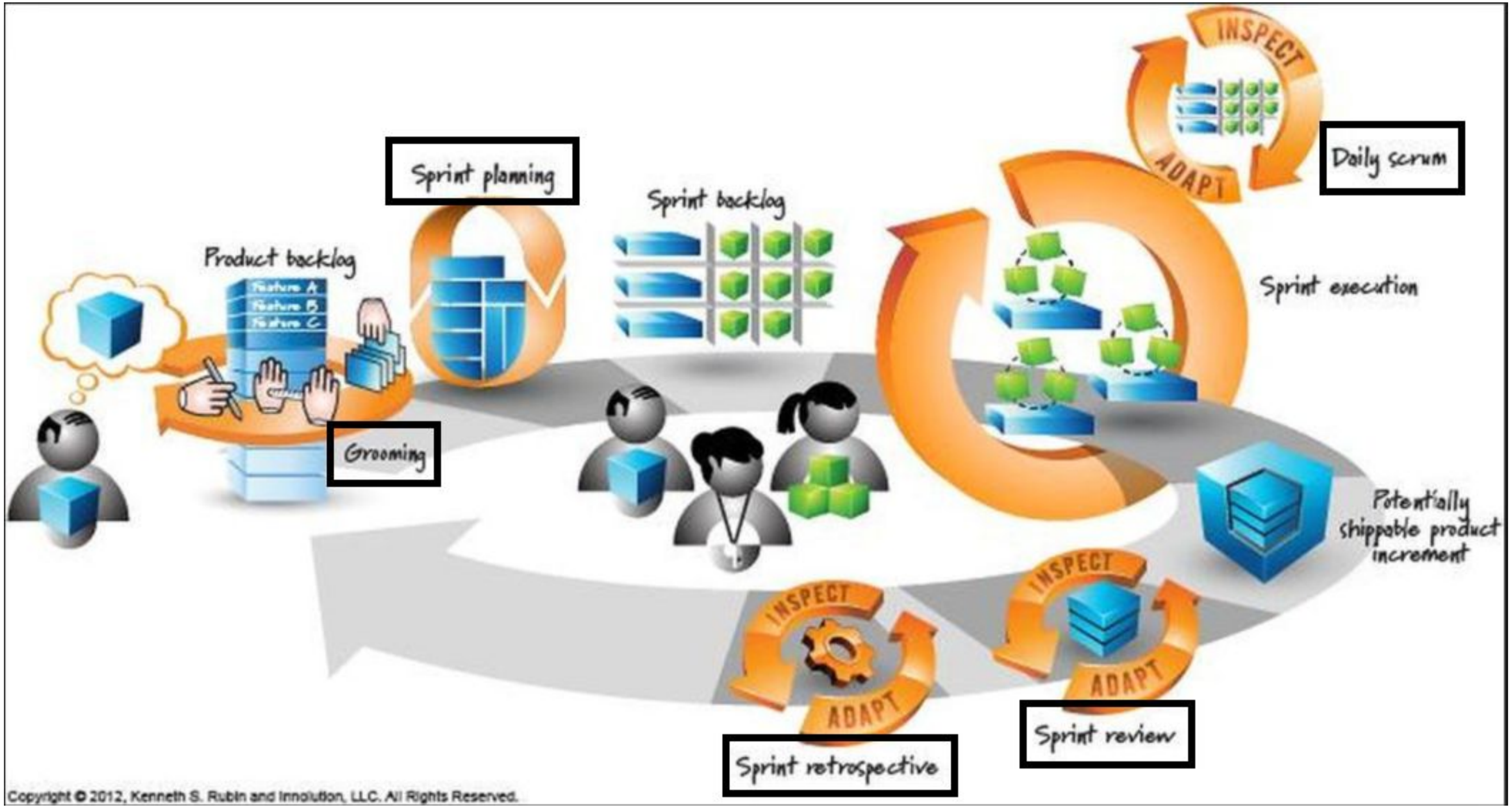
Product Management Process

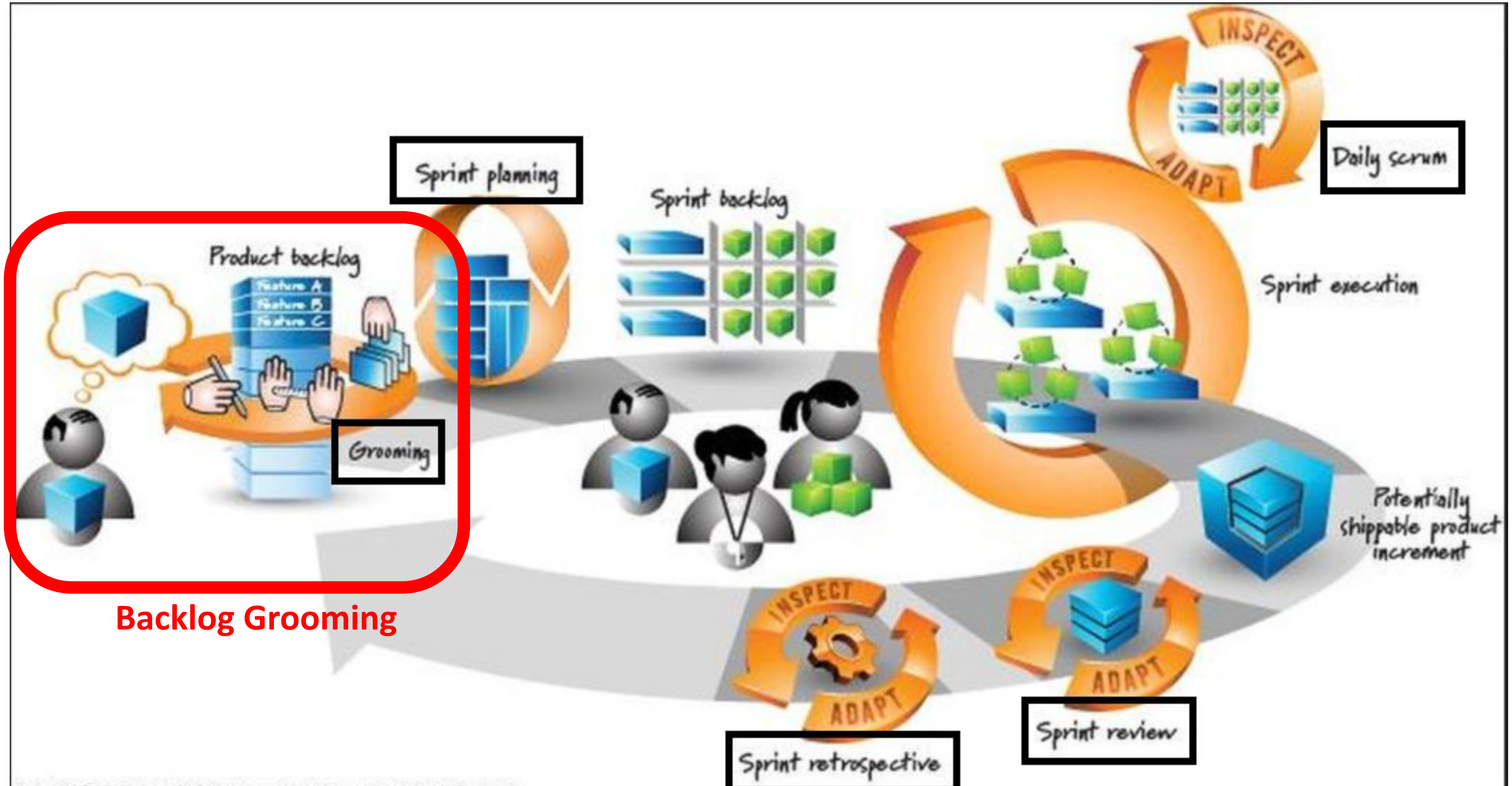
Agility & Ways of Working

- Dr. Ernesto Lee

Overall Scrum Process



Backlog Grooming



Backlog Grooming

Purpose and Context

- Prepare User Stories for sprint planning
- Develop shared understanding of Customer needs as represented by user Stories
- Partner with Product Owner to optimize delivery of business value

Level of Facilitation

- Full-scale facilitation by the Product Owner
- Well-designed, closely facilitated task oriented, outcome driven, time-boxed

Outcomes / Results / Deliverables

- Well Groomed User Stories Include:
  - Business Acceptance Criteria
  - Understand how to demo
- Understand how to Test
- Understand / Extend Definition of Ready and Done for the Stories

Tools & Techniques

- Triangulation
- Powerful Questions
- Listening
- ATDD (Acceptance Test Driven Development)

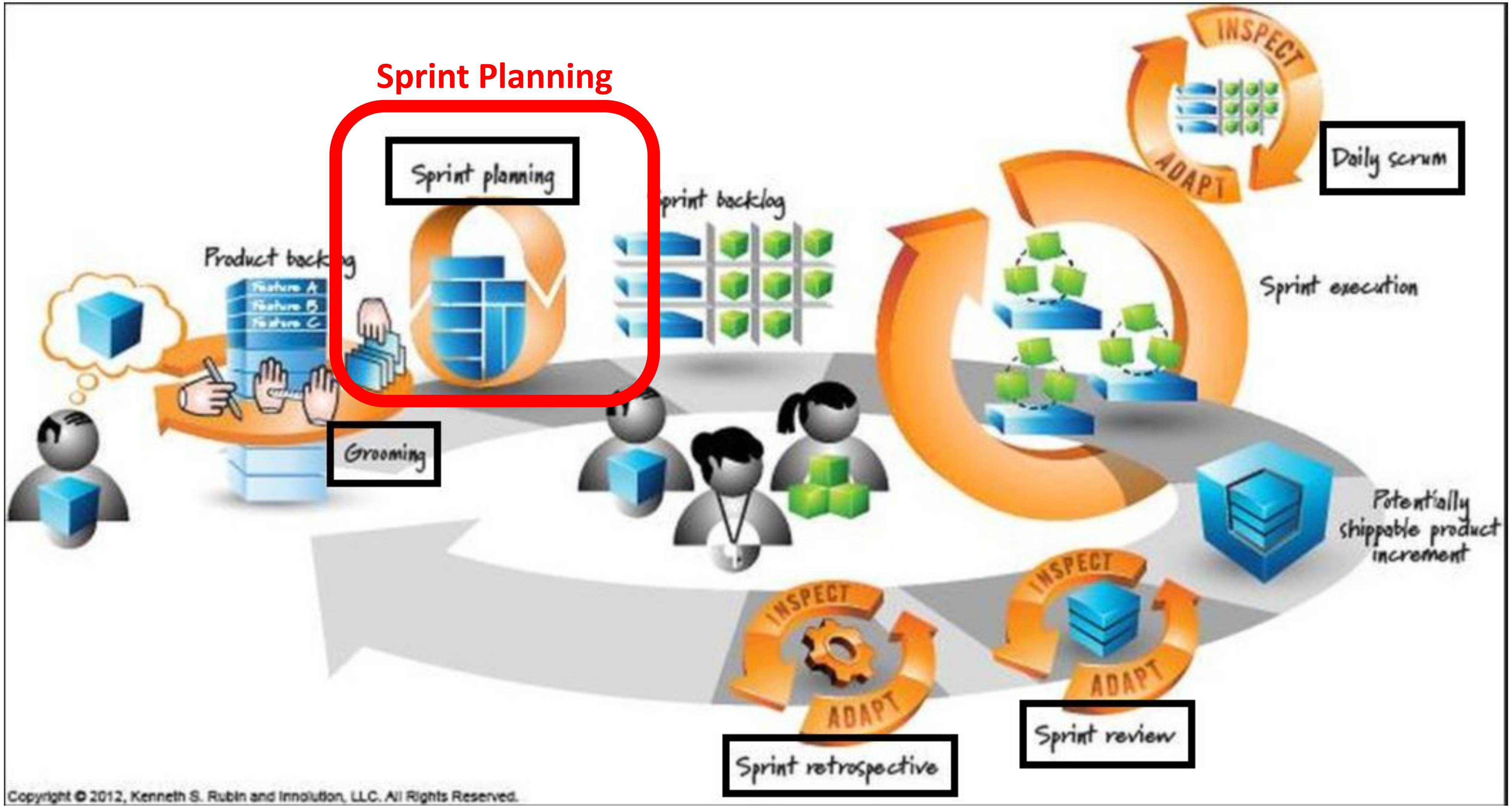
Success Factors

- Engaging & authentic
- Stories moving towards Definition of Ready
- Encourage conversation
- Transparency/Safety

Pitfalls

- Not actually done
- Going thru the motions
- Fear
- Insufficient support from the business stakeholders

Sprint Planning



Sprint Planning

Purpose and Context

- Aligns team and Product Owner on work plan / commitment
- Management gets transparency into the team's thinking instead of needing a detailed long range plan
- Helps team focus and control their destiny
- Provides empirical measure of teams capability / capacity

Level of Facilitation

- Full-scale facilitation by the Scrum Master with Product Owner
- Well-designed, closely facilitated task oriented, outcome driven, time-boxed

Outcomes / Results / Deliverables

- Defined Iteration Goals
- Iteration Backlog with acceptance criteria
- Team iteration Commitment
- Deeper shared understanding
- User Story Task Breakdown
- User Story Size Estimates

Tools & Techniques

- User Story Grooming
- Silent Work
- Brainstorming
- Story Refactoring
- Task Bucket Estimating
- Time Boxing
- Working Edge Cases First
- Just Enough / Just In Time

Success Factors

- Highly Engaging
- Team Alignment
- Clear desired outcome

Pitfalls

- Ungroomed user Stories
- Wishful Thinking
- Undue Influence from senior team members

Daily Scrum

INSPECT
ADAPT
Daily scrum
Sprint planning
Sprint backlog
Product backlog
Feature A
Feature B
Feature C
Grooming
Sprint execution
Daily Scrum
Potentially shippable product increment
INSPECT
ADAPT
INSPECT
ADAPT
Sprint review
Sprint retrospective

Daily Scrum

Purpose and Context

- Helps Team coordinate their actions in detail
- Allows micro-planning and emergent discoveries
- Daily team commitment meeting (Tasks <1 day)
- Provides visibility into team's process

Outcomes / Results / Deliverables

- Team coordination
- Early identification of impediments
- Visibility for non- team members
- Team mutual accountability

Tools & Techniques

- Time Boxing
- Bottom lining
- Just Enough / Just In Time
- "Talking Stick"
- Working the Task Board

Success Factors

- Start on time
- Rigorous Time-boxing
- Commitment based conversation
- Full permission to interrupt

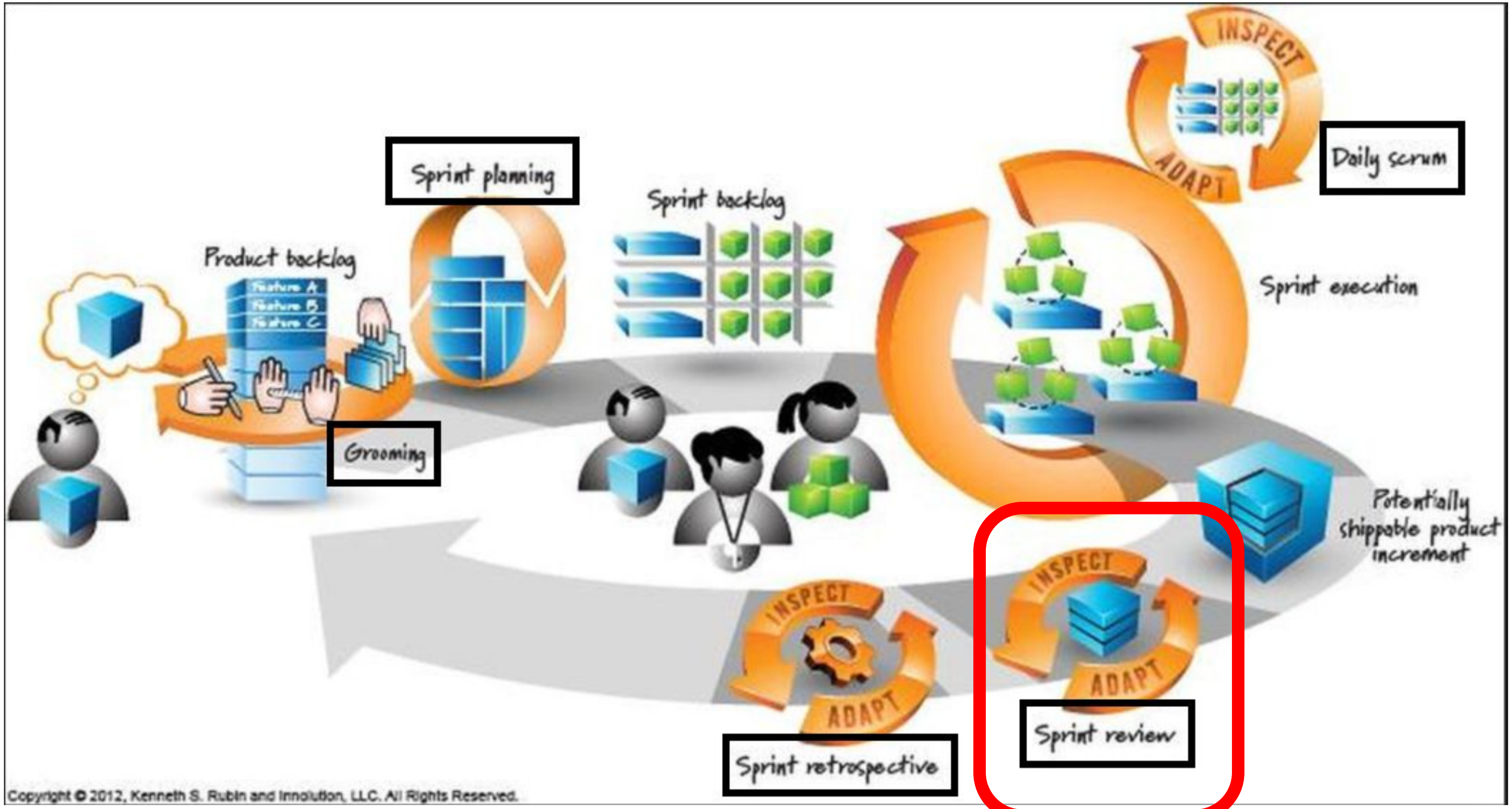
Pitfalls

- Going through the motions
- Digression into details
- Sitting down
- Entire Team not present

Level of Facilitation

- Varies based on team maturity (Shu, Ha, Ri), task oriented, outcome driven, time-boxed
- Facilitated by the Scrum Master and be rotated by members on the team

Sprint Review



Sprint Review

Sprint Review

Purpose and Context

- Review the goal of the sprint and compare to what actually got accomplished during sprint
- Demonstrate progress "show off" the product
- Real-time customer feedback from all stakeholders
- Allows management to give feedback & make requests
- Accept the iteration

Level of Facilitation

- Full-scale facilitation by Scrum Master until a team member wants to, and is capable of, facilitating; then facilitation can rotate.

Outcomes / Results / Deliverables

- Feedback on work completed vs goal
- Demonstrate work completed
- Shared understanding of value delivered
- Improved team morale
- Team takes public responsibility for success/failure (bonds them)

Tools & Techniques

- Time Boxing
- Bottom lining
- Information sharing
- Powerful Questions
- Listening

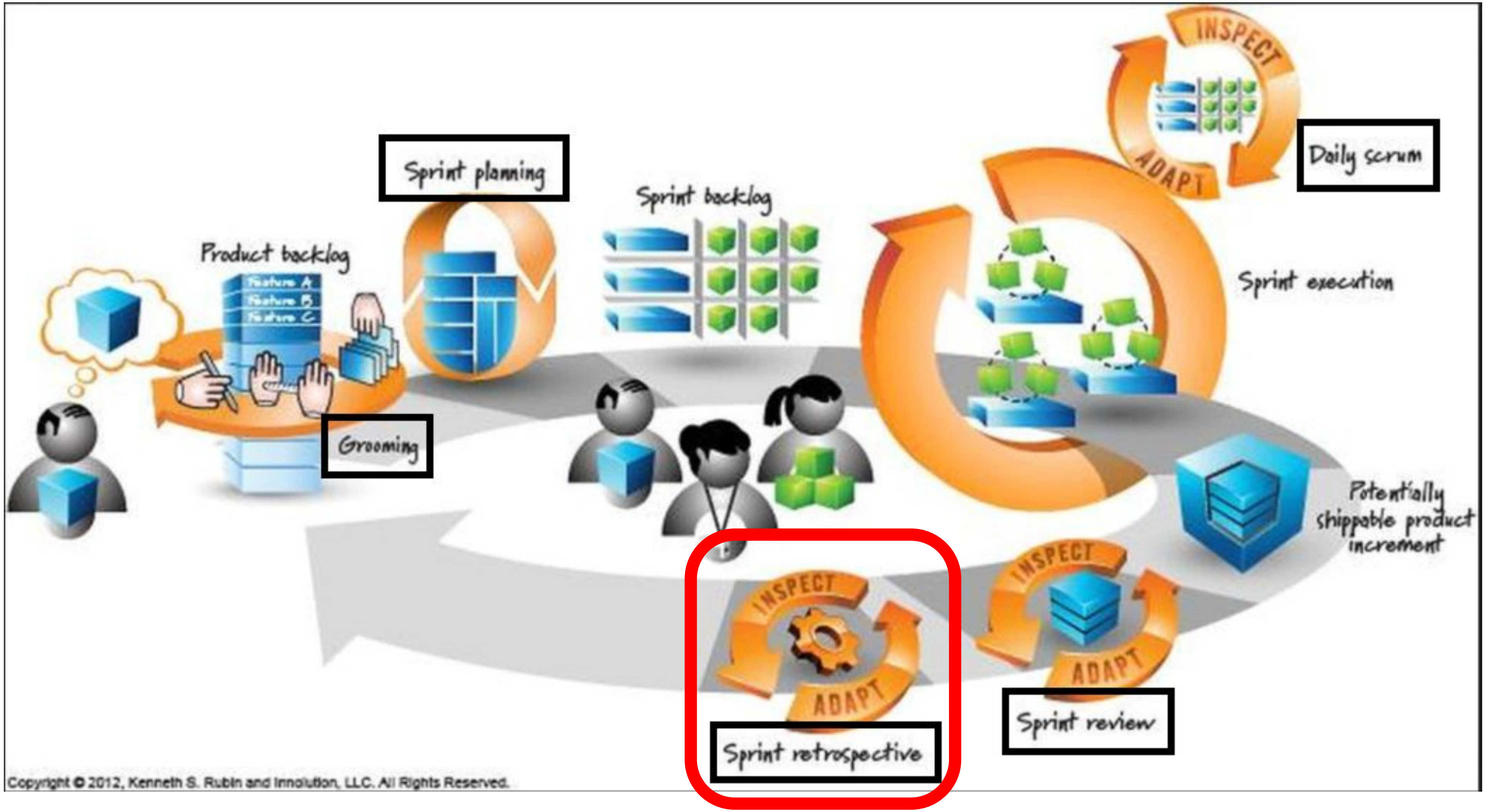
Success Factors

- Engaging & authentic
- "Real" working software
- Encourage conversation
- Transparency/Safety

Pitfalls

- Going through the motions
- PowerPoint
- Fear
- One way information flow
- Lack of interest from business stakeholders

Retrospective



Sprint Retrospective

Retrospective

Purpose and Context

- Primary focal point of the Inspect and Adapt cycle
- The way for the team to take responsibility for their process
- Focuses on three major areas: productivity, process & team

Level of Facilitation

- Full-scale facilitation owned by the Scrum Master until a team member wants to, and is capable of, facilitating; then facilitation can rotate

Outcomes / Results / Deliverables

- Greater team understanding of what actually happened
- Short list (e.g. 1 – 3) of "actions" to improve
- Inspect and Adapt regarding Productivity/process/team
- Team bonding (we're in this together)

Tools & Techniques

- Steps from *Agile Retrospectives* Book
- Meeting is owned by the team, therefore, they may invite non-team members
- Prod Owner may attend IF they are member of team (rather than an external customer)
- Light and frequent follow up is crucial during sprint

