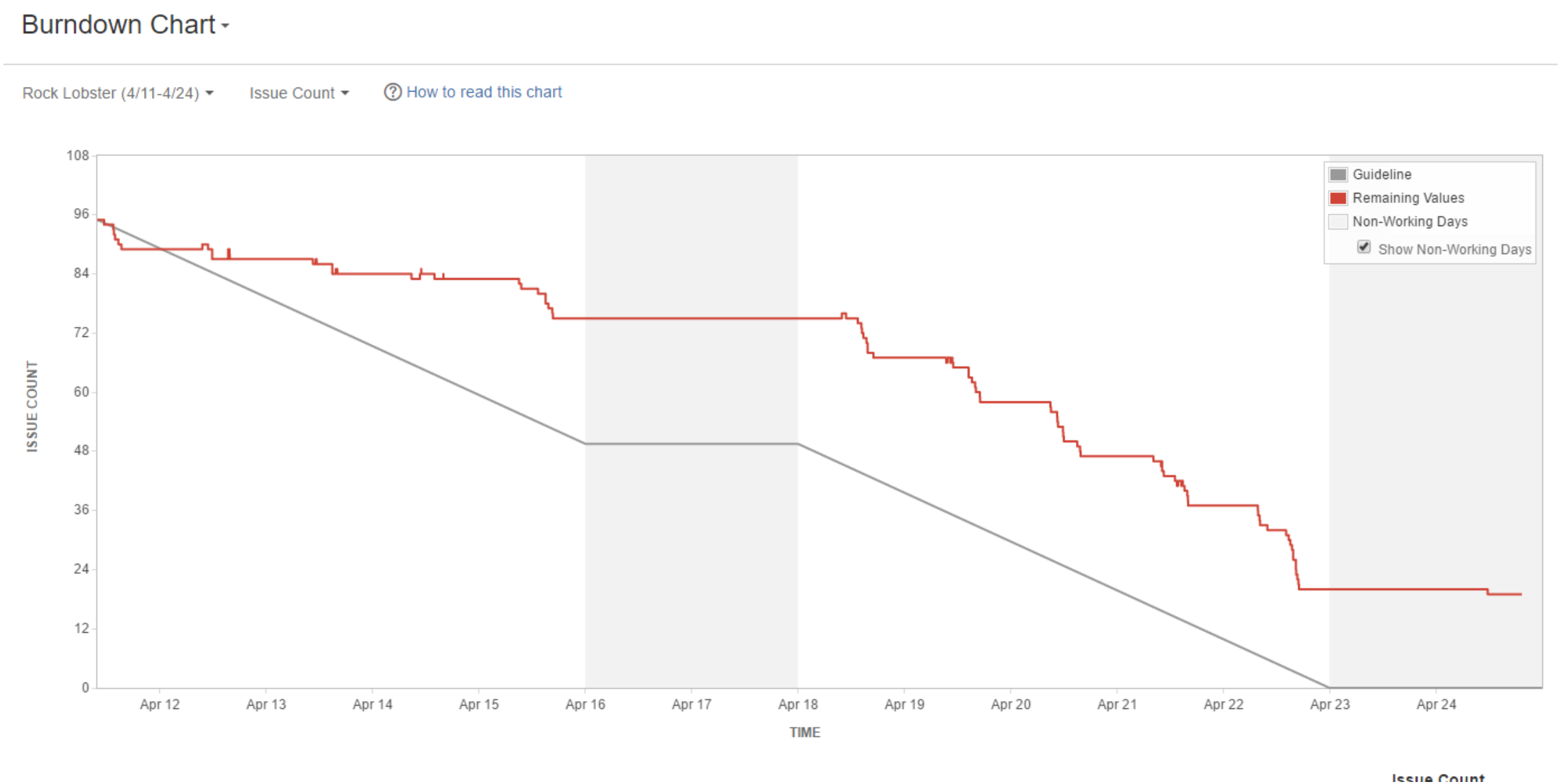
Burndown (Hours)

Burndown Chart

Rock Lobster (4/11-4/24) ▾ Remaining Time Estimate ▾ How to read this chart



Burndown (Story Points)



Dashboards

Sprint Health Gadget

Overall sprint progress (Story Points) 6 days left

63 | 108 | 55

25 %	24 %	0%	0	0
Time elapsed	Work complete	Scope change	Blocker	Flagged

Issue Statistics: Olympus Active Sprint (Status)