Success Factors

- Planned ahead
- Skillfully facilitated
- Team interacts with the team directly
- All voices are heard

Pitfalls

- Same old – same old
- Everything "is outside the team's control" (never completely true)

Team and Scope

The Team

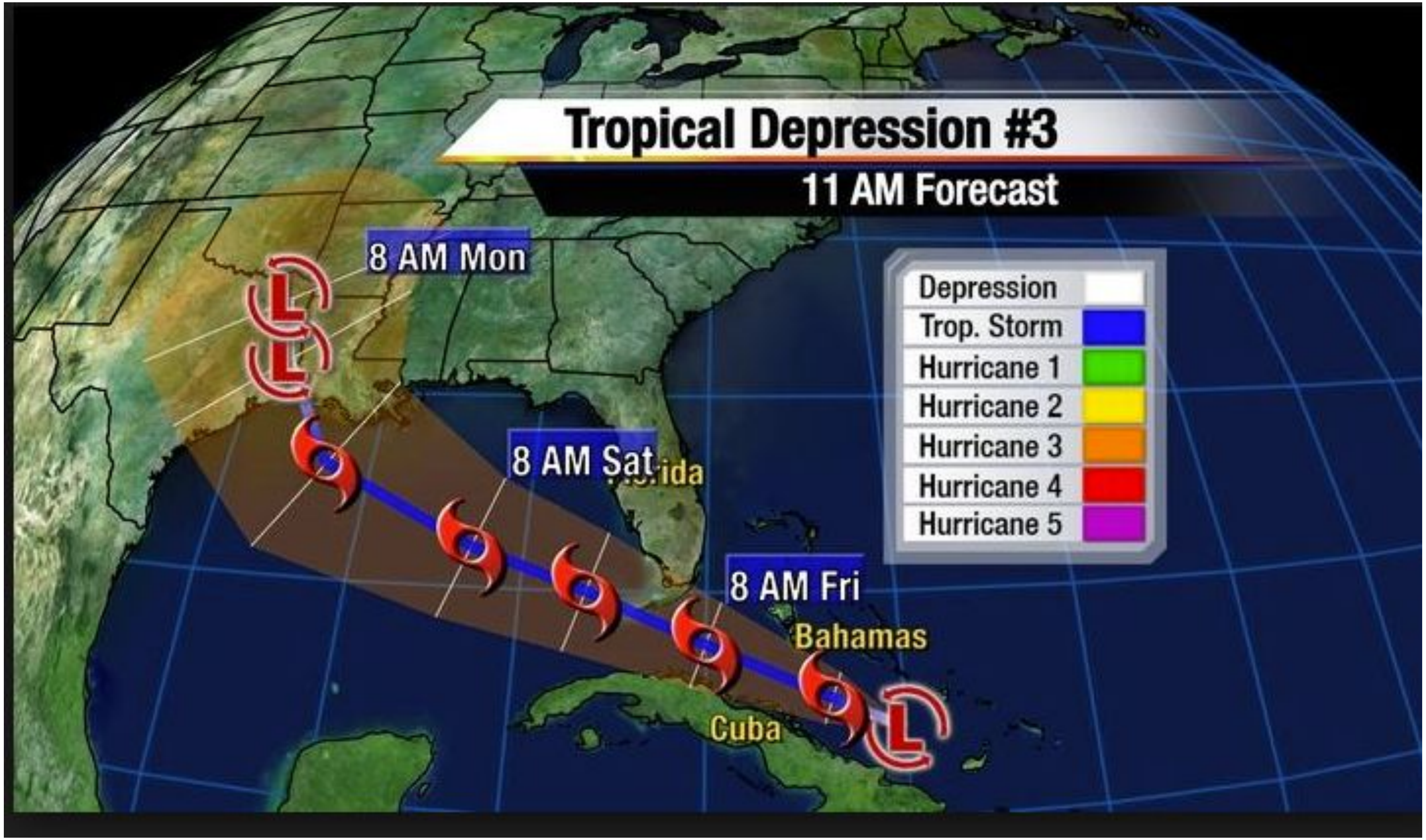
- Roles:
  - Product Manager (PdM)
  - Scrum Master + Team = Development
- All skills needed to implement
  - Generalists, versatile
  - May include specialists
- Egos are put aside
  - High Trust
  - Is it good enough? Not looking for perfection
- Empowered and Self Organized
  - Members accountable for planning and delivery as a team

Team and Scope

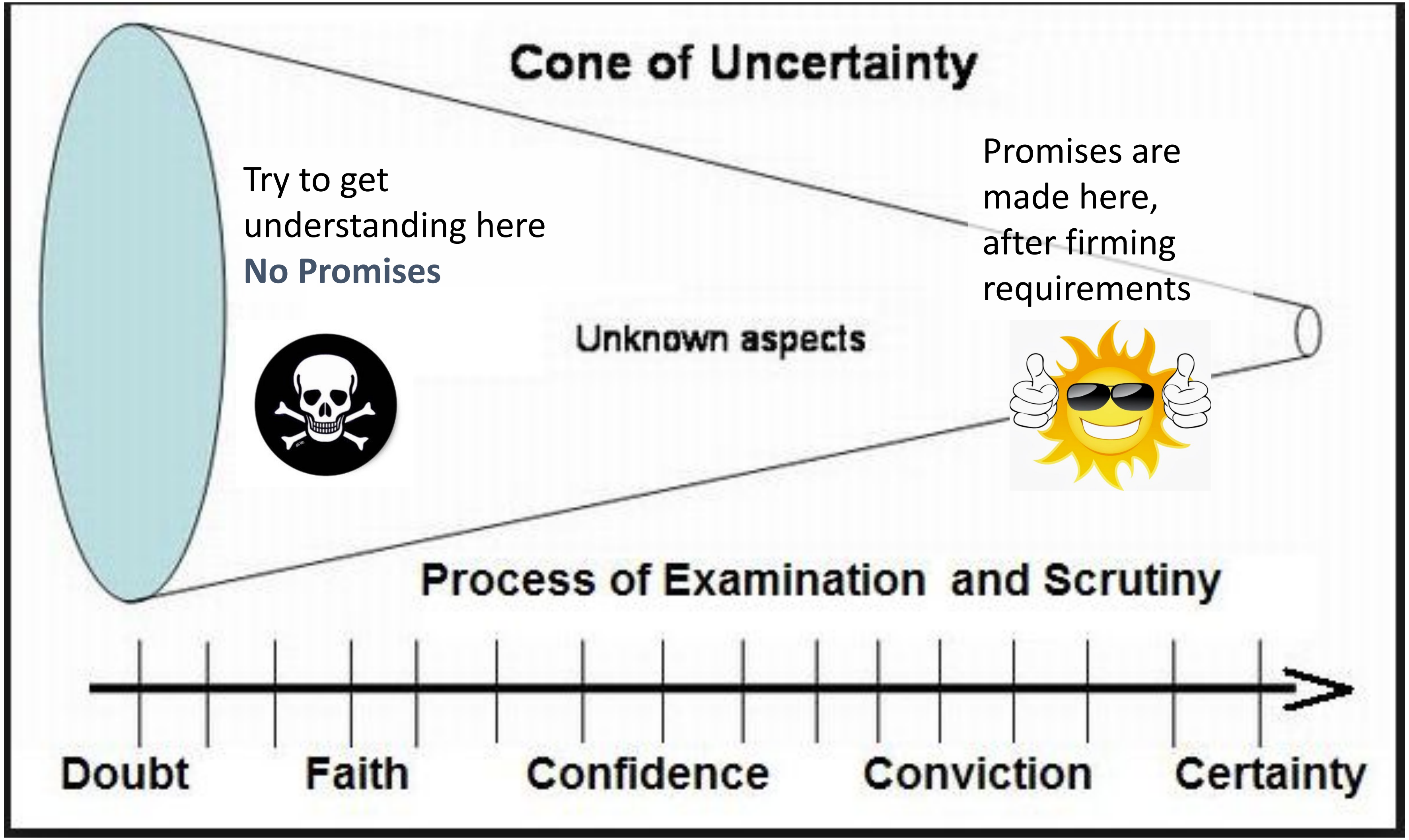
Team Roles

Product Manager	Scrum Master	Development
• Owns Product vision • Defines features, decides on release date and content • Responsible for market success • Prioritizes features according to market value • Can change features and priorities every Sprint • Advocate for Business	• Responsible for facilitating process • Focuses Team and protects them from external interruption • Stabilizes Current Sprint • Looks for ways to enhance productivity • Assists Product Owner in leveraging Scrum • Advocate for the Engineer	• Small group containing all necessary people and project skills • Focuses on steady delivery of high quality features • Generates options for delivery • Manages own work within Sprints

Cone of Uncertainty

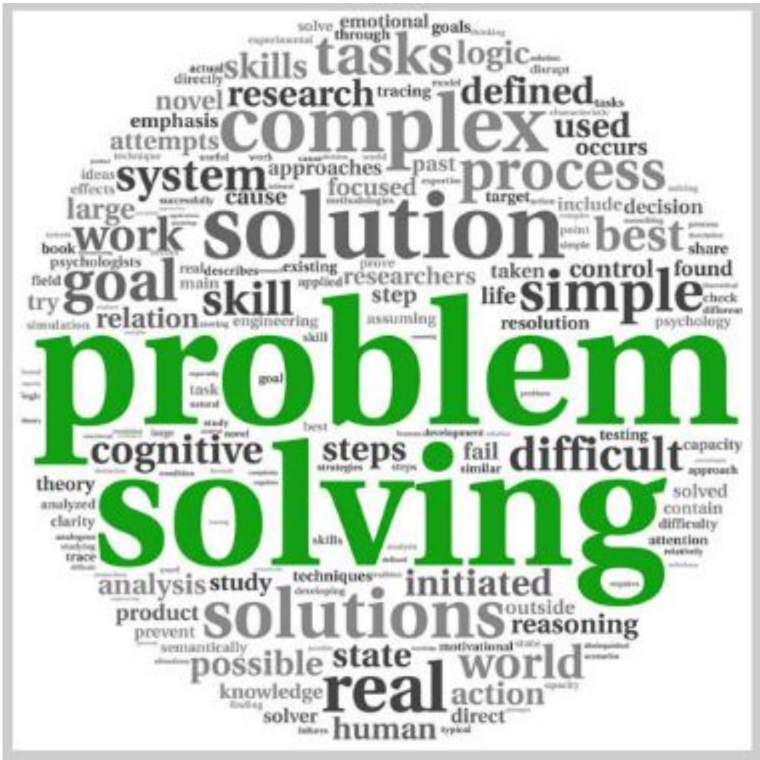


Cone of Uncertainty



User Stories

Helping to solve the Business Problem…



Start with an Epic...

Story Writing

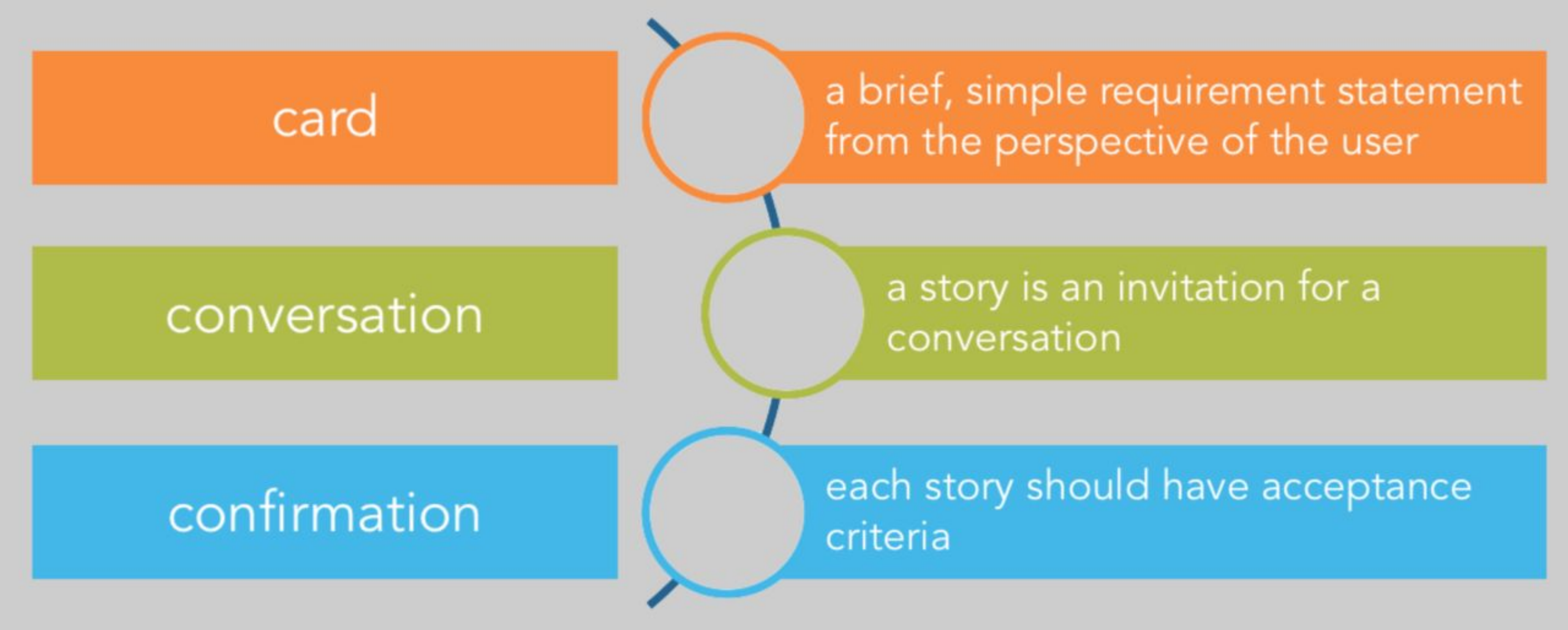
- A user story is not a specification, it is a communication and collaboration tool to help explain the why.
  - *Ron Jefferies - A Card, Conversation and Confirmation.*

- The Product Owner should not hand stories off to the development team:
  - They should rather be part of a conversation together or better yet write them as a team.
  - Share and Leverage each others knowledge and will result in better user stories.

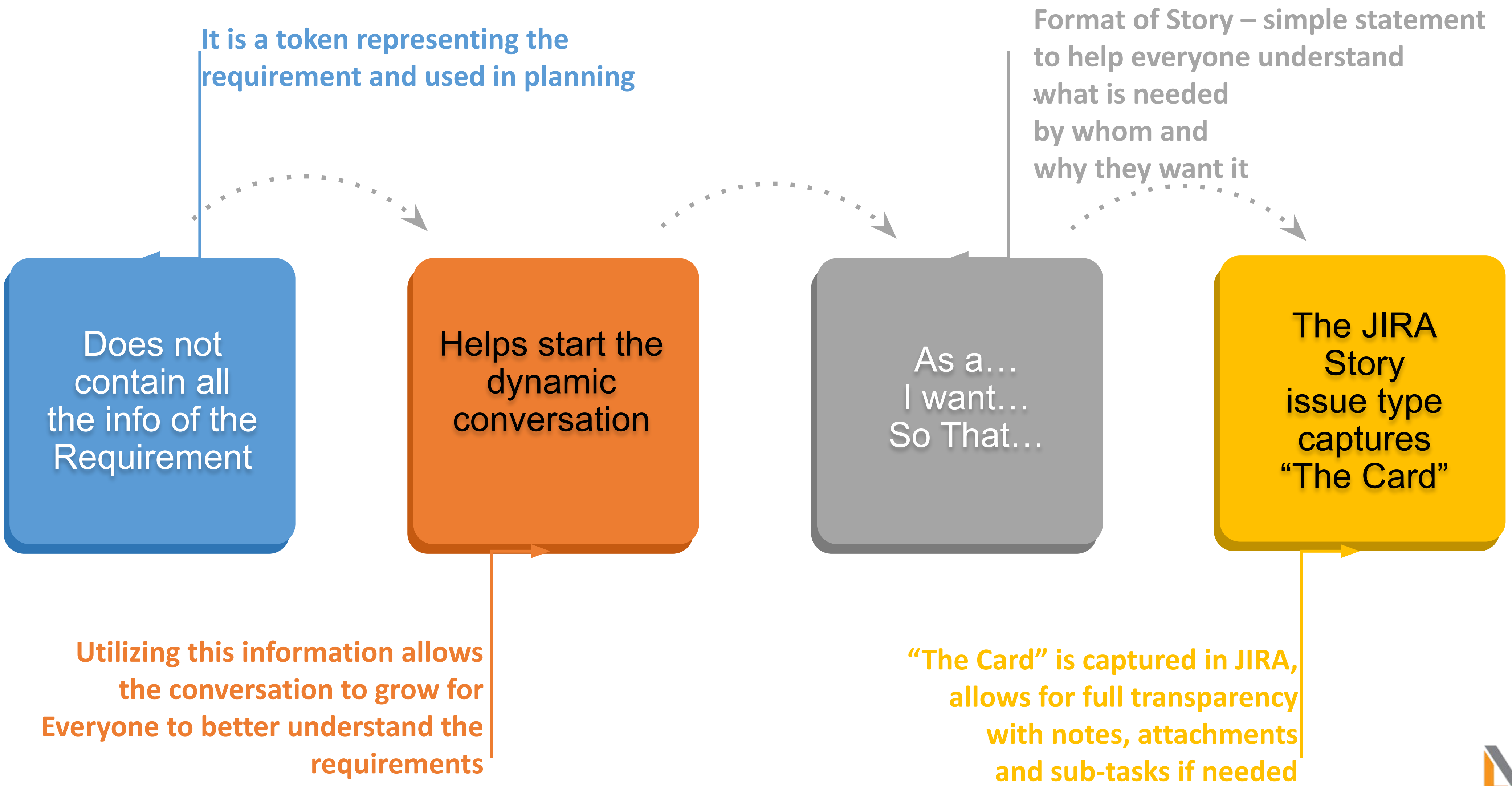
Structure of a Story INVEST

Independent	Can deliver value by itself. We want to be able to develop in any sequence.
Negotiable	Avoid too much detail; keep them flexible so the team can adjust how much of the story to implement.
Valuable	Users or customers get some value from the story.
Estimatable	The team must be able to use them for planning
Small	Large stories are harder to estimate and plan. By the time of sprint planning, the story should be able to be designed, coded and tested within the sprint.
Testable	Document acceptance criteria, or the definition of done for the story which leads to test cases

Structure of a Story



The Card


It is a token representing the requirement and used in planning
Format of Story – simple statement to help everyone understand what is needed by whom and why they want it
Does not contain all the info of the Requirement
Helps start the dynamic conversation
As a… I want… So That…
The JIRA Story issue type captures "The Card"
Utilizing this information allows the conversation to grow for Everyone to better understand the requirements
"The Card" is captured in JIRA, allows for full transparency with notes, attachments and sub-tasks if needed

The Conversation

Use Story as Foundation

Allows everyone to further discuss and understand what is required

Notes Captured in JIRA Story

Allows to enhance the story and keep the records of the team understanding

Acceptance Criteria

The Acceptance Criteria is discussed and enhanced to keep everyone on the same page

The Confirmation

Story Meets INVEST

The story needs to meet the INVEST Criteria that was discussed earlier in the presentation

Ensure there are Test Cases

Having Test Cases will allow the team to understand what needs to be completed and how will everyone understand how they will validate this.

Tests are Clear

By having clear tests will allow the Developer Test and Customer or User perform UAT

Estimation is Completed

Estimation will allow the team to ensure they understand the size, complexity and effort that is being asked of them. In addition it will allow to see the Burndown and also Velocity of the Team

Agreement by All

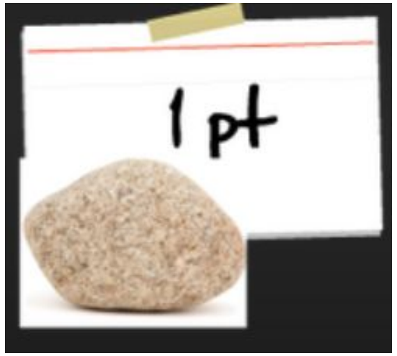
Agreement by everyone on the team on what will be completed within the sprint will allow everyone to be on the same page.

Agile Estimation (Fibonacci) and Planning

- Fibonacci Numbers
  - 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ….
  - We use the following numbers in Agile
  - 0, 1, 2, 3, 5, 8, 13, 20, 40, 100, ?

0	1	2	3	4	5	6	7	8	9	10
0	1	2	3	5	8	13	20	40	100	?

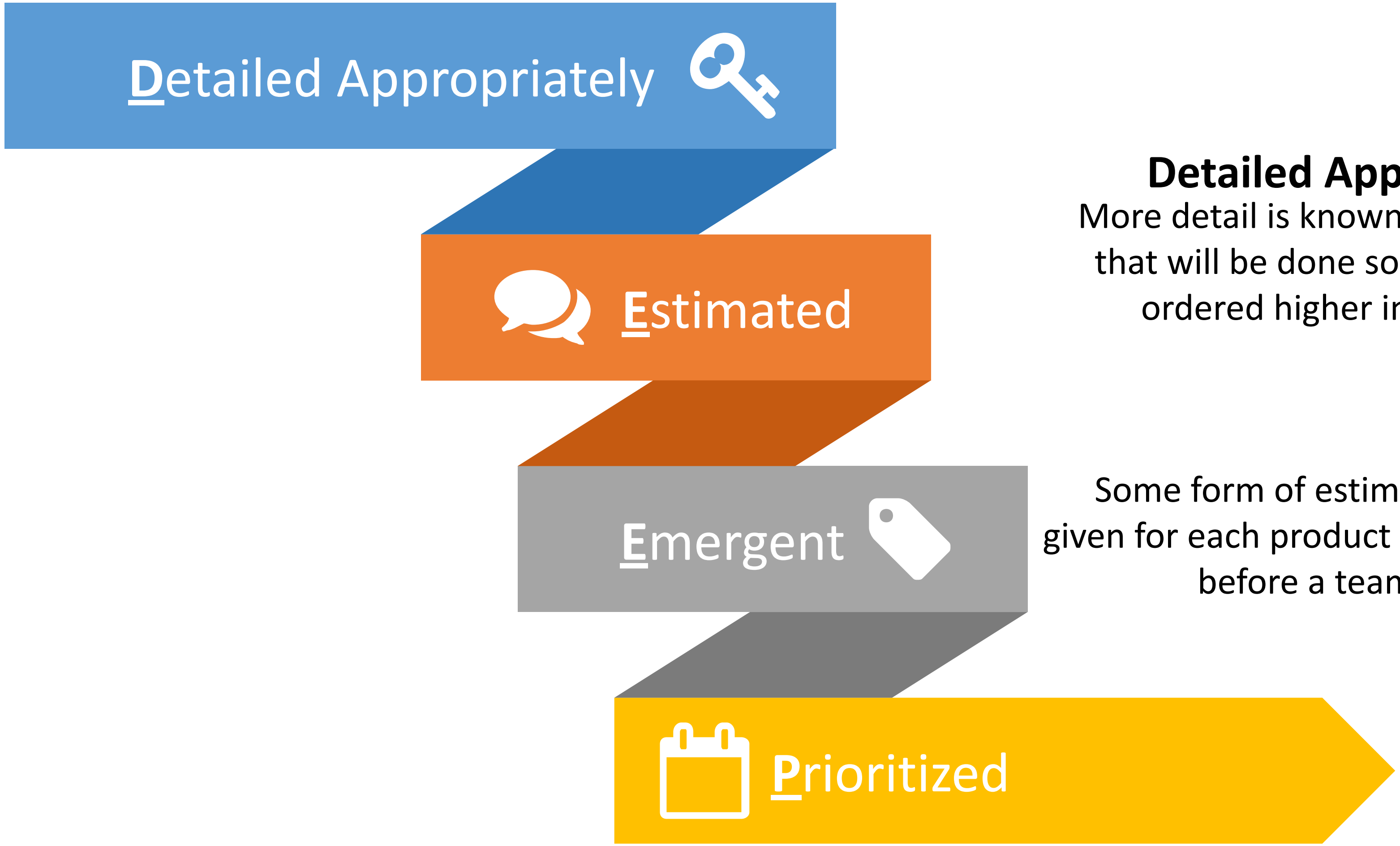
  - Let's use Fibonacci and explain how we can use these for sizing stories



Backlogs – Product Manager

Product Backlog is DEEP

Goal - Refine user stories in the backlog for future sprints - “story time” (get them to a “ready” state)



Detailed Appropriately
More detail is known about items that will be done sooner and are ordered higher in the backlog

Estimated
Some form of estimate has been given for each product backlog item before a team commits to implement

Emergent
Not everything is known upfront. We acknowledge that change happens, so planning is continual

Prioritized
The backlog is prioritized, so that stories are ordered by value to the business continually as learning progresses

Definition of Ready and Done

Example – Definition of Ready

Definition of Ready (DOR) – checked prior to starting a story in a sprint
Description and Acceptance Criteria are clearly defined
UX Documentation has been reviewed and approved by Team
All functional requirements, non-functional requirements, documents, designs and use cases have been included or linked in ticket
Dependencies are identified and documented or linked in ticket
Ticket must be 100% filled out where required by JIRA

This is suggested DoR.
Each Team needs to define their own DoR.

Definition of Done

Example – Definition of Done

Product Owner accepts the delivered story and signs off

Example Definition of Done (DOD)
The story meets all listed Acceptance Criteria
The story visually matches UX documentation (if applicable)
The code for the story is reviewed, suggestions incorporated and checked in
Unit tests and test cases are written and executed successfully
The code has been merged to master, and pushed to source control
Product owner has reviewed the story to validate that it meets the Acceptance Criteria

This is suggested DoD.
Each Team needs to define their own DoD.

Scrum Framework - Artifacts

Product Backlog

All work items related to a product/project, ordered by a Product Owner.

Burndown Chart

Visual aid for tracking team progress and forecasting expected completion dates.

Sprint Report

The Sprint Report shows the list of issues in each sprint. It is useful for your Sprint Retrospective meetings, and also for mid-sprint progress checks..

Velocity Chart

Tracks rate of completion for team. Shows the sum of promises per sprint vs. the work that was delivered

Product and Sprint Backlog

- Priority is looked at from the Product Backlog and moved to the Sprint Backlog
- Some teams look at hours at the task level, others only look at Story Points of stories

Product Backlog

PB Item	Priority	
User Story	High	5
User Story	High	8
User Story	High	3
User Story	Med	13
User Story	Med	
User Story	Low	
User Story	Low	13

Sprint **Planning**

Top priority stories are discussed and decomposed into Tasks for the Sprint Backlog.

Sprint Backlog

SB Item	Priority	Hrs
User Story	High	
Task 1		4
Task 2		12
Task 3		6
User Story	Med	
Task 1		9
Task 3		2
User Story	Low	
Task 1		5
Task 2		2
Task 3		7

Sprint

Team **pulls** and completes Tasks until each User Story is done.

How to read Burndown Chart

- Visual aid for tracking team progress and forecasting expected completion dates.
- Burndown Charts can provide context to make tough decisions for sprints

Backlog

- Story 1
- Story 2
- Story 3
- Story 4
- Story 5
- Story 6
- Story 7
- Story 8
- Story 9
- Story 10

Product Owner Speaking

To date, we have completed Story 1 through Story 4.

Unfortunately, we lost several key members of our team during iteration 6 and **we are unlikely to get all planned Stories done for this release,** unless we execute with perfection.

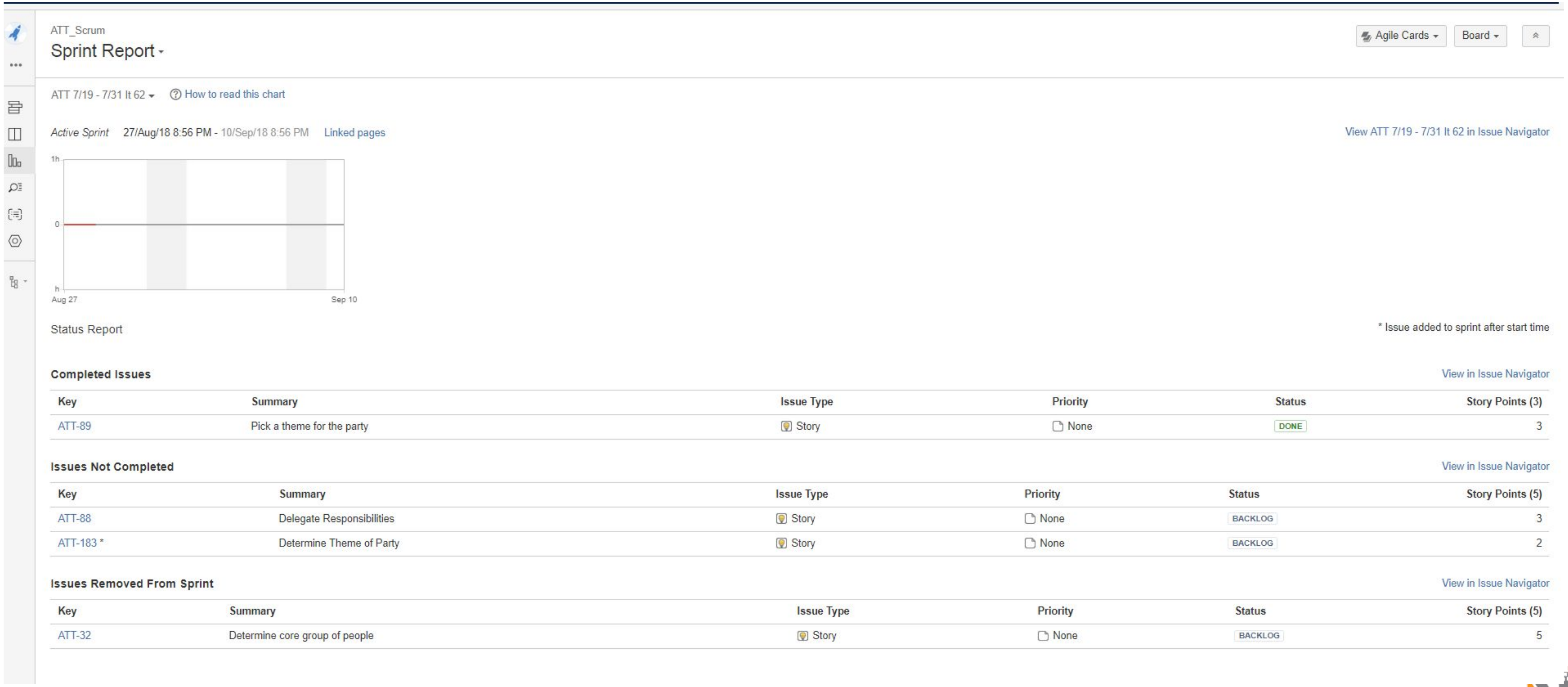
We will likely **delay Story 9 and 10 until the next release,** unless we make some tradeoffs**.**

We already started discussions with sales and marketing and we may limit our work on Story 5 and 6 in the next Sprint.

LEARNING VOYAGE

JIRA Sprint Report

The Sprint Report shows the list of issues in each sprint. It is useful for your Sprint Retrospective meetings, and also for mid-sprint progress checks.



ATT_Scrum

Sprint Report

Agile Cards | Board

ATT 7/19 - 7/31 It 62 | How to read this chart

Active Sprint 27/Aug/18 8:56 PM - 10/Sep/18 8:56 PM | Linked pages

View ATT 7/19 - 7/31 It 62 in Issue Navigator

Status Report

* Issue added to sprint after start time

Completed Issues

View in Issue Navigator

Key	Summary	Issue Type	Priority	Status	Story Points (3)
ATT-89	Pick a theme for the party	Story	None	DONE	3

Issues Not Completed

View in Issue Navigator

Key	Summary	Issue Type	Priority	Status	Story Points (5)
ATT-88	Delegate Responsibilities	Story	None	BACKLOG	3
ATT-183 *	Determine Theme of Party	Story	None	BACKLOG	2

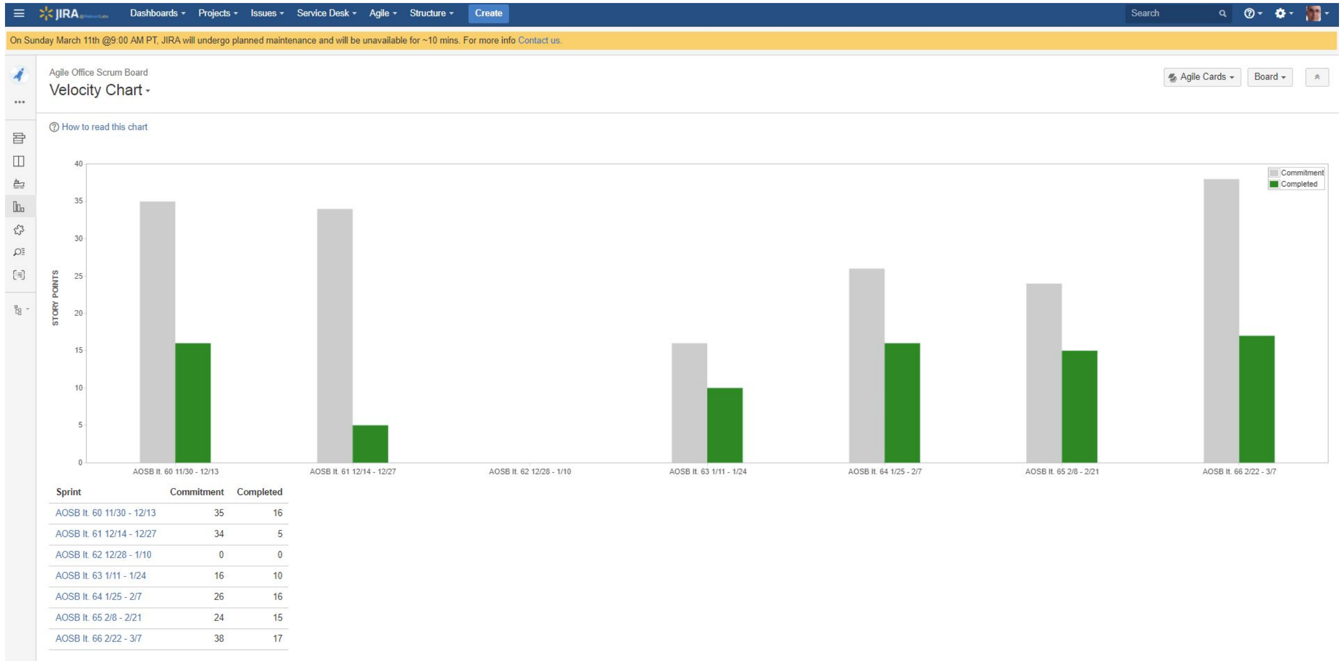
Issues Removed From Sprint

View in Issue Navigator

Key	Summary	Issue Type	Priority	Status	Story Points (5)
ATT-32	Determine core group of people	Story	None	BACKLOG	5

Using Velocity for Planning

- Velocity is a measure of how much work can be delivered in a timebox
- It is based on the historical performance of the team for recent sprints
- We use established velocity for planning sprints and for use in estimating releases

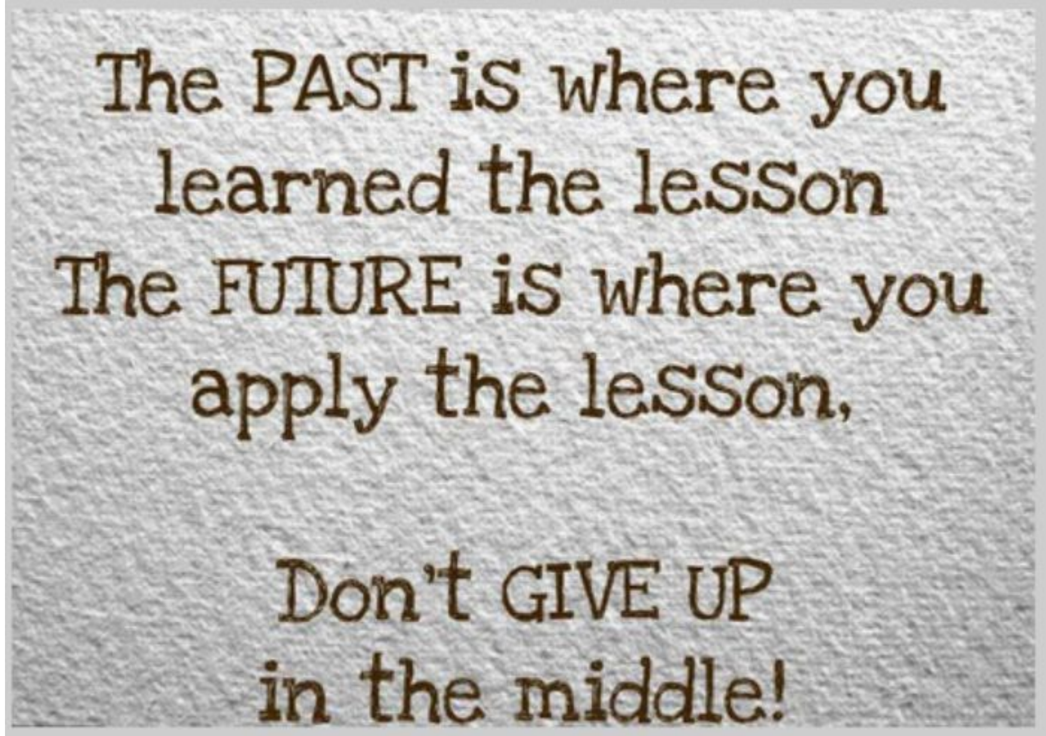


LEARNING VOYAGE

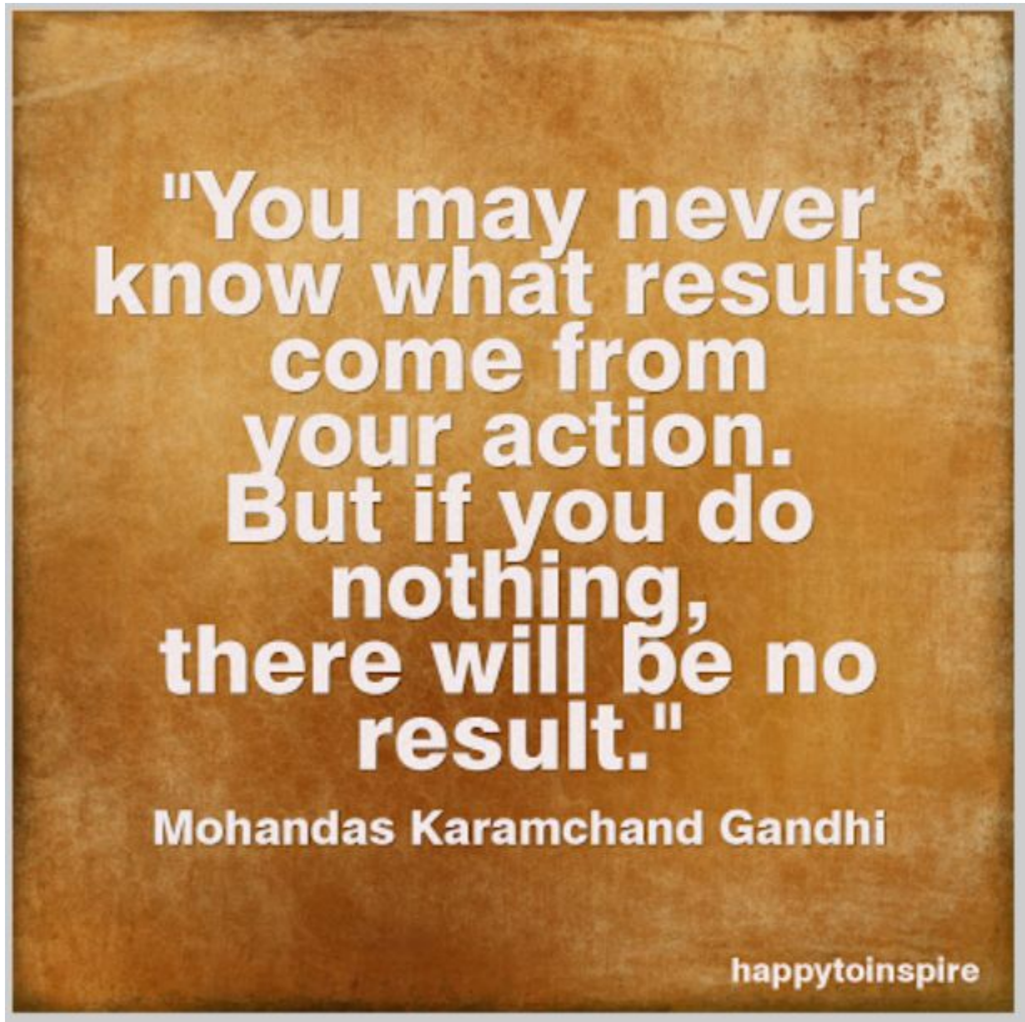
Retrospective

Retrospectives = Focus?

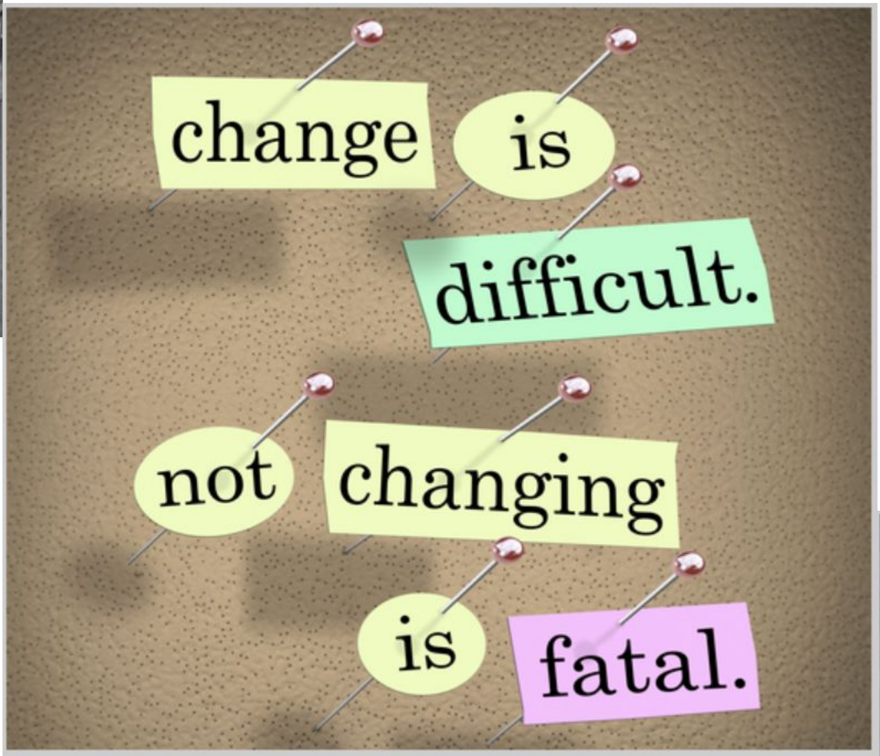
Retrospective – Purpose and When

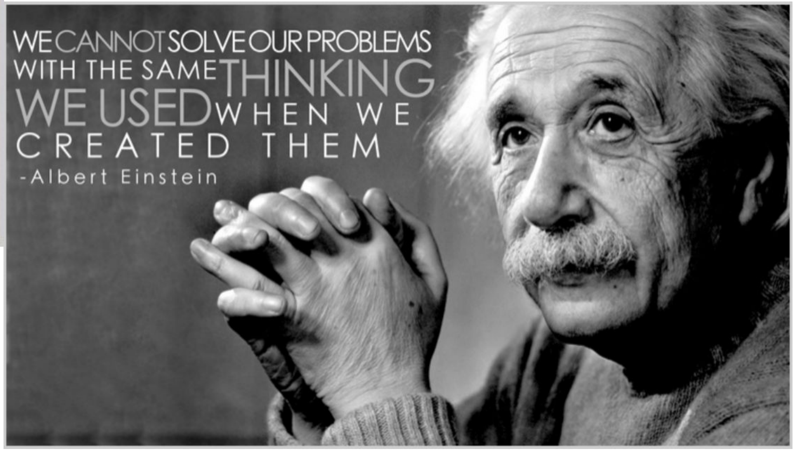


Why do Retrospecitves



Mindset during Retrospective

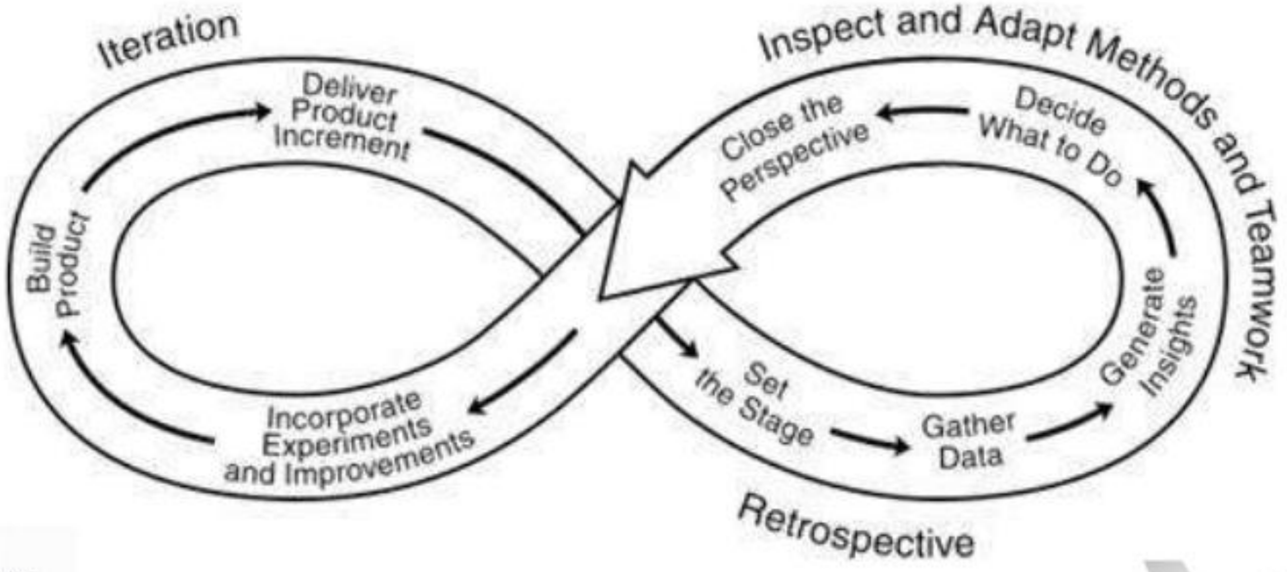




Tip prior to Retrospective

Making good teams Great!!!