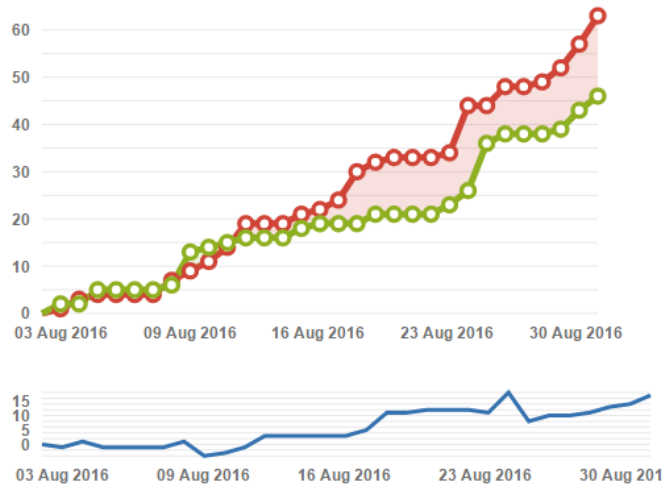
Status	Count	Percentage
OPEN	41	36%
IN PROGRESS	23	20%
IN QA	11	10%
CLOSED	11	10%
CODE REVIEW	25	22%
BLOCKED	4	3%
Total	**115**	

Sprint Burndown Gadget

Scrum - Olympus

SPRINT: Berlin (OL/BF)

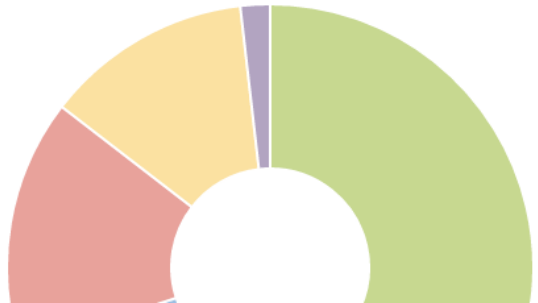
Created vs. Resolved Chart: Olympus Defects



Issues in the last 30 days (grouped daily) View in Issue Navigator

- Created issues (63)
- Resolved issues (46)

Pie Chart: Olympus Defect Backlog



Activity Stream

Activity Stream

Today

Samuel Wheat resolved OL-3492 - Customer > Edit Customer > Social Media Integration

3 minutes ago Comment Watch

Samuel Wheat resolved OL-3948 - Error appears behind modal when no email address is entered

3 minutes ago Comment Watch

Samuel Wheat updated 2 fields of OL-3948 - Error appears behind modal when no email address is entered

- Changed the Remaining Estimate to '0h'
- Logged '0.5h'

5 minutes ago Comment Watch

Mandy Lawman commented on ~~OL-3763~~ - Platform > Recents modal doesn't work outside sales module

done

11 minutes ago Comment Watch

Mandy Lawman closed ~~OL-3763~~ - Platform > Recents modal doesn't work outside sales module

11 minutes ago Comment Watch

Steve Matthews removed the Flagged of OL-1909 - Deal> Search> Filter Deals by Employee Dropdown Should Be Limited By Role

13 minutes ago Comment Watch

Steve Matthews changed the Assignee to 'Freddy Henin' on ~~OL-1724~~ - Platform > Notifications > Clear All notifications

14 minutes ago Comment Watch

Steve Matthews changed the status to Code Review on ~~OL-1724~~ - Platform > Notifications > Clear All notifications with a resolution of 'Done'

they are now all being cleared. the grouping icon is removed, and refreshing the page won't bring them back. There's no confirmation but that wasn't part of this story.

14 minutes ago Comment Watch

Honestly.. your sprints are kind of slutty

Your sprints are not really a closed commitment to the work that is originally planned.

Meeting the sprint goal should be the primary focus of the whole team because it’s the only way you stay on track.

How to Fix It

- Don’t remove things from the sprint before it’s over because you won’t get them done. You need that data for your retrospective.
- If you’ve finished early, help others before working on things in the backlog
- The sprint goal is do or die (within reason) you work until it’s done
- If your team is overworked in a given sprint, your sprint data should show this and you can adjust so it doesn’t happen again
- If the whole team finishes early, consider sending people home early (unless you’re behind, or prefer to get ahead, or whatever..)