Why Retrospect



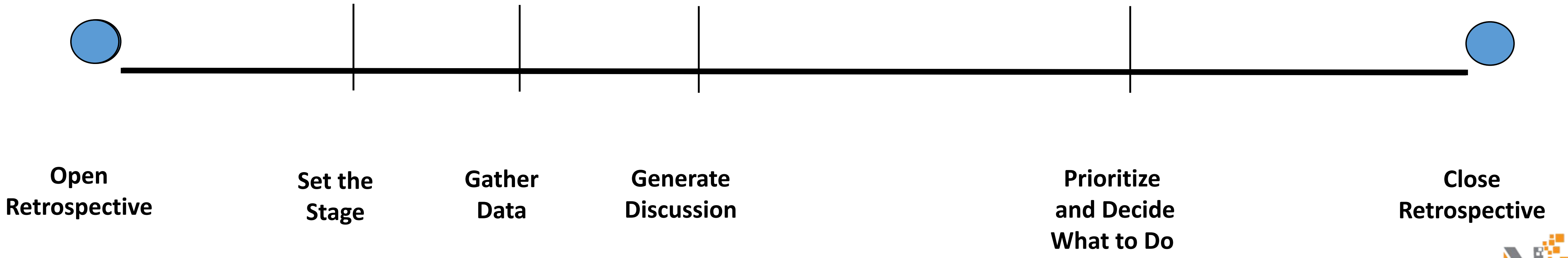
Credit: Agile Retrospectives – Making good teams Great by Esther Derby & Diana Larsen
http://pragprog.com/book/dlret/agile-retrospectives

AgileSparks

Structuring the Retrospective – 6 points

- Open Retrospective
- Set the Stage
- Gather Data
- Generate Discussion
- Prioritize and Decide What to Do
- Close Retrospective

- Below is timeline of meeting:



Gathering Data – Learning Matrix

Happy Face – aspects you liked should be repeated

Sad Face – aspects you disliked – should be changed

Flower bouquet for people you wish to appreciate

Light bulb for new ideas to try

Gathering Data - Actions

Actions

Appreciations

What you liked during the pervious iteration

Risks

Future pitfalls that can endanger the project

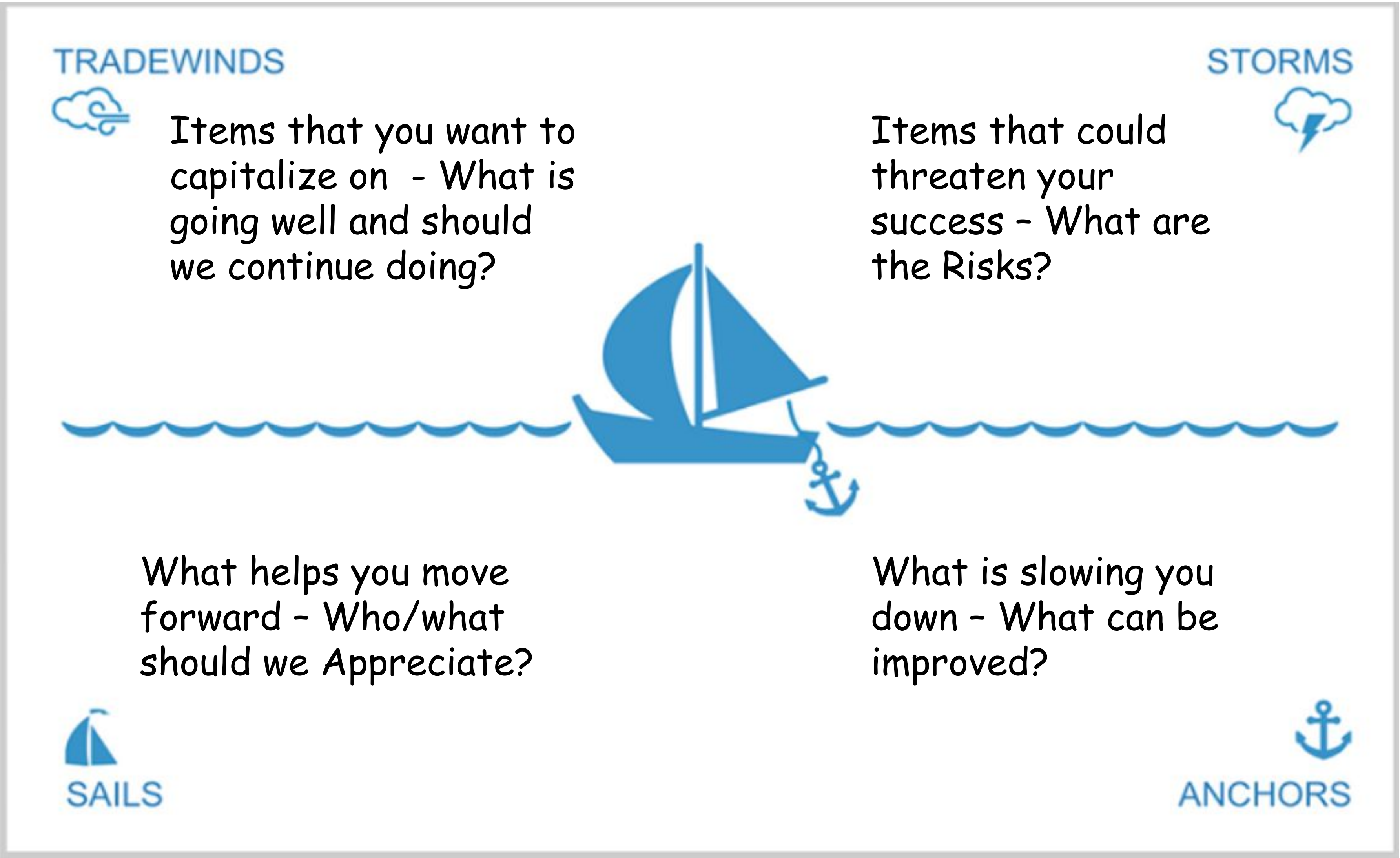
Puzzles

Questions for which you have no answer

Wishes

Not improvements, but rather ideas of your ideal project

Gathering Data – Sailboat

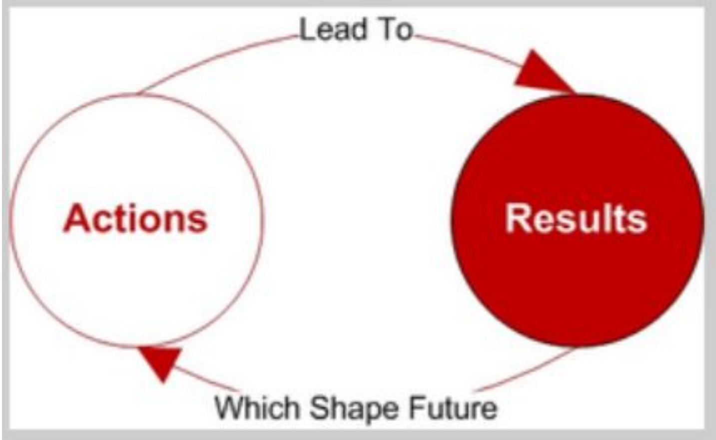


Gathering Data – Start, Stop, Continue

- Simplest retrospective method – very effective
- Reflect on three things
- Facilitator has team focus on each question separately in 3 short "rounds"
- Team votes on specific items to focus on during the coming sprint
- Review these items at the beginning of next retrospective

Prioritize and Close Retrospective

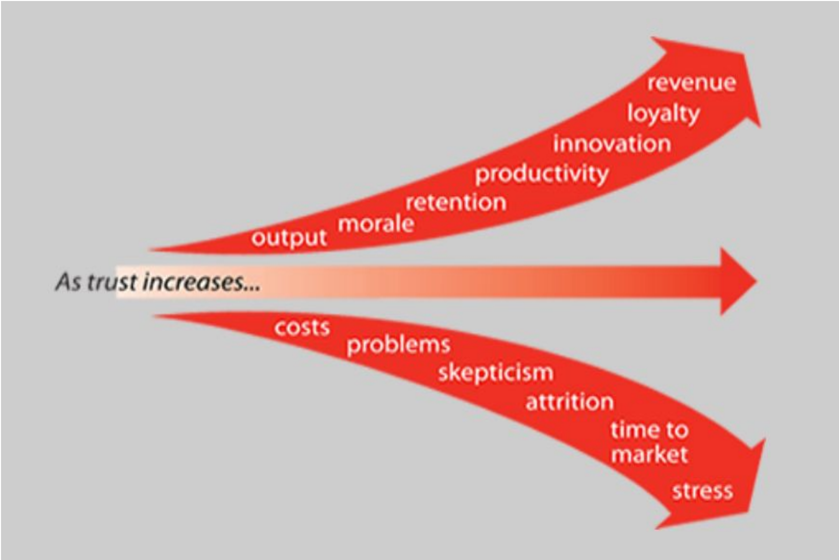
Reiterate actions and follow-up



Appreciate contributions

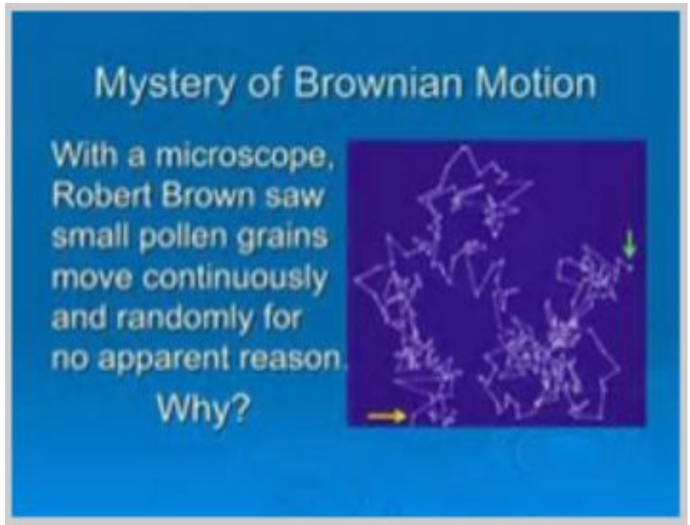
Identify ways to make the next retrospective better

Benefits of doing Retrospective



Leave out…what happens

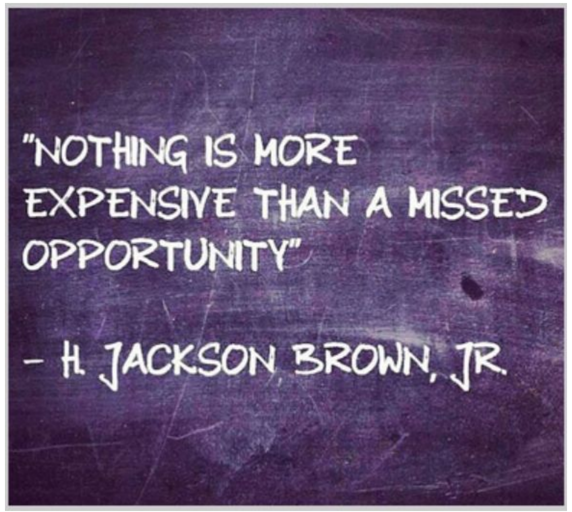
Setting the Stage:



Generating Discussion:

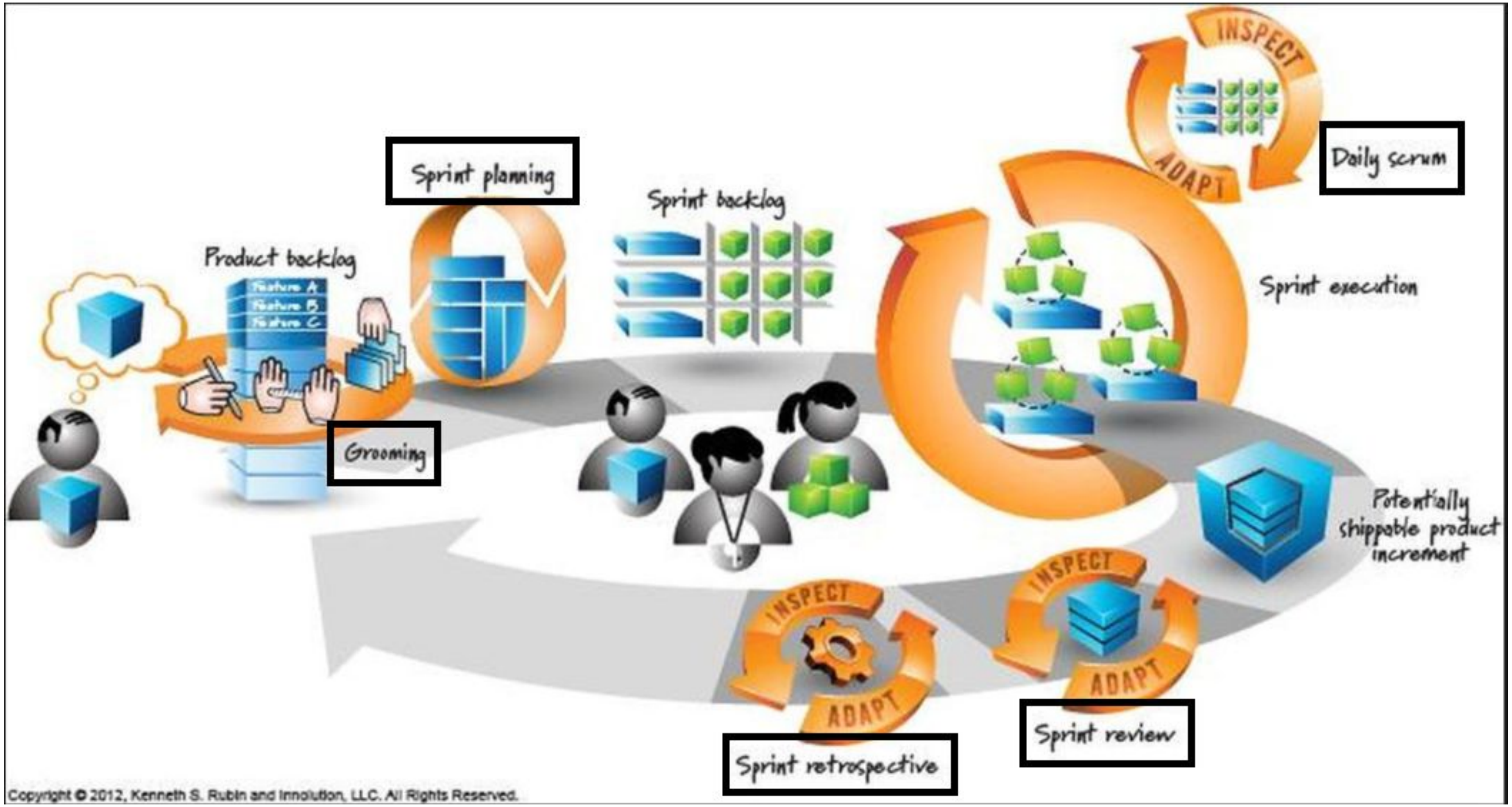
Gathering Data:

Closing the Retrospective:



Deciding What to Do:

Better Understanding – Scrum Framework



Thanks

Professor Ernesto Lee

Product requirements documents, downsized

- Nobody likes writing bloated, ultra-detailed product requirements documents.
- Turns out nobody likes using them, either.

Product requirements documents, downsized

- A product requirements document defines the product you are about to build: It outlines the product's purpose, its features, functionalities, and behavior.

Gathering requirements in an agile world

- Agile requirements are a product owner's best friend. Product owners who don't use agile requirements get caught up with spec'ing out every detail to deliver the right software (then cross their fingers hoping they've spec'ed out the right things).

Creating a shared understanding among teams

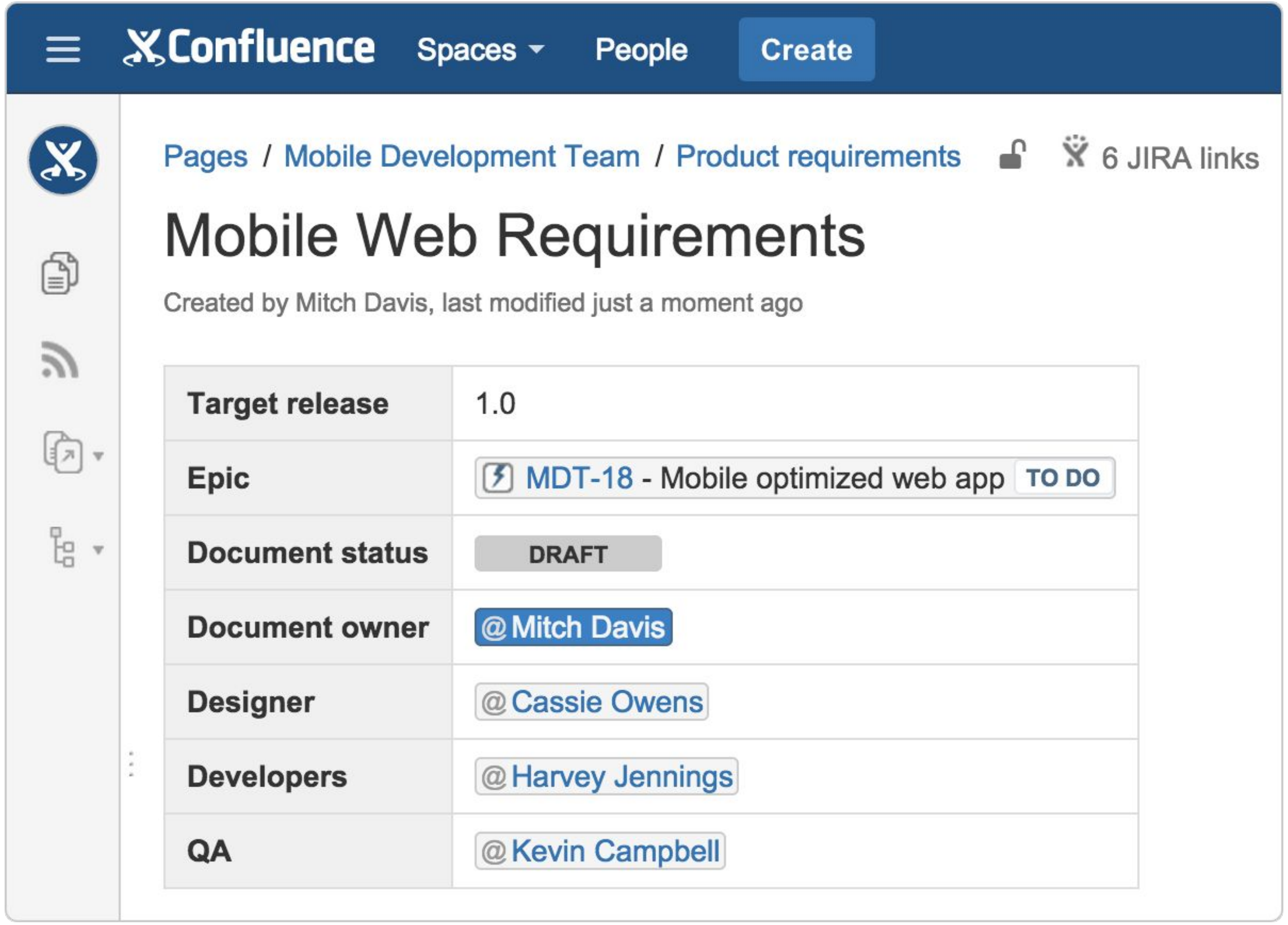
If you're excited by the idea of a shared understanding, but haven't a clue how to create it, try some of these tips:

- When conducting customer interviews, include a member of the design and development teams so they can hear from a customer directly instead of relying on the product owner's notes. It will also give them the chance to probe deeper while the topic is fresh in the customer's mind.

Keeping requirements lean with a one-page dashboard

- When writing a requirements document, it's helpful to use a consistent template across the team so everyone can follow along and give feedback.

1. Define project specifics



2. Team goals and business objectives

- Get straight to the point.
- Inform, but don't bore.

3. Background and strategic fit

Goals

- Our goal is to create a mobile version of the website. Sometimes users click on a link in an email notification using their mobile phone and need to be able to access our application from mobile Chrome or Safari.
- We want to meet feature parity with most functions - except we can skip creating events.

Background and strategic fit

We all know mobile is on the rise. A recent survey to customers showed that 85% of users use their mobile on a daily basis. Most of our customers also use competitor apps, so this is something we need to have. We will be able to measure our success through analytics and can use the website today as a baseline.

Customer research

- Customer interview - Netflix
- Customer interview - Homeaway
- Customer interview - Bitbucket

4. Assumptions

- List the technical, business, or user assumptions you might be making.

5. User Stories

Requirements

#	User story title	User story description	Priority	Notes
1	Facebook Integration JIRA | MDT-13	A user wants to sign up via Facebook	MUST HAVE	• We will need to talk to Cassie Owens about this one. • There has also been some research done on this (see Facebook integration prototype)
2	Activity Stream JIRA | MDT-14	A user wants to view the latest updates via the mobile dashboard so that they can get a better understanding of what is in place	MUST HAVE	
3	Post Updates JIRA | MDT-15	A user wants to be able to post status updates on the go	MUST HAVE	The key things we will need to support: • Text status updates • Mentions • Support for images • Smart embedding for things like YouTube videos etc.
4	API JIRA | MDT-16	A developer wants to integrate with the mobile app so that they can embed the activity stream on their website	SHOULD HA...	• We should chat to Team Dyno as they did something similar.

6. User interaction and design

User interaction and design

<table>
<tr><th>Description</th><th>Login screen</th><th>Activity stream</th></tr>
<tr><td>Mockup</td><td>
</td><td>
</td></tr>
</table>

7. Questions

- As the team understands the problems to solve, they often have questions.
- Create a table of "things we need to decide or research" to track these items.

8. What we're not doing

- Keep the team focused on the work at hand by clearly calling out what you're not doing.
- Flag things that are out of scope at the moment, but might be considered at a later time.

Example of a one-page PRD

Mobile Web Requirements

Created by Mitch Davis, last modified just a moment ago

Target release	1.0
Epic	MDT-18 - Mobile optimized web app TO DO
Document status	DRAFT
Document owner	@Mitch Davis
Designer	@Cassie Owens
Developers	@Harvey Jennings
QA	@Kevin Campbell

Background and strategic fit

We all know mobile is on the rise. A recent survey to customers showed that 85% of users use their mobile on a daily basis. Most of our customers also use competitor apps, so this is something we need to have.

Customer research

- Customer interview - Netflix
- Customer interview - Homeaway
- Customer interview - Bitbucket

Requirements

#	User story title	User story description	Priority	Notes
1	Facebook Integration MDT-13 TO DO	A user wants to sign up via Facebook	Must Have	• We will need to talk to Cassie Owens. • There has also been some research done on this (see Facebook integration prototype)
2	Activity Stream MDT-14 TO DO	A user wants to view the latest updates via the mobile dashboard so that they can get a better understanding of what is in place	Must Have	
3	Post Updates MDT-15 TO DO	A user wants to be able to post status updates on the go	Must Have	The key things we will need to support: • Text status updates • Mentions • Support for images • Smart embedding for YouTube vids
4	API MDT-16 TO DO	A developer wants to integrate with the mobile app so that they can embed the activity stream on their website	Should Have	• We should chat to Team Dyno as they did something similar.

User interaction and design

Description	Login screen	Activity stream
Mockup		

Questions

Below is a list of questions to be addressed as a result of this requirements document:

Question	Outcome
What about Google Apps	• We think this is important, but not for version one. • We can look at this at a later stage. • It might be worth someone looking into a shared notification library to do this.
Are we supporting Blackberry?	• Again, not for initial version - but we haven't had much demand for this.
Should we have an offline mode?	• We've talked about the pros and cons. In brief: + Seamless experience for customers, they won't notice if there is a connection issue + Most of our competitors don't have this − Could be expensive to build ? Should we spike this at a later sprint?

Not Doing

- Google Apps Authentication - out of scope, see above for details
- Blackberry support - we won't look at doing this, if demand picks up we can look at it.
- Native app. We are starting with a mobile web view first and get back to a native app depending on feedback that we get.

Like Be the first to like this requirements

Key takeaways from the one-page approach:

Here are the benefits and challenges we've observed with the one-page dashboard approach:

- One page, one source
- Extra agility
- Just enough context and detail
- Living stories
- Collective wisdom
- Engaging "extras“
- Collaboration!

Challenges

With every approach there are down-sides. Here there are two main challenges we’ve experienced and observed from customers as well:

- Documentation can go stale
- Lack of participation

Now get to work!

- When requirements are nimble, the product owner has more time to understand and keep pace with the market.
- And keeping them informative-but-brief empowers the development team to use whatever implementation fits their architecture and technology stack best.

How to prioritize features using NPS®

- The great thing about NPS? It requires one question, and gives you measurable results.

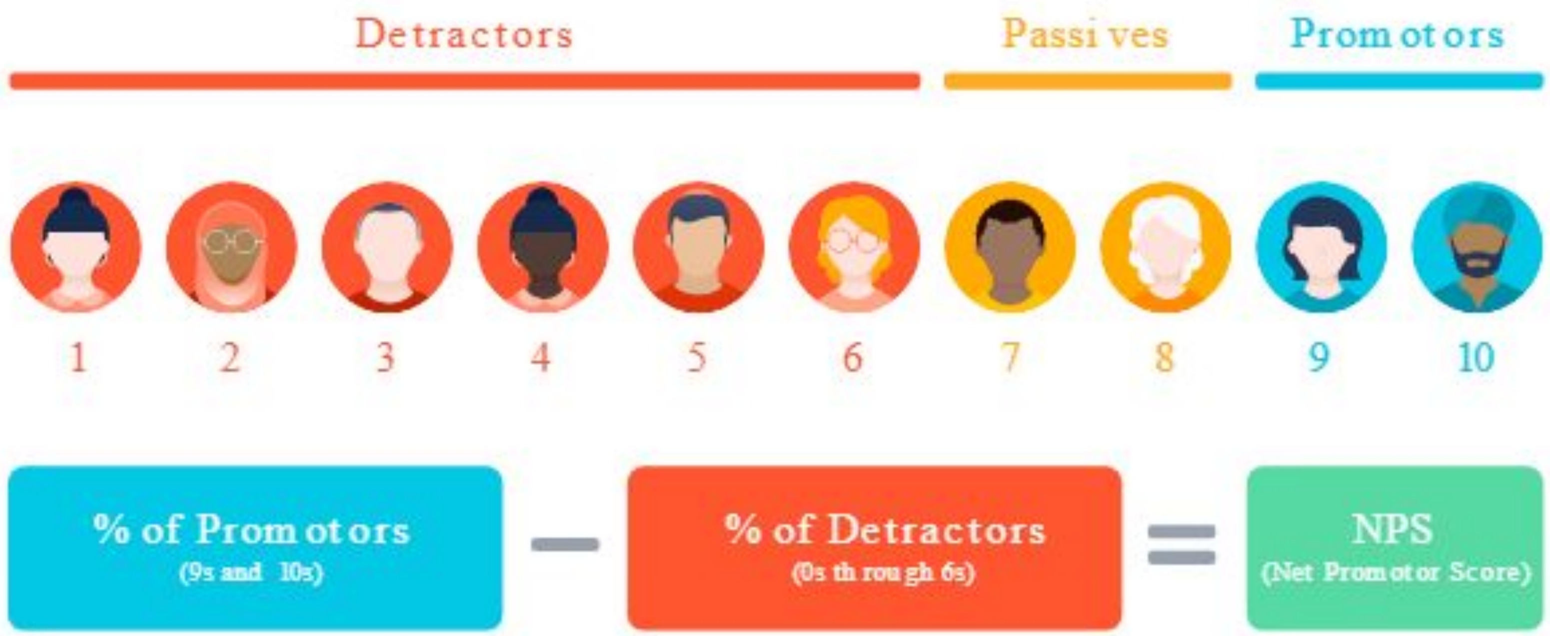
A quick NPS primer

An NPS survey consists of one simple question: “How likely is it that you would recommend to a friend or colleague?"

This is how it looks in an ernesto.net product:

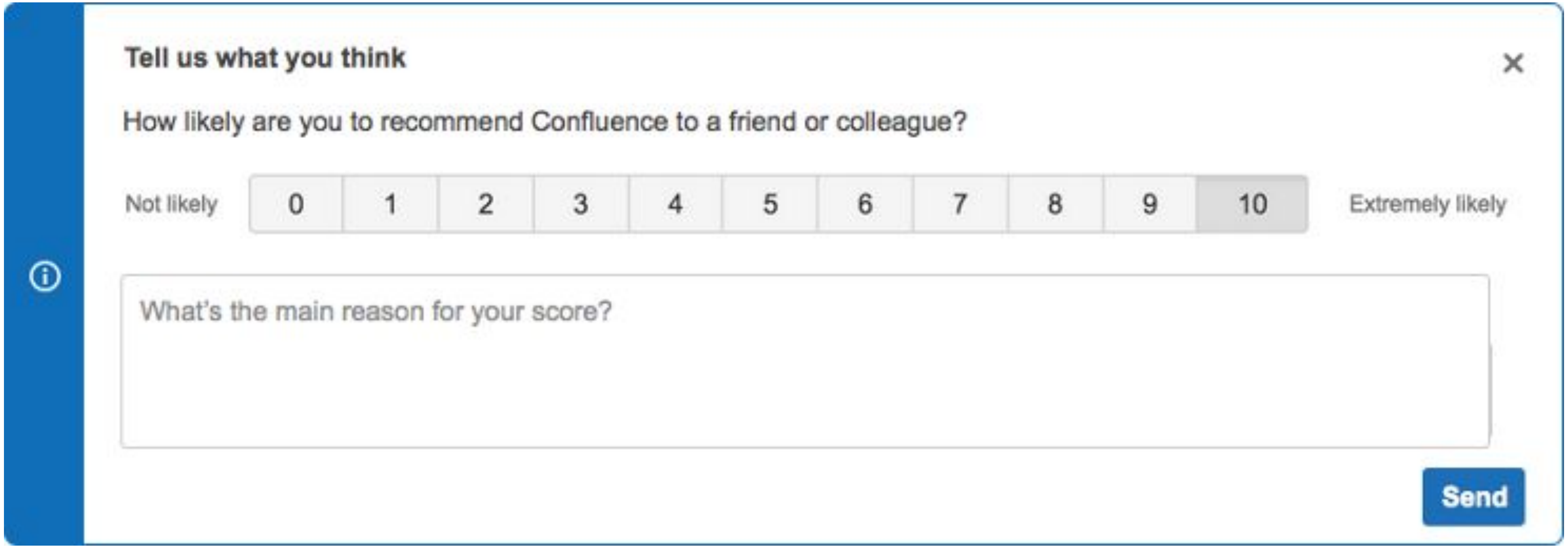
A quick NPS primer

Once you have collected a significant number of response from your NPS survey, you group your responses into three groups: Promoters (9–10), Passives (7–8) and Detractors (0–6).



A quick NPS primer

That's why we ask for feedback once the score is selected:



Step 1: Categorize NPS Verbatims

- Let's look at some examples of verbatims we've received about Confluence, ernesto.net's team collaboration product, and how we use them in our feature prioritization process:

"LOVE YOU, LOVE CONFLUENCE... OUR PRODUCTIVITY HAS IMPROVED ASTRONOMICALLY... THANK YOU... THANK YOUR PRODUCT MANAGEMENT AND DEVELOPMENT STAFF... YOU HAVE BUILT A BEAUTIFUL APPLICATION."

MANAGEMENT

10

"USEFUL FUNCTIONALITY BUT DIFFICULT USABILITY. (GOOD) FORMATTING IS DIFFICULT."

MANAGEMENT

5

Step 1: Categorize NPS Verbatims

- Let's look at another one:

"TABLES AREN'T POWERFUL ENOUGH - I SOMETIMES NEED TO BE ABLE TO CUT-AND-PASTE COLUMNS."

SOFTWARE-ENGINEER

8

Step 1: Categorize NPS Verbatims

- We add the verbatim and two new labels:

NPS Score	Verbatim	Usability	Formatting	Tables	Tables - Cut/Paste Columns
5	Useful functionality but difficult usability. (Good) formatting is difficult	1	1		
8	Tables aren't powerful enough - I sometimes need to be able to cut-and-paste columns.			1	1

LEARNING VOYAGE

Step 2: Analysing the results

The first step is counting the number of feedback items in each category.

Let's say we get:

Category	Totals
Performance	90
Tables	60
Tables - Cut/Paste Rows	20
Tables - Column Width	30
Usability	50
Formatting	45
New Features	25
Concurrent Editing	20

Step 2: Analysing the results

Removing the sub-tags (for now) and reordering, we quickly see our main priorities:

Category	Totals
Performance	90
Tables	60
Usability	50
Formatting	45
New Features	25
Concurrent Editing	20

Step 2: Analysing the results

We can calculate NPS scores for each category to help answer those questions:

Category	Totals	NPS
Performance	90	30
Tables	60	-30
Usability	50	-10
Formatting	45	-5
New Features	25	25
Concurrent Editing	20	20

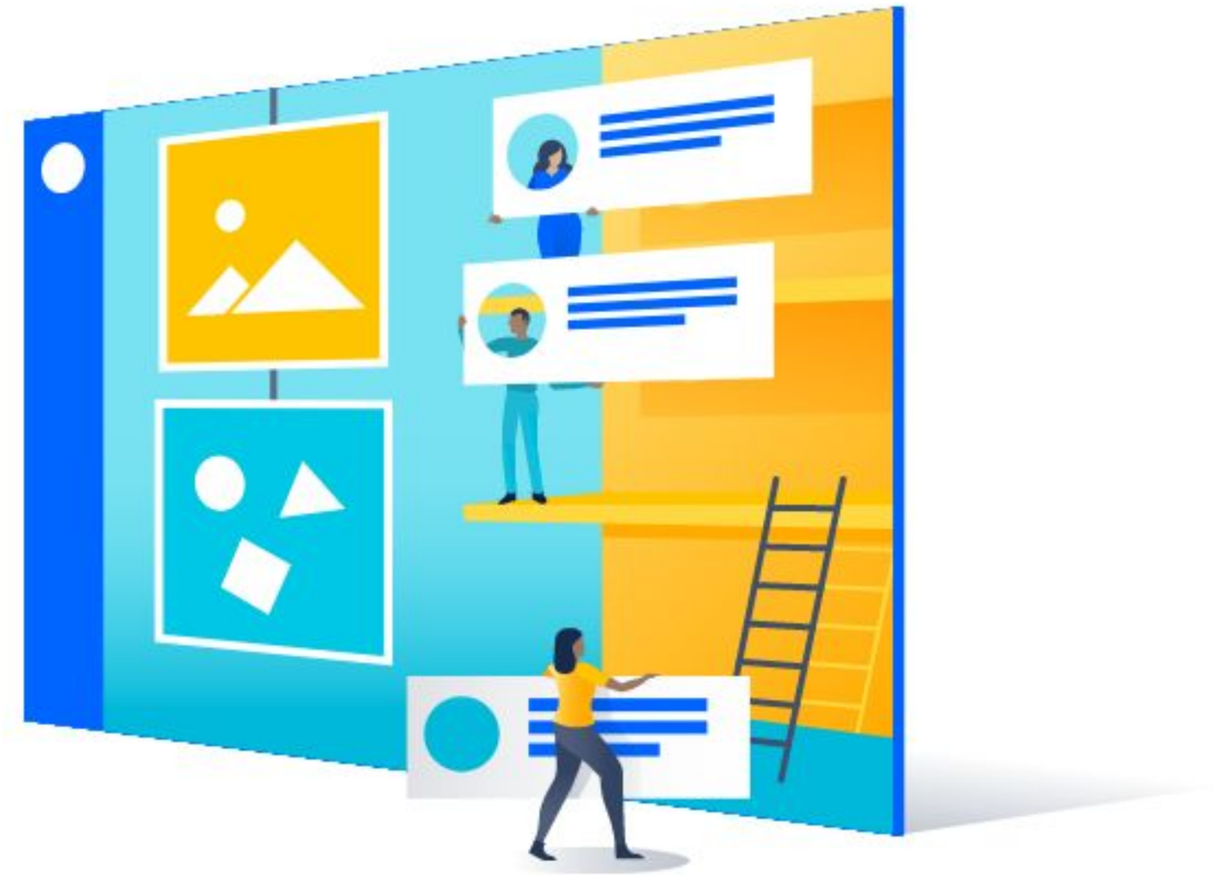
Step 3: Let's get our priorities straight

- You've classified your verbatims and identified the opportunities with the most impact. Although direct customer feedback is powerful, it's one input that we should gut check against other customer data (research, product feedback, support contacts, churn feedback, etc.) so that you're confident that you're improving the most important experience.

- As product managers, we take every opportunity we get to learn more about our customers because understanding their needs is critical to building useful products. This means conducting customer interviews, running surveys, and examining in-product analytics.

An intro to product analytics

- In order to get a quantitative understanding of what users are doing with your product, the first step is instrumenting it with analytics.



“Empathy debt”: the newest kind of debt

- Lets try out this new term, “empathy debt.”

- In-product analytics can help you pay off empathy debt in two ways: with qualitative feedback gathered through activities like concept testing and customer interviews; and with quantitative data collected in-product with things like product analytics and NPS surveys.

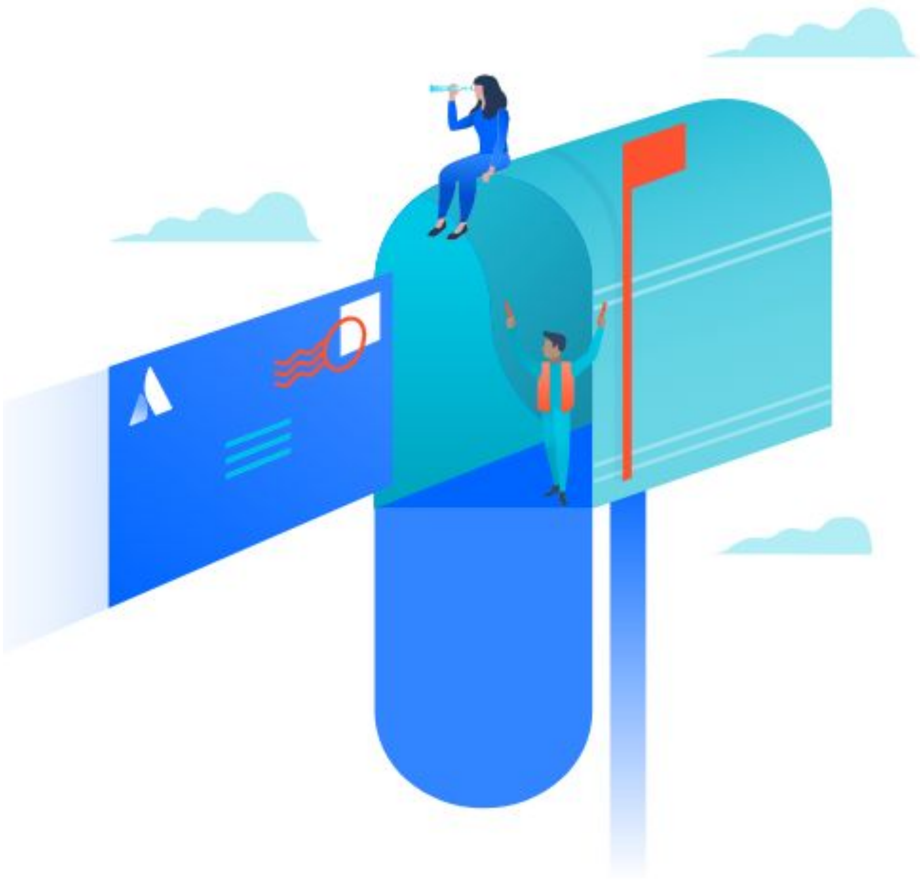
Testing the future before it's here

- While adding product analytics can be valuable for understanding how users use existing features, they're also extremely valuable for testing new features and experiences.

- If you have a clear goal for how much you want your feature to be used, having product analytics helps you work towards that old agile mantra of failing fast and iterating until you succeed.

Just don’t forget to listen to your users, too

- As I mentioned above, it’s great to be data-informed, but being entirely data-driven can sometimes leave you blind to the overall experience that you’re creating for users.
- Being dependent entirely on data can also be a bit crippling when it comes time to make a decision and you don’t have all the data you need.



How to thrive in remote product management

- This year has been challenging to say the least. As the world grappled with a global pandemic, ernesto.net made the choice to make all its employees worldwide fully remote until the end of 2020.

- More recently, ernesto.net announced our “TEAM Anywhere” initiative, allowing our employees to ultimately choose where they work – whether at home, in the office, or a combination of the two. It’s one thing to work independently from home, alone, and quite another to work as a team remotely.

Of course there are challenges

We quickly realized we couldn’t simply work the same way we had always done before. Home offices lend a variety of different distractions and challenges. Children, spouses, roommates, and/or pets can interfere with work and meetings.

In fact, during a presentation I gave over video conferencing, my dog snuck up behind me and stuck a toy on my head.

Tips to be a successful remote product manager

Set work/life boundaries
Personally, I found that clear boundaries are important when working remotely. I created a Confluence page that clearly states when I work, when I'm available (and in what time zone), and what my priorities are for the week.

Megan Cook work expectations including COVID-19

Created by Megan Cook
Last updated Jun 30, 2020 • 1 min read • Analytics • Who's looking?

Last updated Jul 1, 2020

This page details my working schedule as I transition back into working life from 9 months of maternity leave. With the COVID-19 situation I'm now working from home with a baby who absolutely knows I'm here and would like to continue our 24/7 joined-at-the-hip relationship.

Working week	4 days (Tuesday to Friday) 38 hours per week I will not be checking Slack or email on Mondays.	**Duration: 6 months** We will check in periodically to make sure this arrangement is working It's hard to predict how this transition will go. At 6 months we will assess what future arrangements should look like.
Hours	Roughly from 6am to 4pm • allows for good cross over with the US • allows for daycare pick up	

Priorities - week of Jun 30, 2020

HIGH PRIORITY	LOW PRIORITY	NOT DOING

Tips to be a successful remote product manager

Over communicate

- One way to overcome the communication challenges of remote working is to be sure to over communicate, as my colleague Dan Radigan noted.
- When a decision is made, each team member should understand the decision and ideally why it was made.
- This can be done using a content management system like a wiki where team members can easily browse for updates across the team. They can then be notified via a Slack group chat.

Tips to be a successful remote product manager

Remote product research

- Product management requires close contact with customers and is especially important when working remotely.
- Even before COVID-19, much of the research work we did at ernesto.net was performed remotely.
- By using Zoom our customers can share their workspaces and demonstrate how things really work for them.

Tips to be a successful remote product manager

Presenting to a remote audience

- It's important to have a clear product vision outlined with documentation that is concise, clear, compelling, and visually engaging (with diagrams, charts, etc.).
- A product manager should give a presentation that clearly articulates the product value, product elements, expectations, and a clear path to achieve success.
- This helps to cut down on the amount of length of meetings.

Tips to be a successful remote product manager

Information sharing and accountability

- Each Friday, the entire product marketing team meets via video conference and shares accomplishments, challenges, concerns, and more.
- We also have a product marketing meeting each week where customer-facing workers meet, including someone from the support team and technical account managers team.
- In this meeting everyone shares the top five things customers care about.
- You get a really quick, rich digest of what's happening in each team and understand if patterns cross different areas as well.

Tools that rock my remote working

- Zoom
- Slack
- Collaboration and productivity
- G Suite
- Confluence
- Project and task management
- Jira
- Trello
- Bitbucket
- Clockwise
- MURAL

In conclusion…

- While we have certainly been disrupted over the past year and forced to reimagine and change the way we work, I believe disruption is a key element of innovation.

- I also believe that working remotely has helped me and our teams develop better, more refined communication skills.

- Having the right tools has helped make this transition easier.

Welcome

- Enterprise JIRA Overview/Training
- Dr. Ernesto Lee

Agenda

1st Section

- Definition JIRA
- Organizational Structure
- How to Log into JIRA
- Navigation Bar
- Projects
- Demo

2nd Section

- Agile Boards
- Demo
- Issue Navigator
- Demo

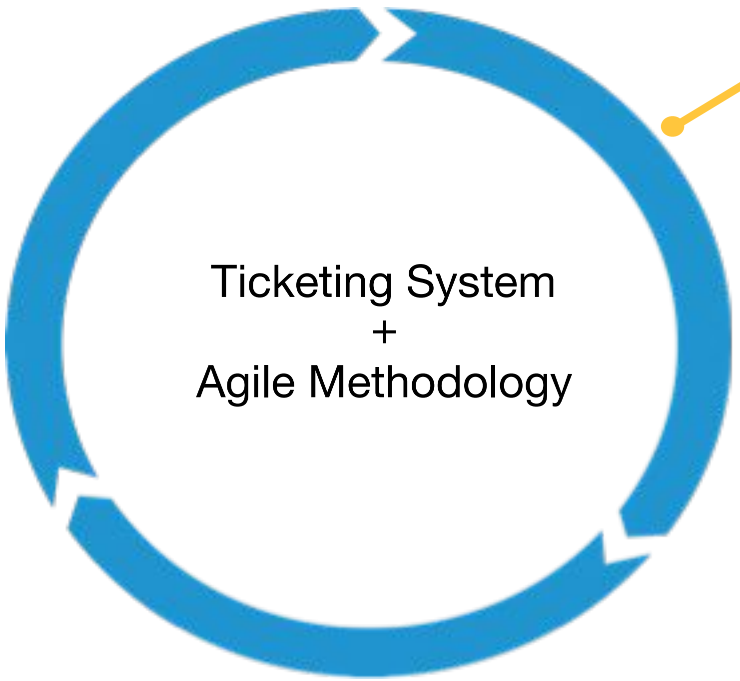
3rd Section

- Q&A

What you should walk away with

What is JIRA?

What is JIRA?