Your confusing this with AA

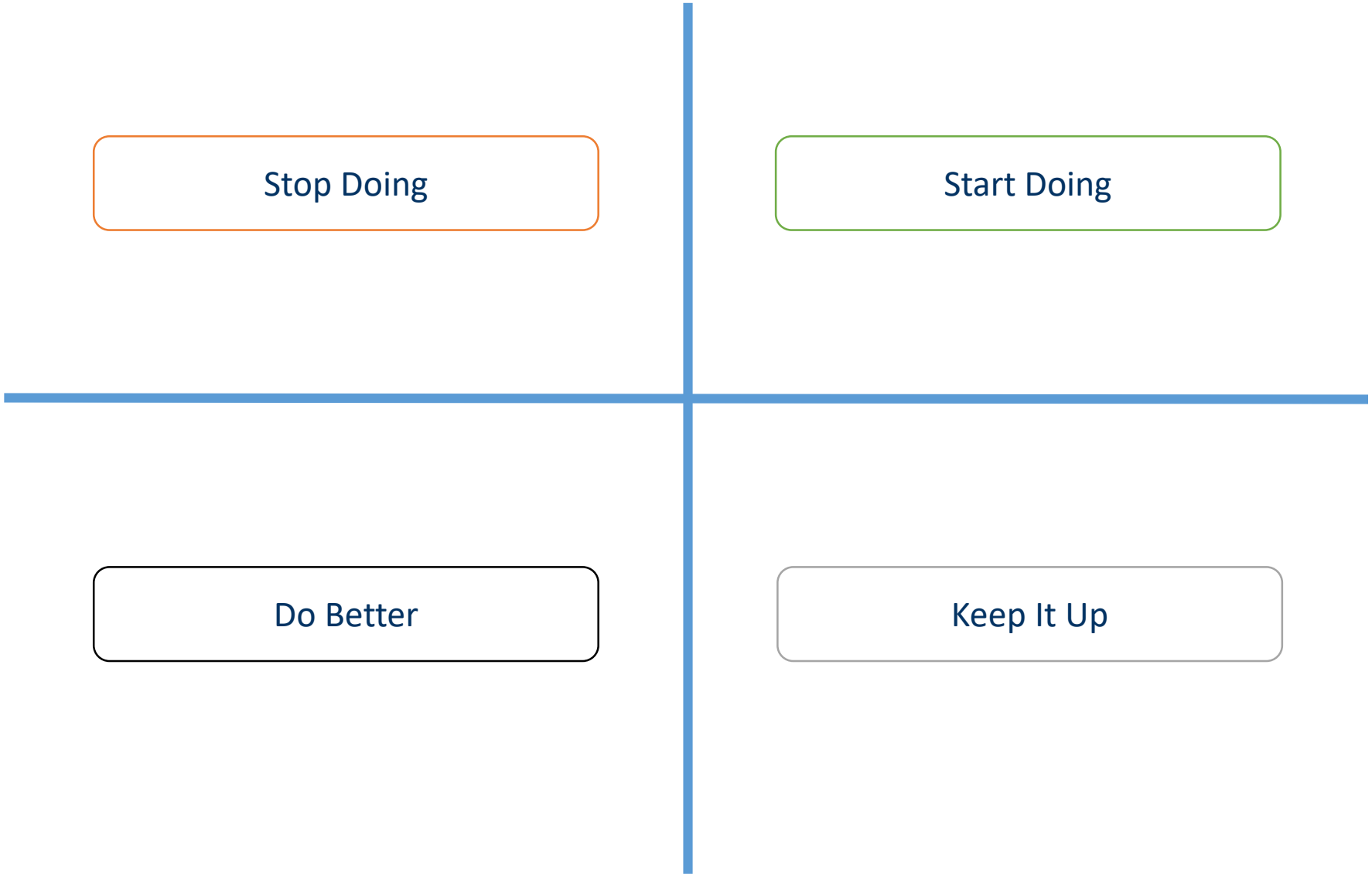
The retrospectives are too often just awkward conversations about someone’s mistake and it make me uncomfortable.

A team analysis of the sprint data is how you learn to improve and if you’re not improving, your team’s not agile.

How to Fix It

- Use sprint reports and talk about what was complete, and what was not
- Look to really understand, as a team, the specifics about why certain things weren't finished and don't stop there, come up with fixes – create tasks for the backlog if necessary
- Analyze burndown charts: How did you hour estimates compare to the time you've logged? (More time logged = under estimate, etc.)
- Analyze how often things stories are closing. This provides insight into if your stories are too big – the shorter the better.

(See? You need the right process in place to get this kind of data..)


Stop Doing
Start Doing
Do Better
Keep It Up

Keep Doing!

- BL Grooming / Poker
- Team Colaboration
- QA's were quick & responsive at sprint end
- Eliminated quite a bit of tech dost
- Planning for next sprint

Do Better!