- Customized for Walmart
- Scrum & Kanban
- Log, Track and Work Issues
- Iterating (plan and work)
- Key Agile

JIRA Structure

Category

Bucket of Projects (allows you to group the projects)
@Services, @Platform, Us Migration

Project

Gets and groups all of the issues
Cart Services, IMSREquest a new project works space - https://jira.walmart.com/servicedesk/customer/portal/1/create/1

Issue

Epics, Story, Bug, Test Case

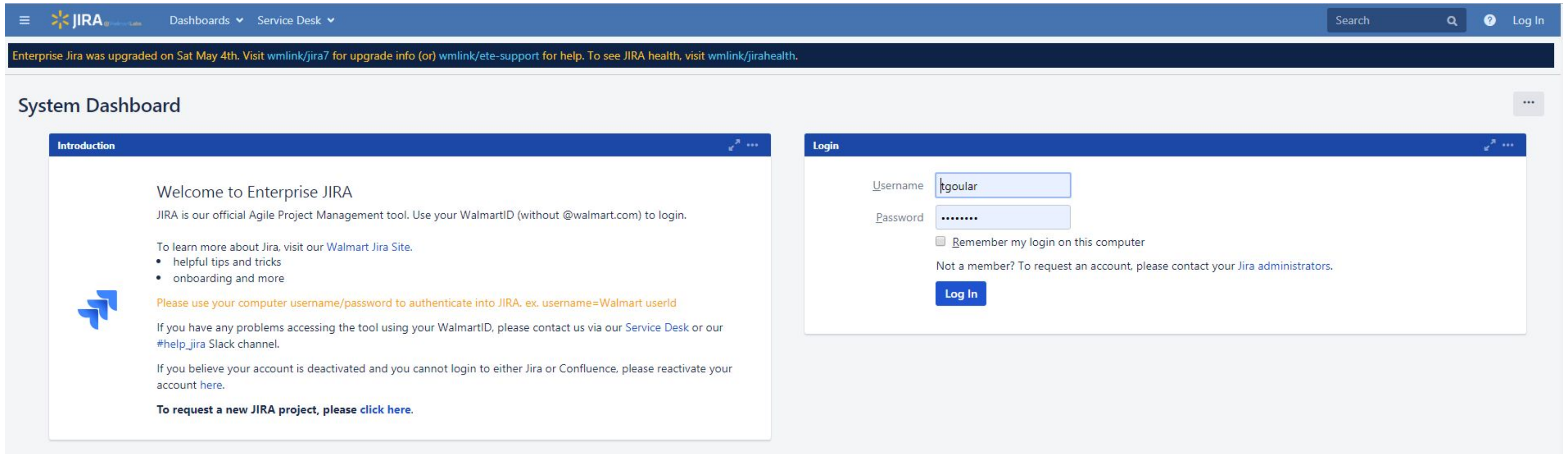
Sub-Task

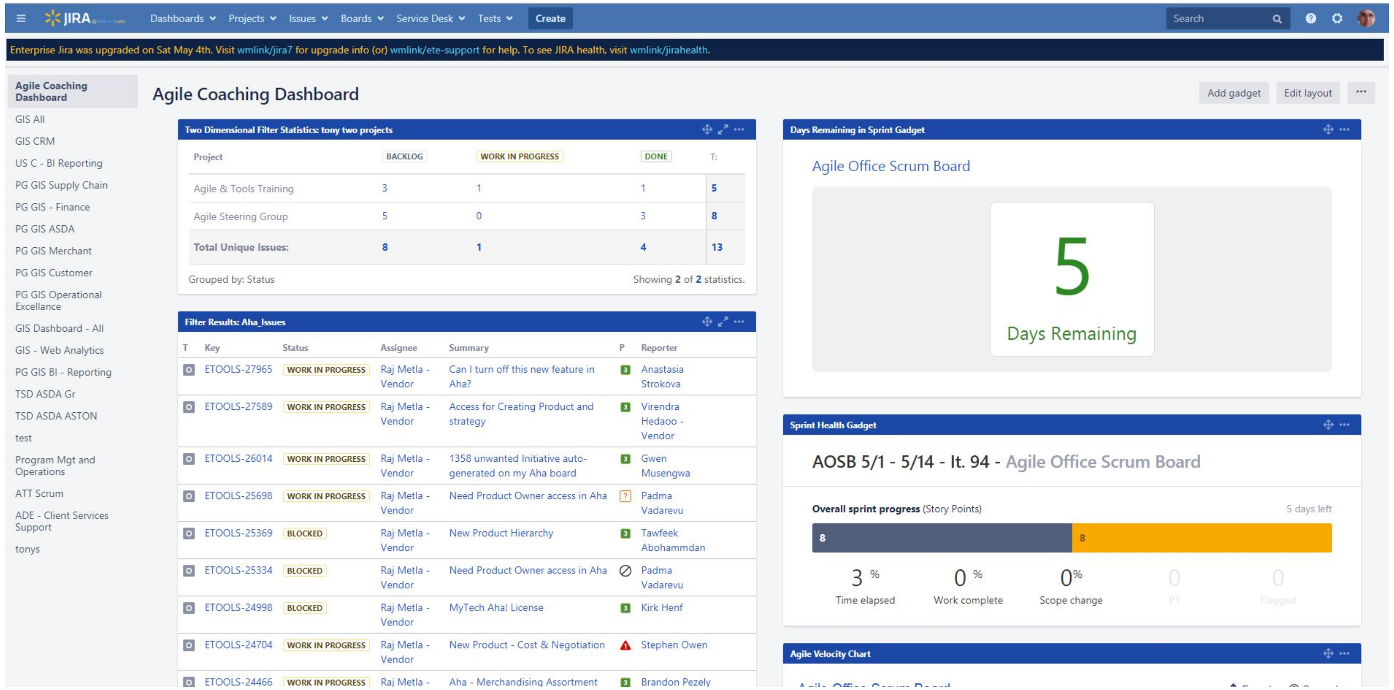
Now we know…

- What JIRA is ✓
- Organizational Structure ✓

~~Need new dashboard screen below~~

How to Log into JIRA





~~Need new screen for header below~~
~~Change the definitions to match header~~

Navigation Bar in JIRA

Navigator – Allows the user to jump between different Aps, JIRA, Confluence, Roadmap, etc

Dashboard – Allows high level view of project

Projects – List of all projects (Current, Recent or Search for a particular Project)

Issues - See the recent issues, search for issues and filters to see only my issues

Boards - see your recent boards that you have been in, search for boards and manage boards

Service Desk – Ability to create service portals

Tests - Multiple manual test cases can be imported

Create – Create Issues (Epics, Stories, Task, Defect, Deployment Request, Pre-condition, Test, Test Execution or Test Set

Search – text search within the JIRA Boards

? – will show you different help areas (JIRA Help, Keyboard Shortcuts)

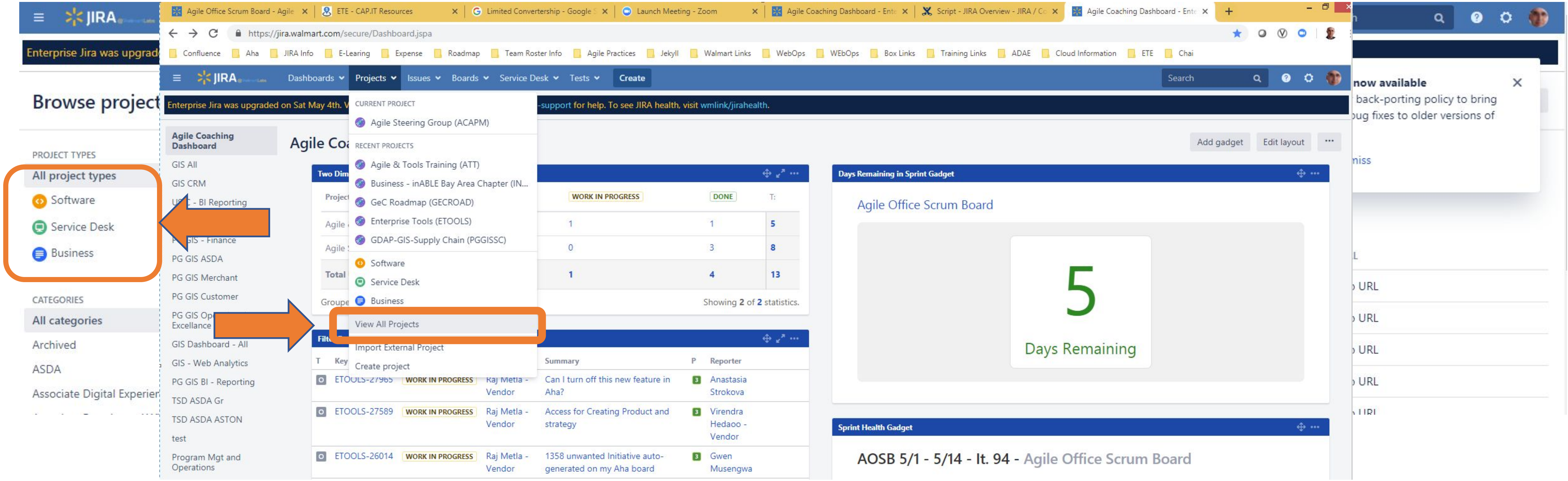
"*" - JIRA Administration

Profile – This will give your JIRA Home and also where you log out

~~Need new screens below~~
~~Add Project Types~~

Projects

- Access recent projects
- See all the projects and their key



Demo

JIRA

Now we know…

- How to log in ✓
- Use the Navigation Bar ✓
- Browse Projects ✓

Discuss purpose of a Story/Epic/Initiative

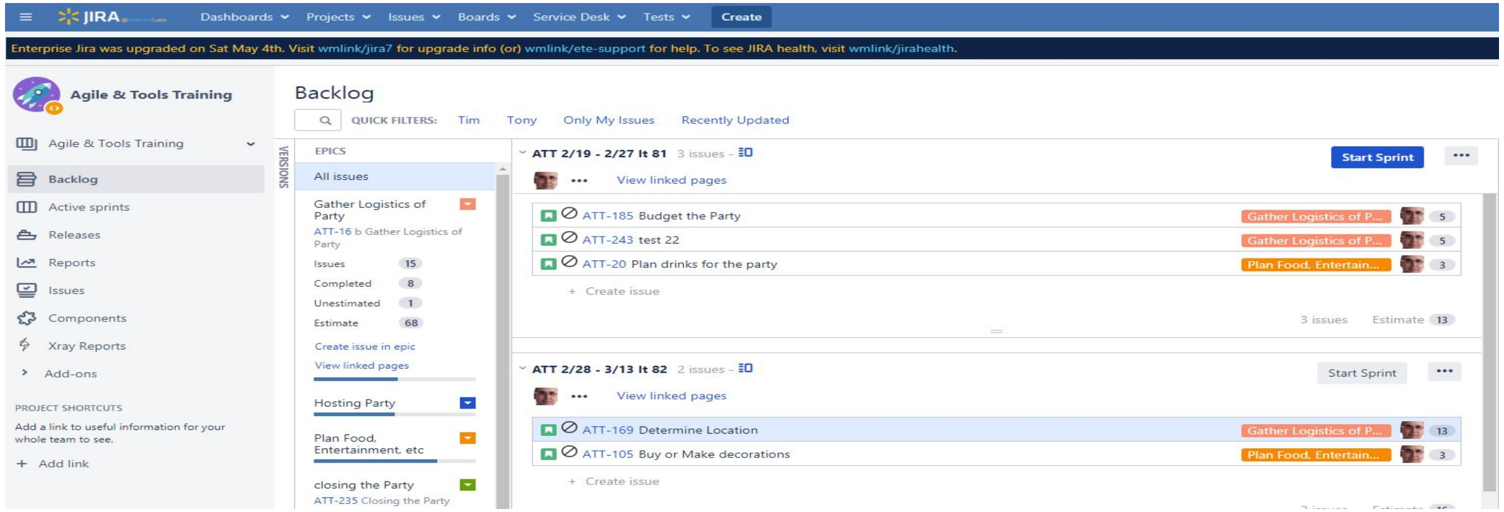
- Explain about a story and an Epic
- Do an overview of entering a story and an Epic in JIRA (Card Conversation and Confirmation)
    - Explain Sub-tasks
- Understand the benefits of sizing stories (Size, Complexity and Effort) as a team to help with velocity of the team

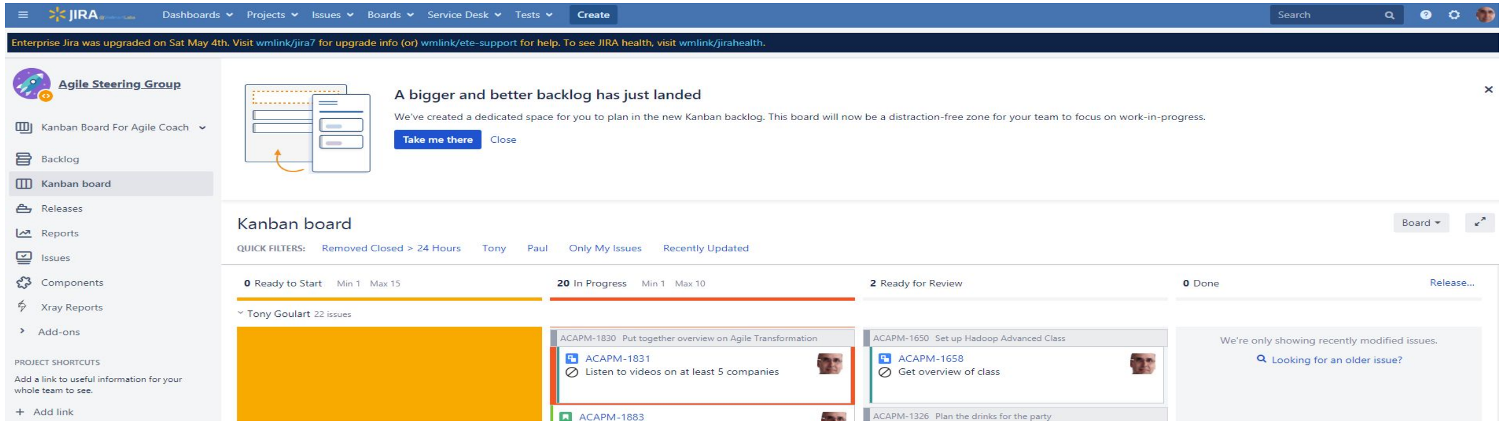
Agile Boards

~~Need new screen below~~

~~Show difference between Scrum and Kanban~~

- Prioritize the backlog
- Allows you to plan Sprint/Iteration
- Associate Epics and Stories
- Quick Filters
- Work in your Iteration/Sprint
- Team per Sprint Report
- Manage unfinished stories





Enterprise Tools Engineering & TechOps

Demo

JIRA

Add difference between Scrum and Kanban

Now we know…

- Prioritize the backlog ✓
- Plan your Iterations (Sprints) ✓
- Associate Epics and Stories ✓
- Quick Filters ✓
- Work in your Iteration/Sprint ✓
- Team per Sprint Report ✓
- Manage unfinished stories ✓

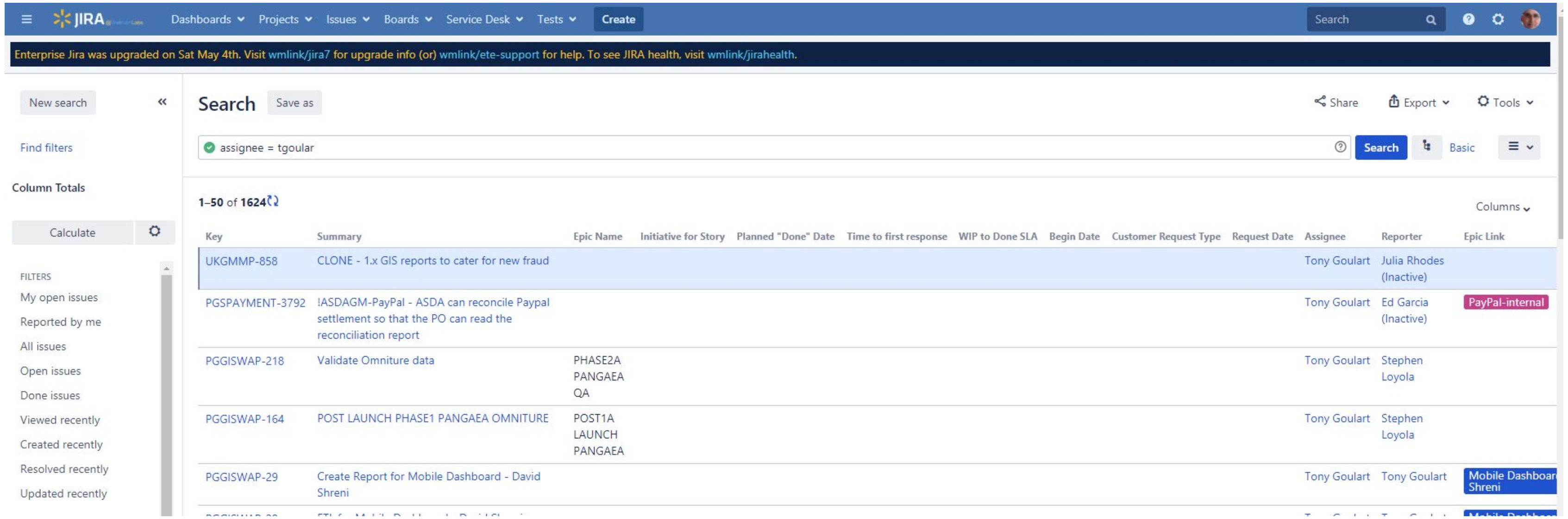
Issues

JIRA

~~Need new screen below~~

Issues

- Display all issues across projects
- Save your search as a filter
- Advanced Mode (JQL)



Demo

Now we know…

- Display all issues across projects ✓
- Save your search as a filter ✓
- Advanced Mode (JQL) ✓

We should also know…

Understanding of JIRA ✓

How to Plan ✓

How to Iterate ✓

Basic Reporting ✓

Helpful Links

Wmlink/helpjira

Confluence – search for "Useful JQL Queries for Project Managers"

Confluence Spaces People Calendars Browse Create

JIRA / Confluence Update / ... / JIRA FAQ

Useful JQL Queries for Project Managers

Edit Watch Share

Created by Stephane Nguyen, last modified by Tony Goulart on Jan 25, 2019

I want to display...	JQL Query
Unresolved issues assigned to me that are overdue	Assignee = currentUser() AND due < now() ORDER BY due asc
Unresolved issues in ATT assigned to me that are between a date	project = "Agile Steering Group" and "Planned \"Done\" Date" >= 2017-9-24 and "Planned \"Done\" Date" <= 2017-10-26 and assignee in (CurrentUser())
Issues in ATT project and in Backlog Status	project = "Agile & Tools Training" and status = Backlog
Issues in the ATT project and in the backlog blocked	project = "Agile & Tools Training" AND status = Blocked
Issues created one month ago by me	created >= endOfMonth(-1) AND reporter = currentUser()
STORIES in iteration 78 (in my project)	1. project = "Agile & Tools Training" AND issueType = story AND sprint = "ATT 2/19 - 2/28 It 78" 2. clicking on view in issue navigator against your sprint , automatically opens up the issue navigator wth JQL query pre-built with results. IT 14 - 5.31 to 6.13 12 - View in Issue Navigator 34 28 0 30/May/13 5:16 PM • 12/Jun/13 5:16 PM
Issues in WIP over 10 days	project = "Agile & Tools Training" AND status changed to "Work In Progress" before endOfDay("-10d")
ATT Stories not completed that were worked on before Dec 21	project = "Agile & Tools Training" and issuetype=Story and status != Done and status was "Work In Progress" before "2018/12/21"

Thanks

JIRA

Helpful Links…

Link to Video
I have created a video from the presentation that was just given – the link is to the right

1 Coming Soon

Link to Presentation
This presentation can be found on the link to right

2 Link to Presentation

Link to JIRA Overview
Here is a board that I created to help with questions you may have – please find link to right

3 JIRA Overview

JQL Cheat Sheet
Here is a link to the JQL Cheat Sheet that I created, please find link to the right

4 JQL Cheat Sheet

Views of a Scrum Board

Backlog

- The Backlog view will show a holistic view of all of the issues in your backlog.

Active Sprint

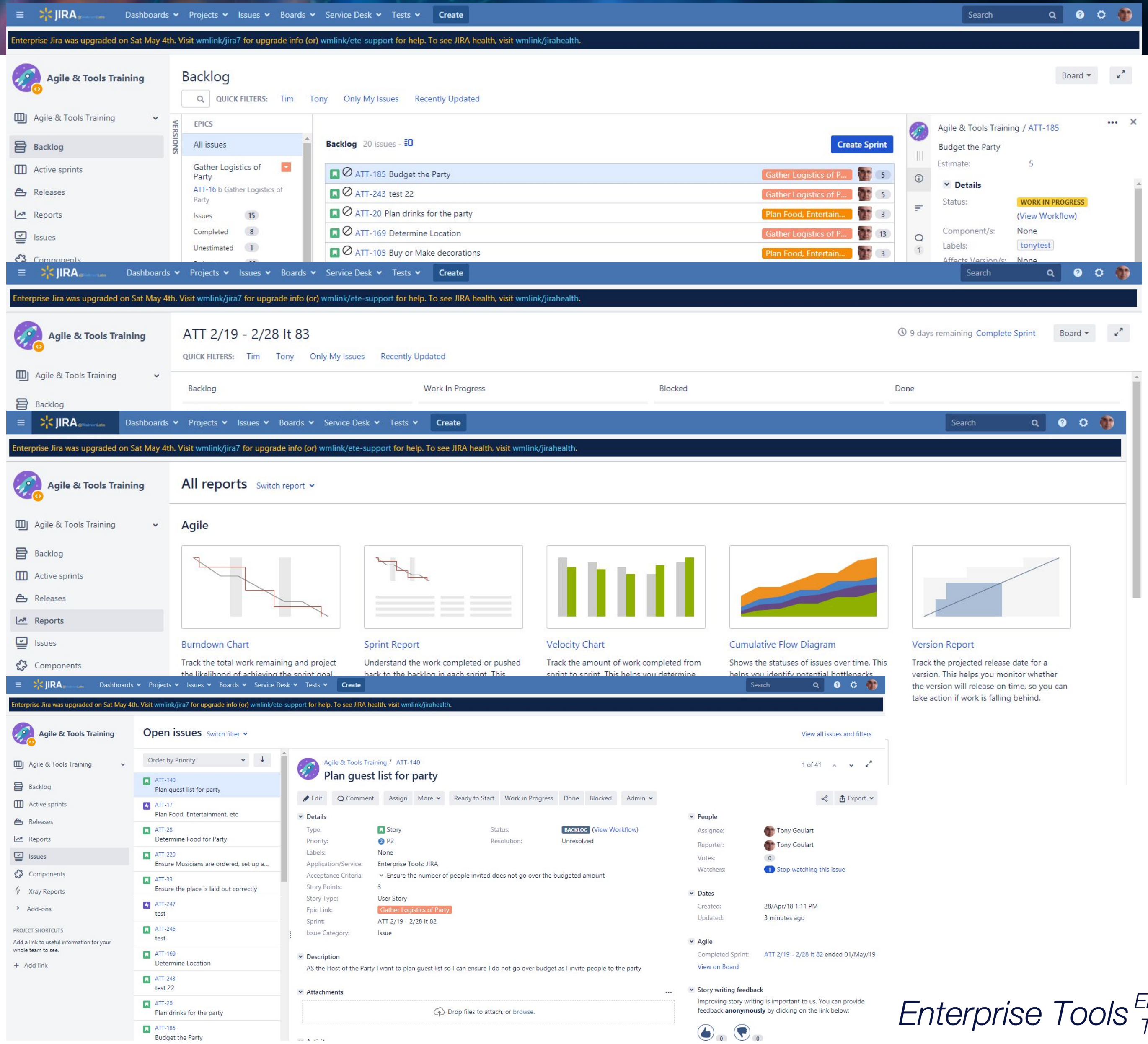
- The Active Sprint will show the issues that you have in your current sprint.

Reports

- There are many reports to choose from.