- Organize cross-used controls
- Need to create "House Rules"
- Requirements within the issue

Stop Doing!

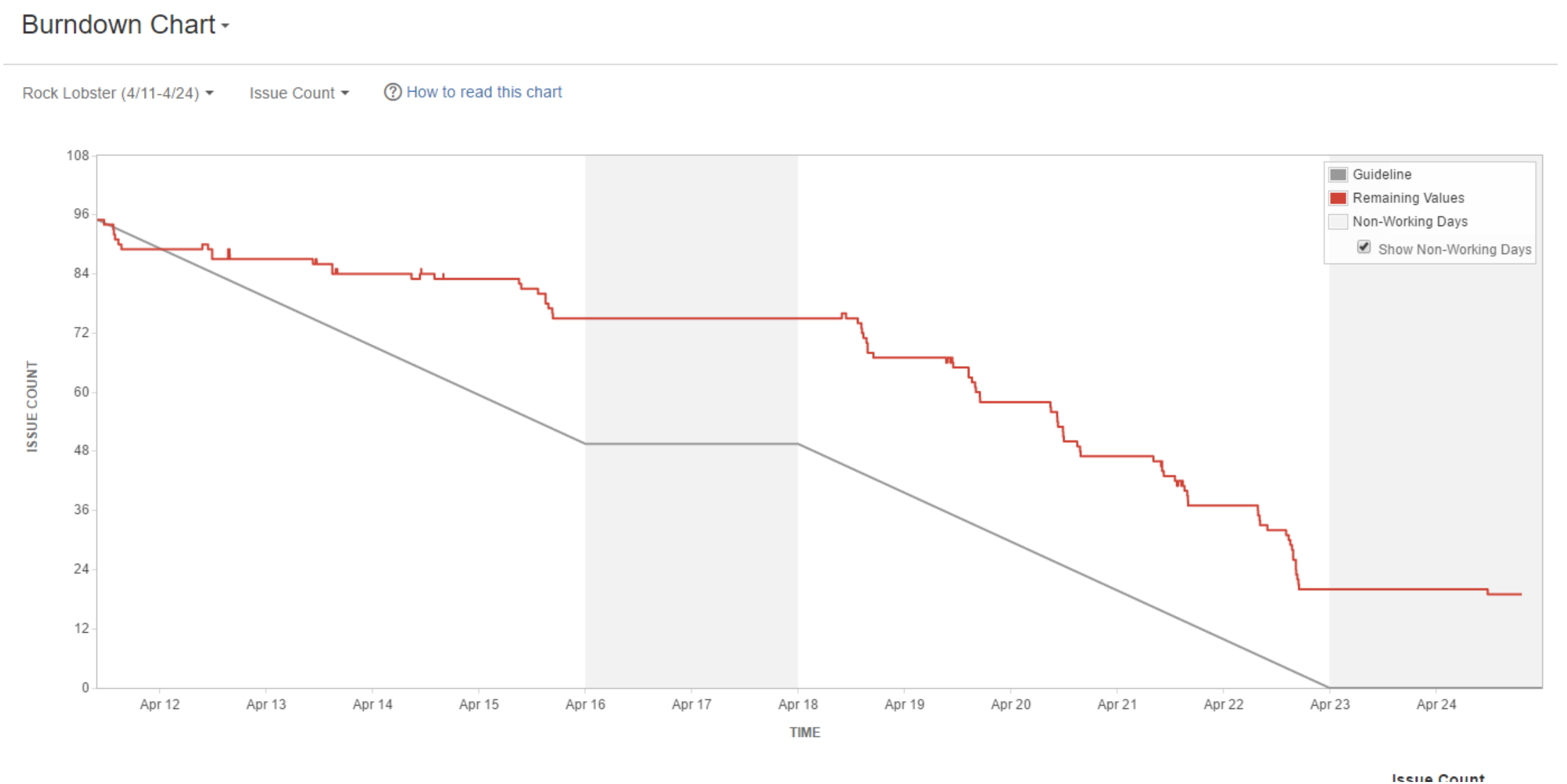
- Adding estimats post sprint-start

Start Doing!