Open Issues

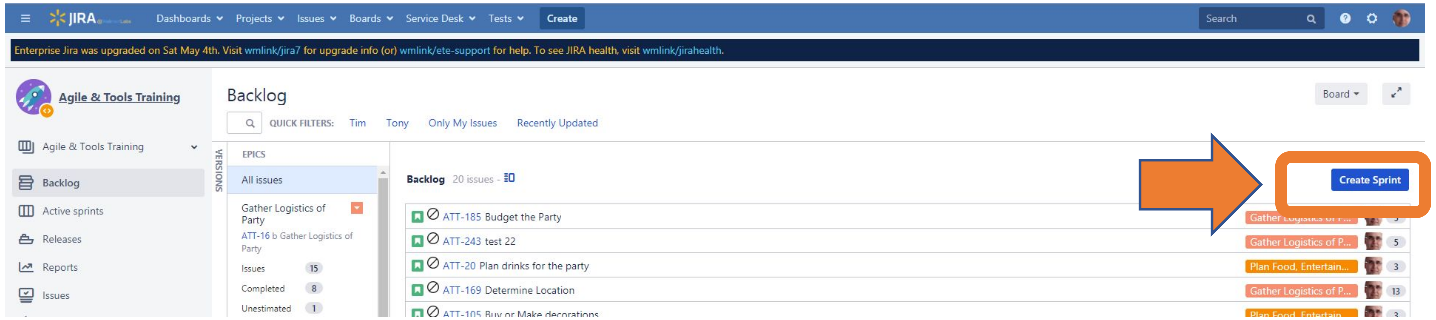
- Open Issues list shows all the items that you have created



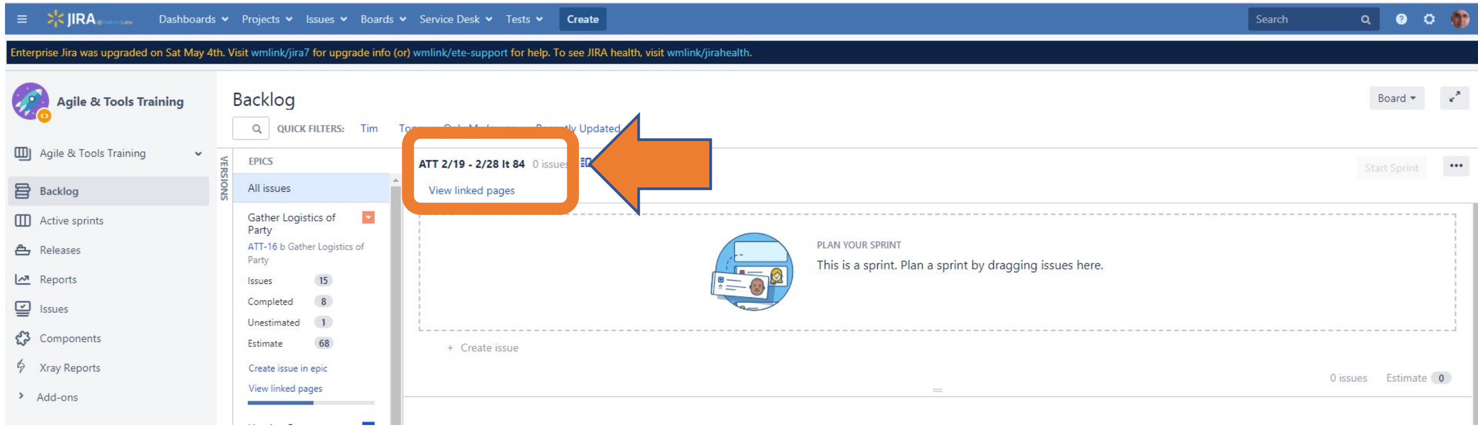
Enterprise Tools Engineering & TechOps

How to create a Sprint in JIRA

To create a sprint within JIRA you will need to go to the Backlog Screen and Click on "Create Sprint"



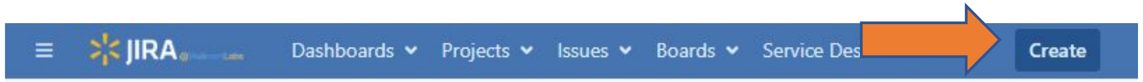
After clicking "Create Sprint" you will see that a sprint is created



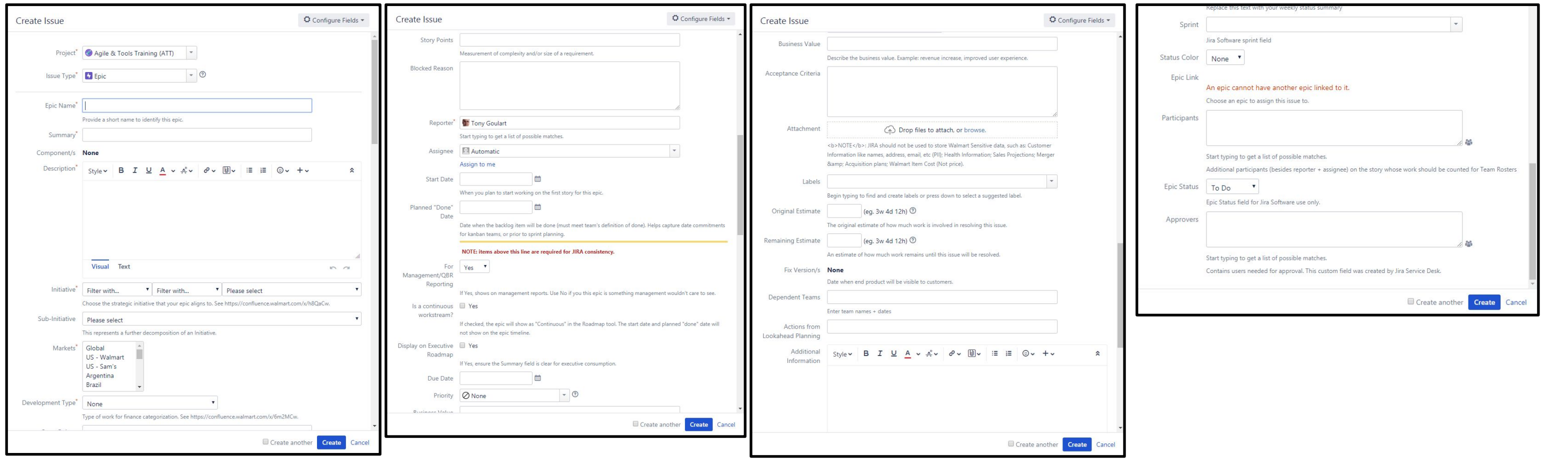
You can then click on the Sprint Name to edit and make it more user friendly

How to Create an Epic in JIRA

1. Ensure that you are in the correct Project in JIRA
2. Click on the Create Button
3. Fill out the following fields (minimum):

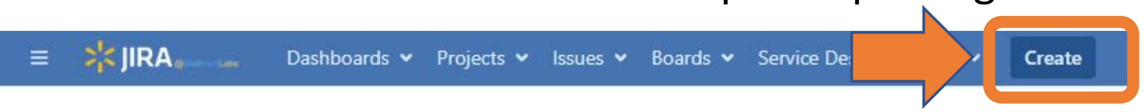


- Project
- Issue Type
- Epic Name
- Summary
- Description
- Initiative
- Markets
- Development Type
- Start Date
- Planned Done Date
- For Management/QBR Reporting
- Display on Executive Roadmap



How to create a Story in JIRA

Click on the Create Button at the top of Top Navigation



Enter the information needed to create story:

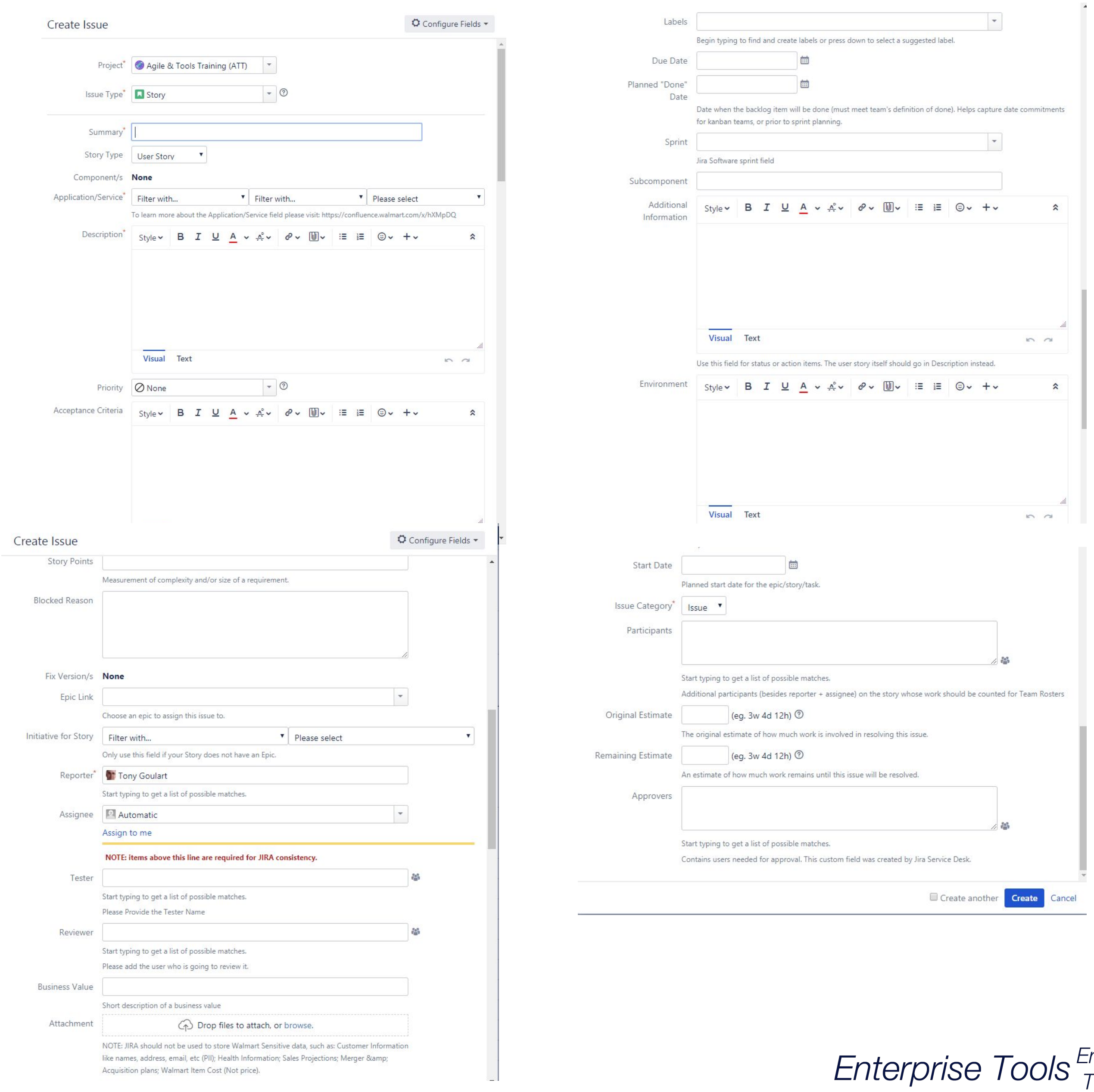
Ensure Project is correct

Issue Type = Story

Summary

Epic Link or Initiative if you do not have Epic

Click Create



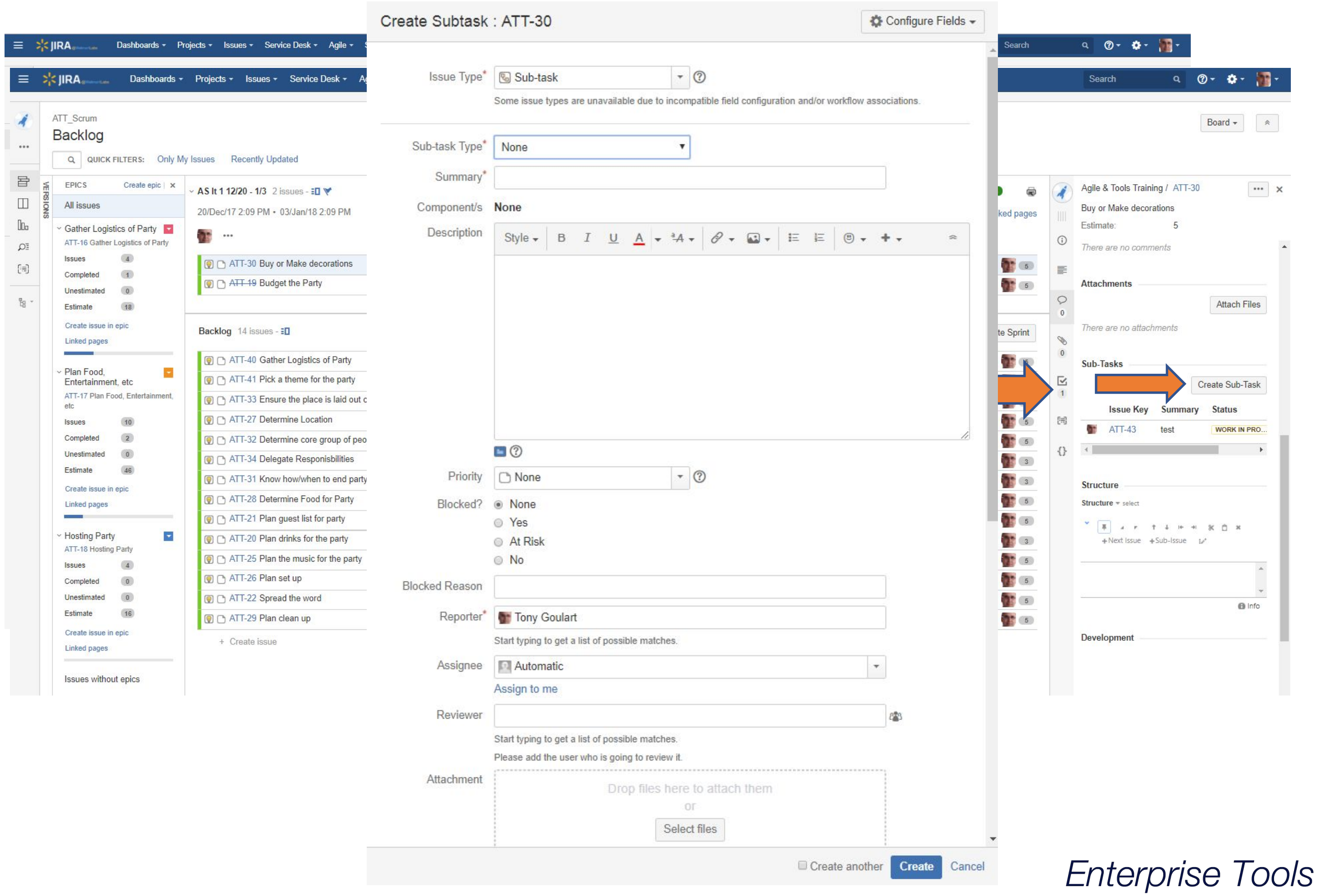
How to create a Sub-Task in JIRA –**stopped here**

Backlog View Click on a story and the right hand navigation will show the story

You can click on either the checkbox in the Right Hand Nav or scroll down and click on Create Sub-Task

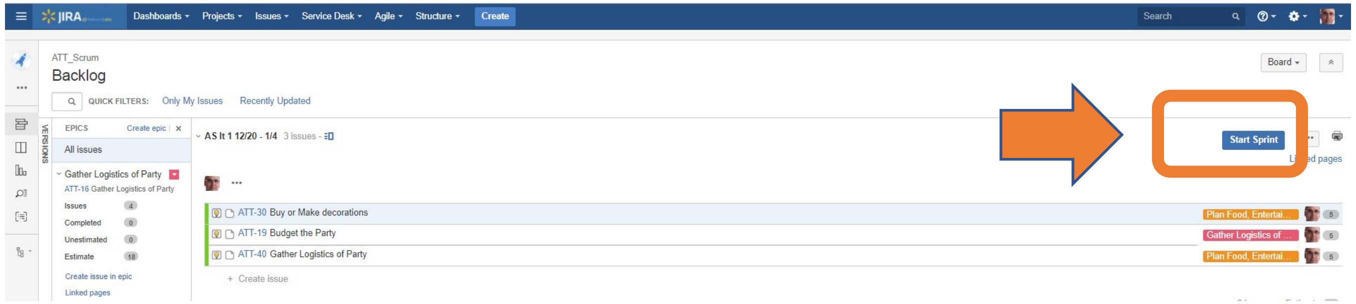
A pop up will open where you must enter:

- Summary
- Then click “Create”



How to Start a Sprint in JIRA

to the Backlog Screen in JIRA
If there is no sprint started you can click on “Start Sprint” highlighted in the Orange Box/Arrow



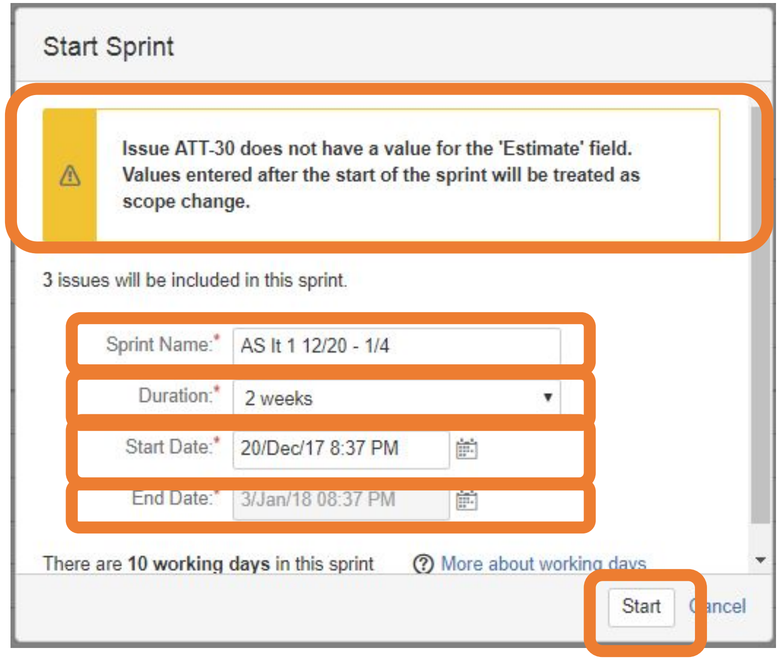
After “Start Sprint” is clicked the following pop up will show –

- If there are any stories that do not have values for Estimate you will get this pop up

Please ensure there is a

- Sprint Name (correct format)
- Duration of your Sprint
- Start Date of your Sprint
- End Date of your Sprint

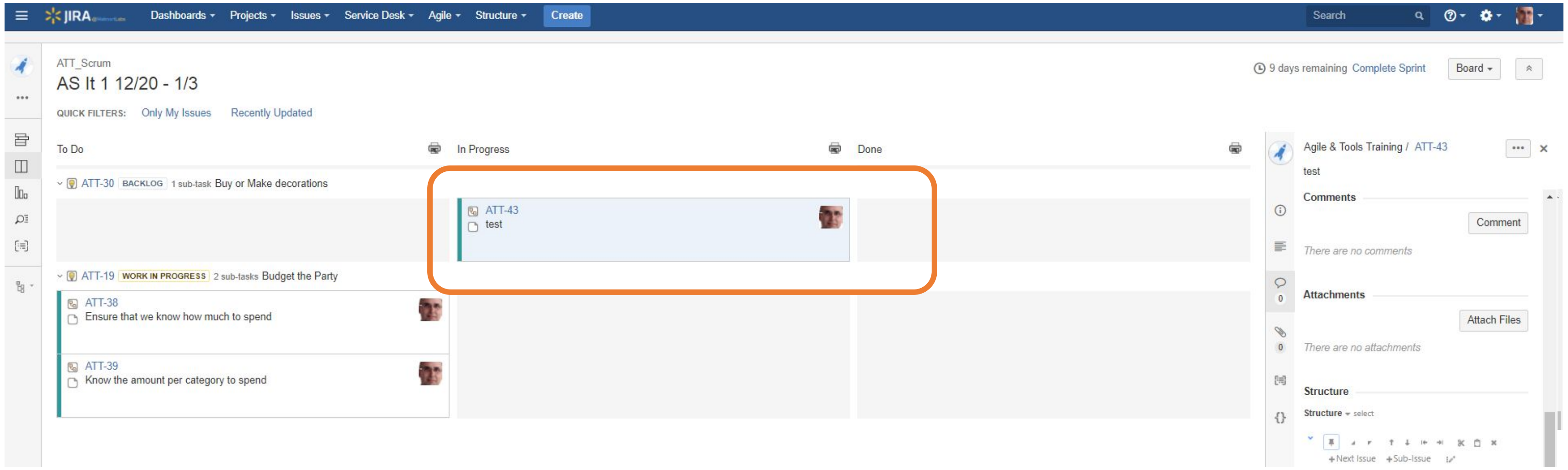
Then Click Start



How to Move Items across a JIRA Board

move and item across the JIRA Board you can drag and drop

Move your mouse over the issue on the board, hold the left click down and move the issue



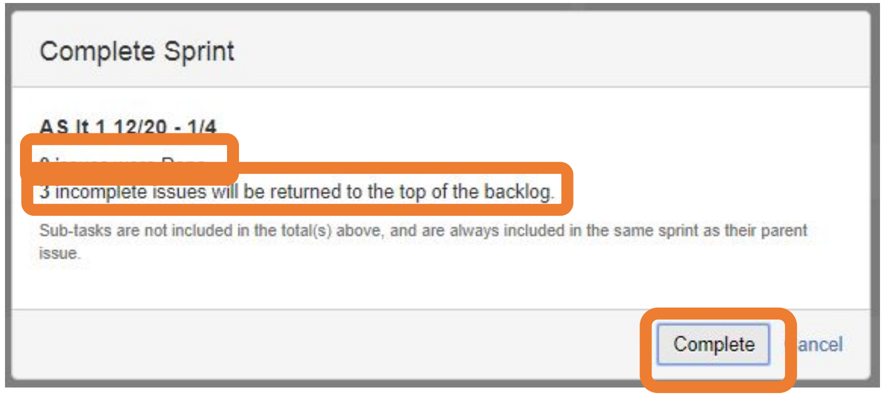
After moving the issue, you release the left click and your issue will be moved

How to End a Sprint in JIRA

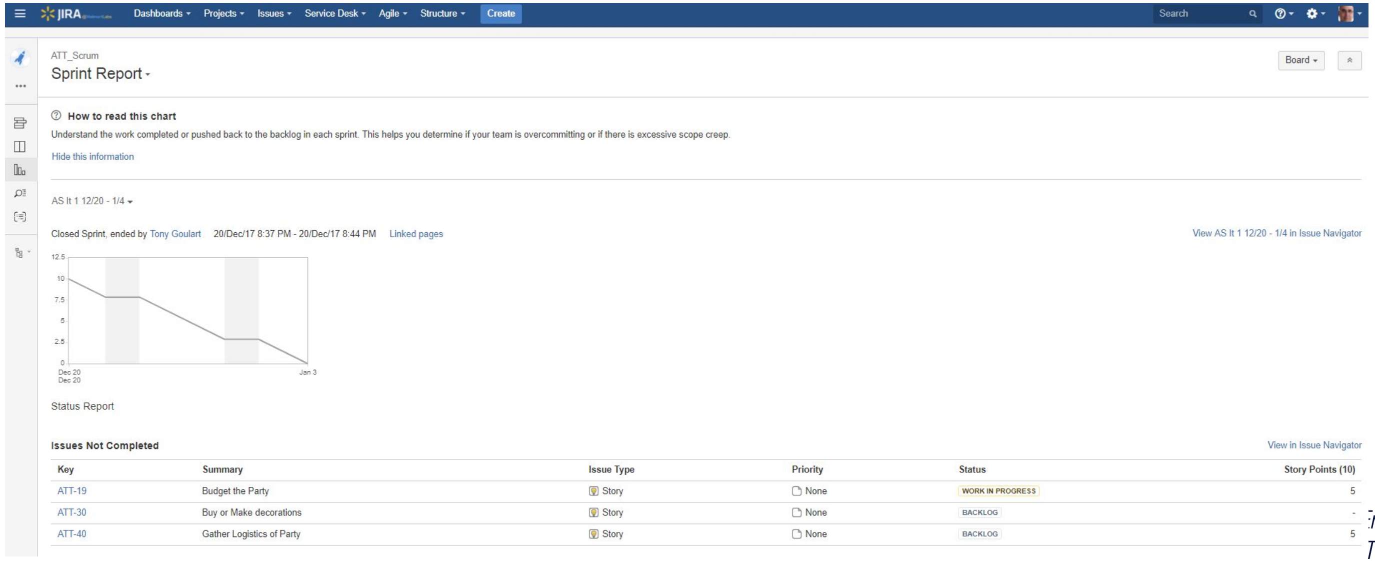
Go to the Active Sprint Screen in JIRA

If you have access you will see the "Complete Sprint" in the right hand side in blue that you can click on to close the sprint

- After "Complete Sprint" is clicked the following pop up will show –
  - Tells you the number of Stories that are "Done"
  - Shows you the number of stories that will be returned to the top of the next sprint (or backlog if there is not a sprint in the backlog)
  - Then click "Complete"



- Finally you will see the Sprint Report for the sprint you just closed. It will show the number of items completed and not completed. .
- You will find the stories that were not completed on the top of the next sprint if there is one or at the top of the backlog if there is not a sprint in the backlog



Clone an issue in JIRA

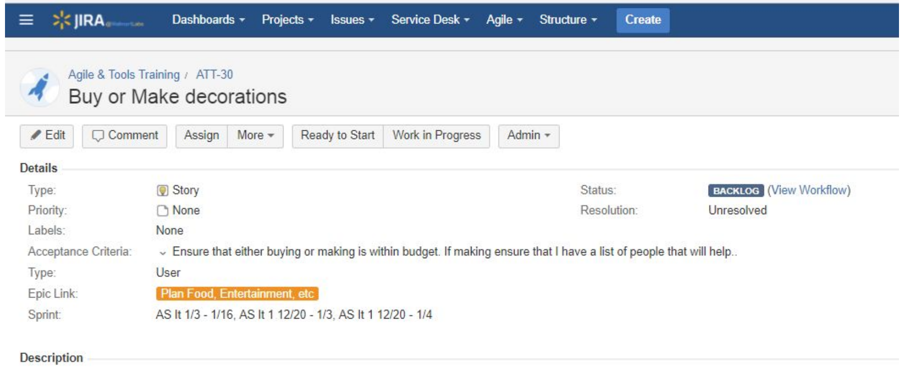
- Cloning an issue in JIRA lets you quickly create a duplicate of an issue within the same project. It creates a replica of the original issue retaining the following information:
  - Summary
  - Description
  - Assignee
  - Environment
  - Priority
  - Issue Type
  - Security
  - Reporter
  - Components
  - Project

- Concerns on Cloning
  - Naming convention on the following items have a "Clone" at the beginning of the following
    - Summary – if summary was "This Issue" the cloned issue would state "Cloned This Issue"
    - Sub Tasks – if sub tasks was "Work on this" the cloned sub tasks would be "Cloned Work on this"
  - Issue Links
    - Show the item is cloned from in the story
  - Sprint Information will also be cloned

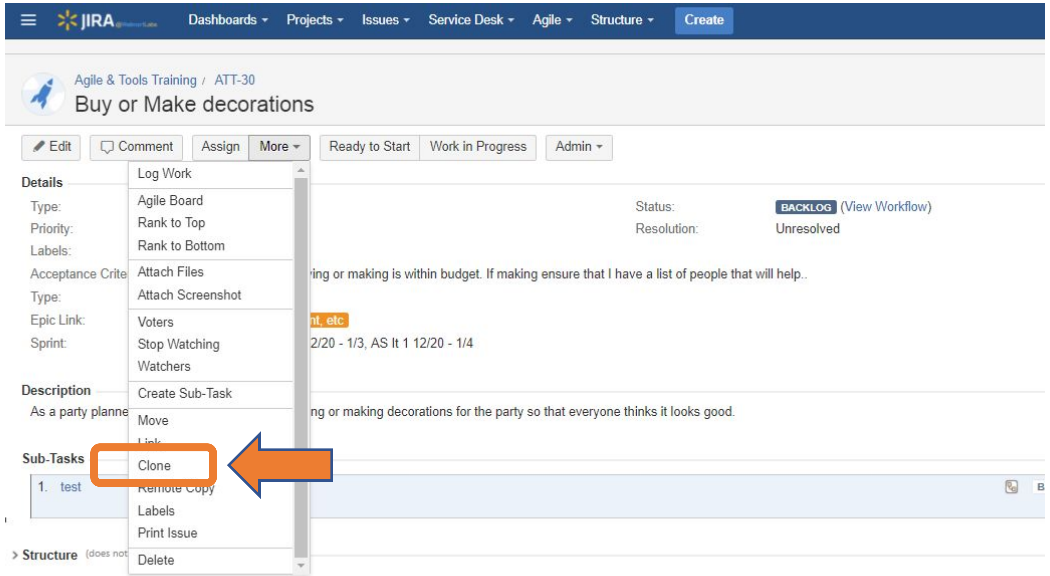
- I have found that best practice is to update the Summary and Sub Tasks along with removing the issue link and removing the Sprint Information after cloning an issue.