- Create "House Rules" Document
- Use team-dedicated QA servers
- Estimate & Track QA time
- Re-look at B.E. code before start
- More scrum of scrum meetings

Plenty up slack

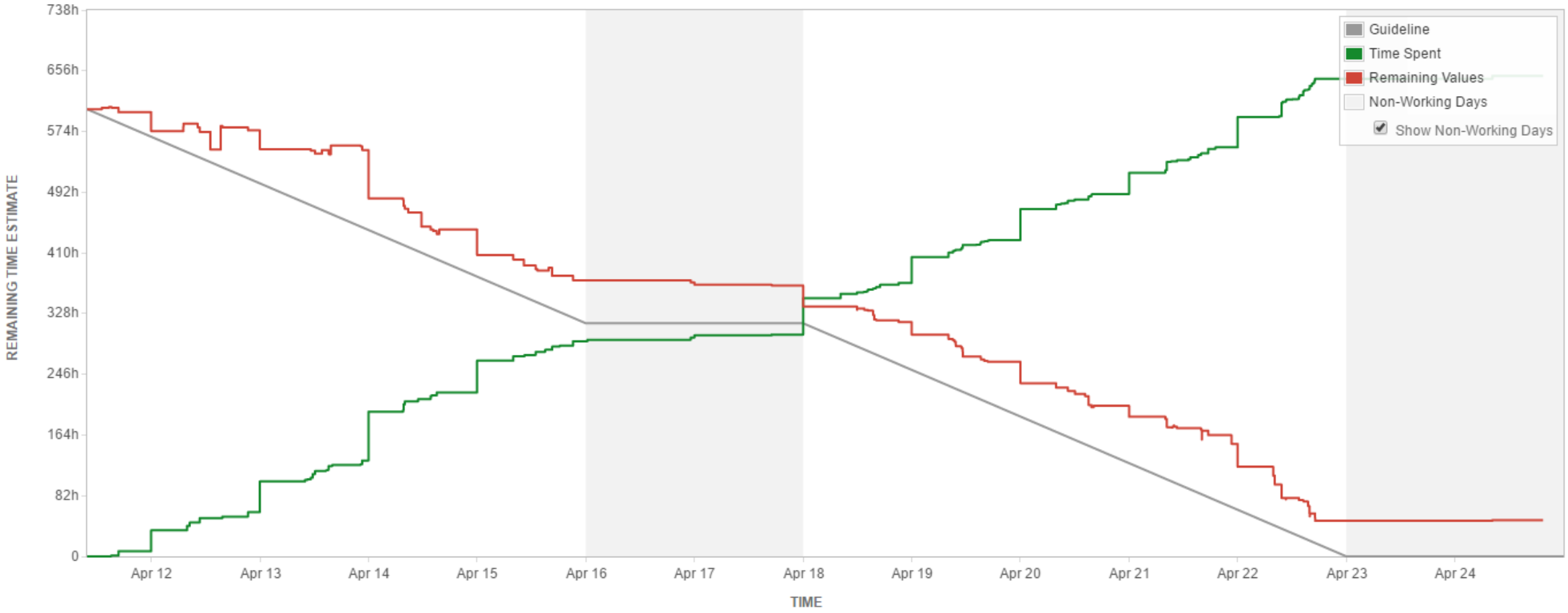
Burndown (Story Points)



Burndown (Hours)

Burndown Chart

Rock Lobster (4/11-4/24) ▾ Remaining Time Estimate ▾ How to read this chart



You didn’t fire Nick

There may be a few strong personalities on the team that are not conducive to effective communication.

A high-performance team likes
each other and communicates often.

A technical genius with a bad attitude is
cultural cancer to the team.

You can easily teach/learn technical skills
to/from someone who’s a cultural fit.

How to Fix It

- Culture / Everything
- Get things into a decent place regarding scrum and then assess if it's still the same
- Address people's negativity one-on-one; technically speaking – this is a task for the Scrum Master, but use good judgement
- When people are being negative for seemingly good reason, don't participate
- Be a model of someone who gets things done and participate in ceremonies – make it the cool thing to do
- Fire people with bad attitudes – The team should be known for not putting up with an unreasonable amount of negativity. Do this regardless of the individual's "irreplaceable" knowledge. Trust me, you'll get by and it's often not as bad as you think.
- Have a work environment that breeds teamwork and positivity


30
Big
Thirty
EXIT


Scrum - Olympus :
Turkey Sandwich (5/9-5/22)
8
DAYS
REMAINING
STORY POINTS
Guideline
Remaining Values
Non-Working Days
May 11
May 13
May 15
Team Gwin: Turkey Sandwich (5/9-5/22)
Team Pifer: Turkey Sandwich (5/9-5/22)
Issues: 61 created and 76 resolved
Period: last 30 days (grouped Daily)

travisa@empyra.com