Process to Clone an Issue

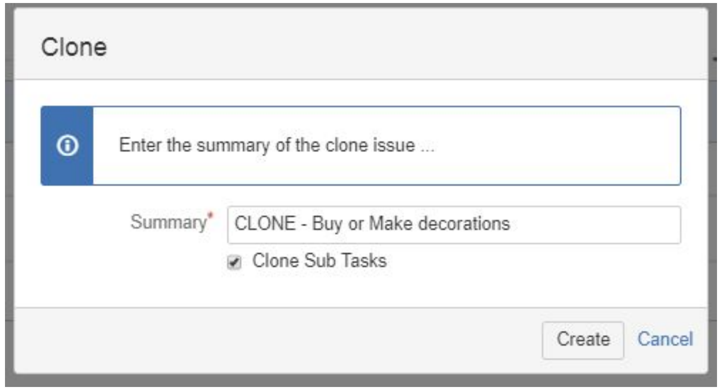
- Open in JIRA the Issue you want to Clone



- On the drop down for More, click on Clone



- A popup appears and you can click "Create" to Clone your issue (You have the option to Clone the Sub-Tasks if desired).

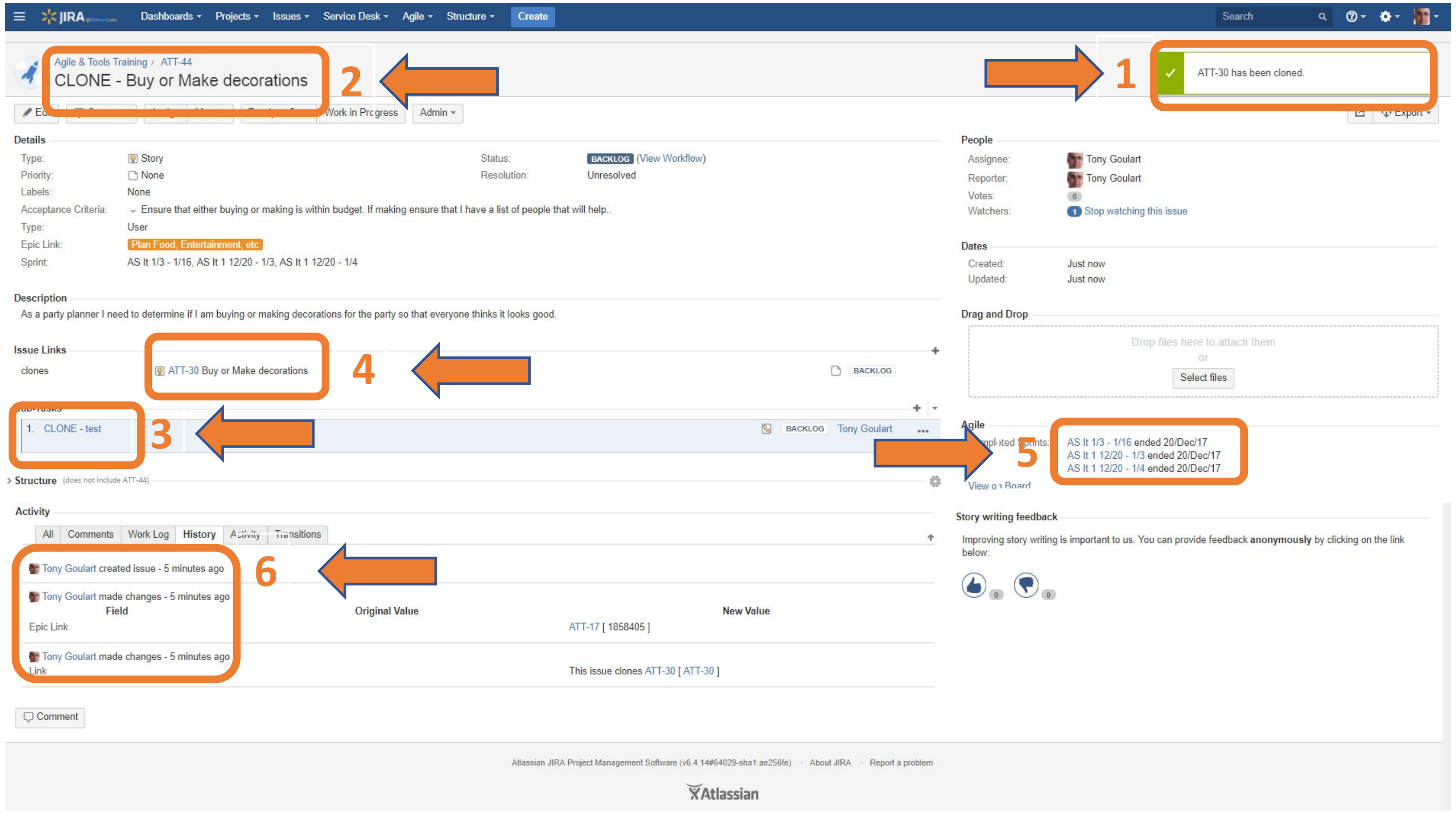


Enterprise Tools Engineering & TechOps

Process to Clone an Issue (continued)

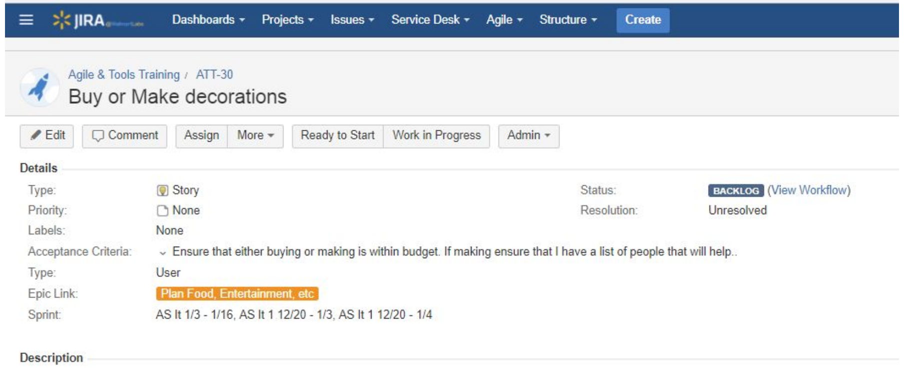
After you have cloned your issue it should open up similar to the right:

1. On the right hand side of the screen you will see that the issue has been cloned
2. You will note that the Summary has a “CLONE” stated prior
3. You will note that if you cloned your Sub-Tasks they will also have a “CLONE” in front of the sub-task description
4. You will note that on the Issue Links will show what item was cloned from
5. You will note the Completed Sprints the issue you cloned was in
6. You will see the history of the cloned issue if you click “History”

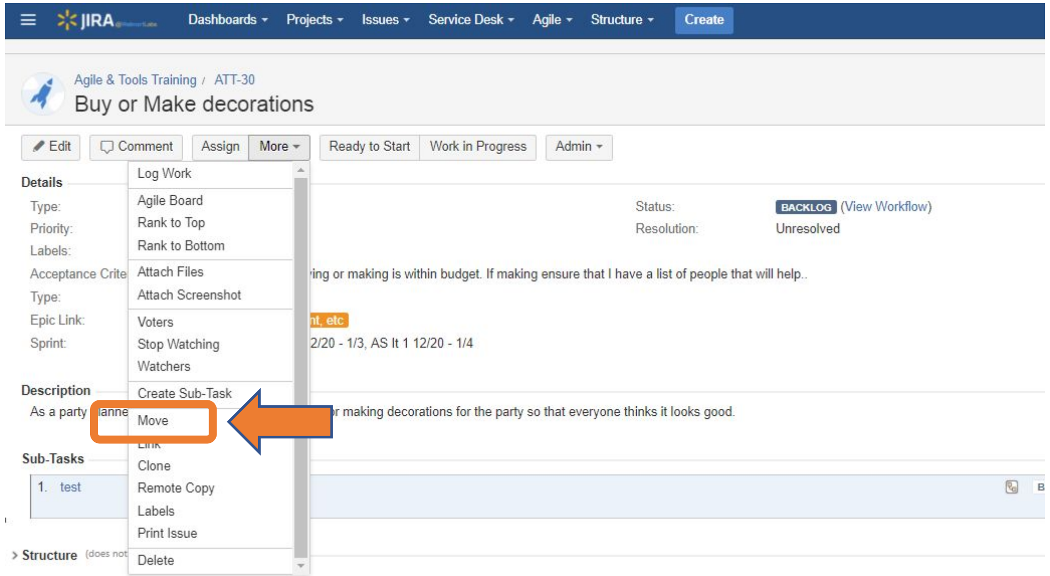


Moving an issue in JIRA

- Open in JIRA the Issue you want to Move

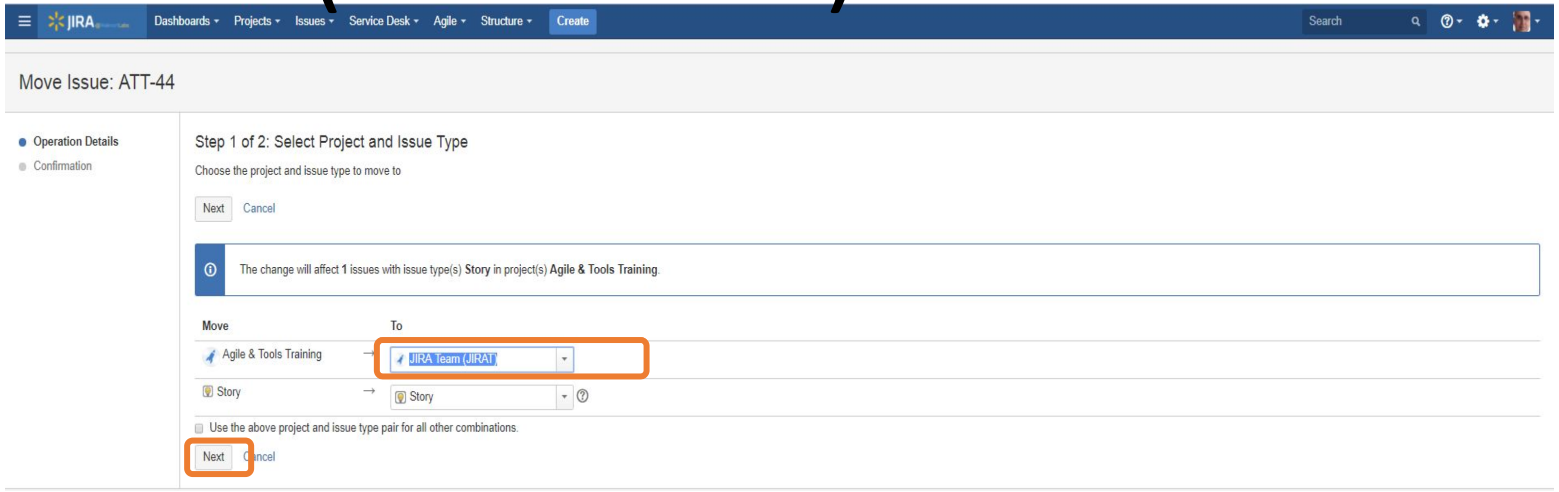


- On the drop down for More, click on Move

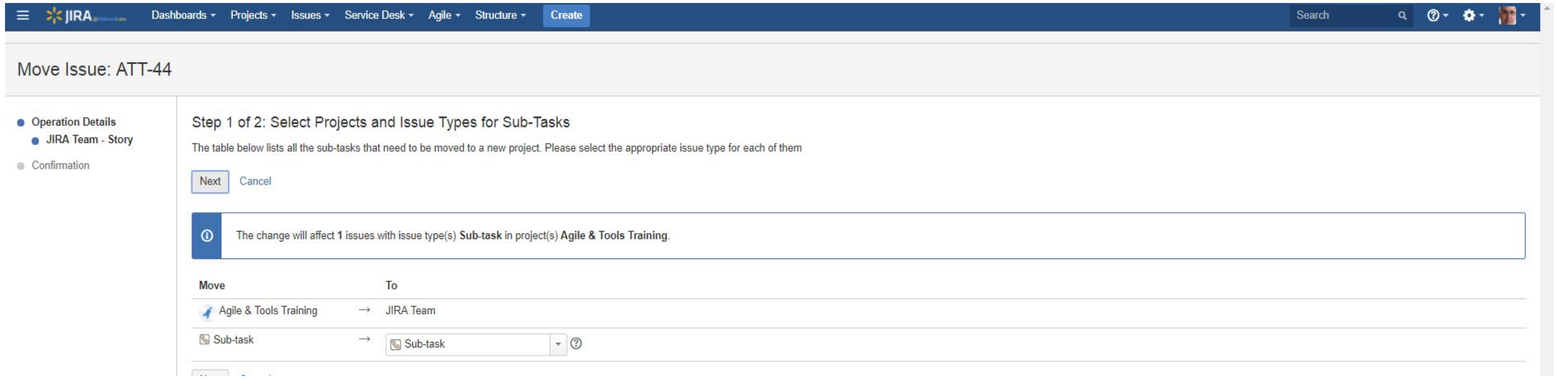


Process to Move an Issue (continued)

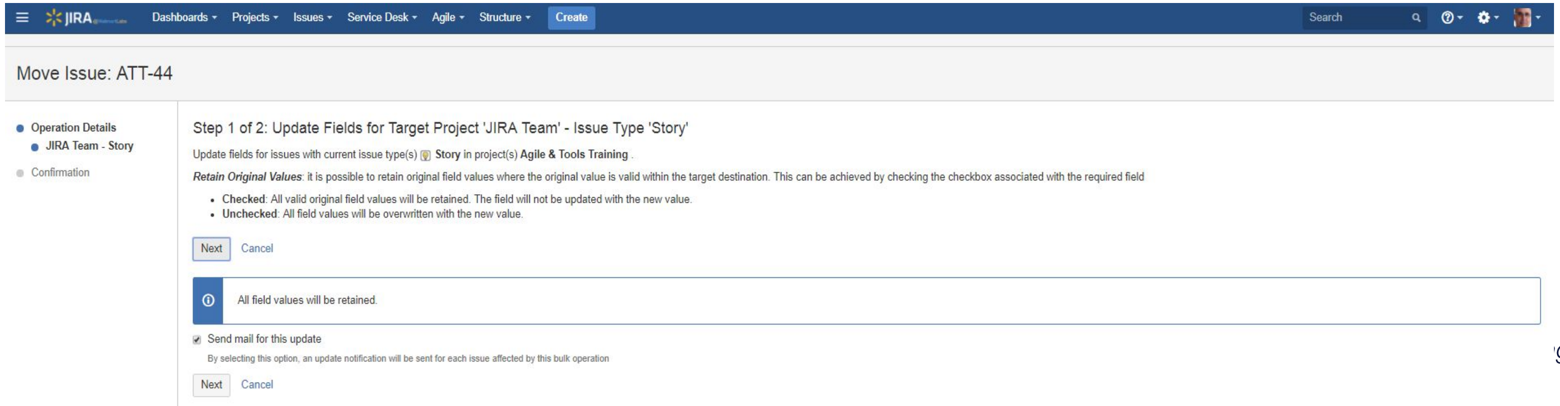
- Enter the Board that you would like the issue moved to



- If you have sub-tasks you will see a screen to move the sub-tasks.

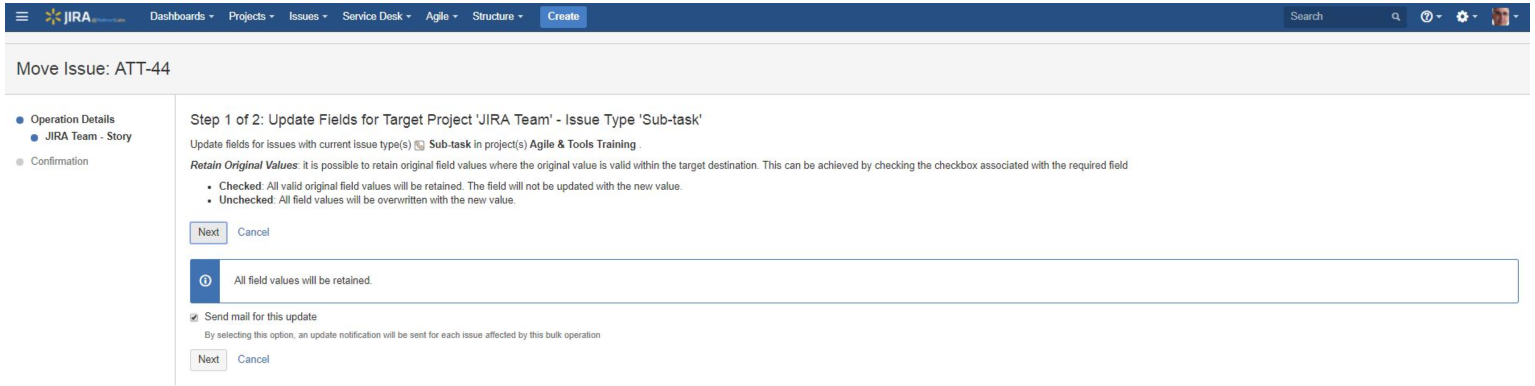


- The next screen will let you know that the "Story will be moved.

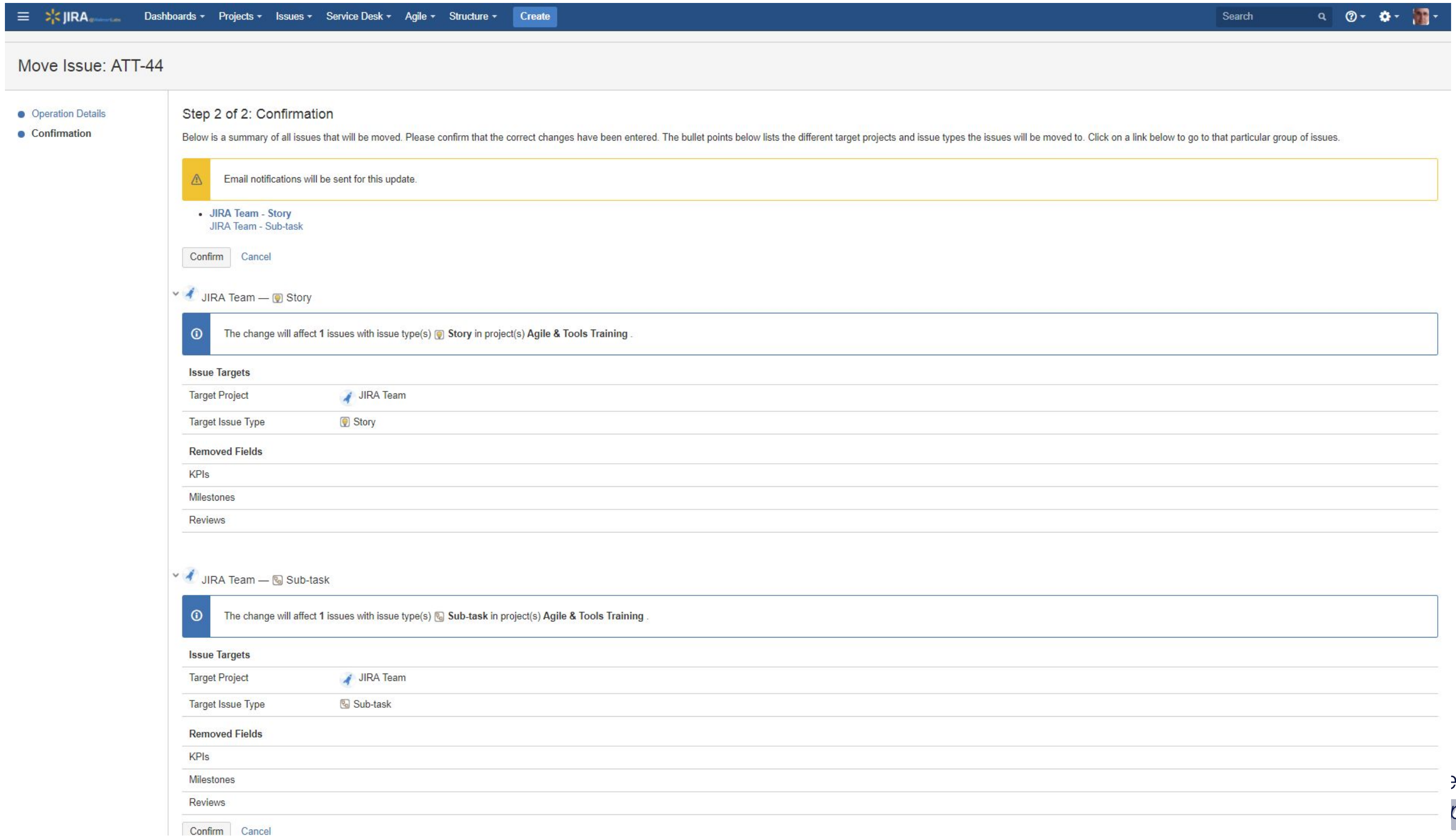


Process to Move an Issue (continued)

- The next screen will let you know that the Sub-Tasks will be moved



- The next screen will give you a confirmation of what you will be moving from and to

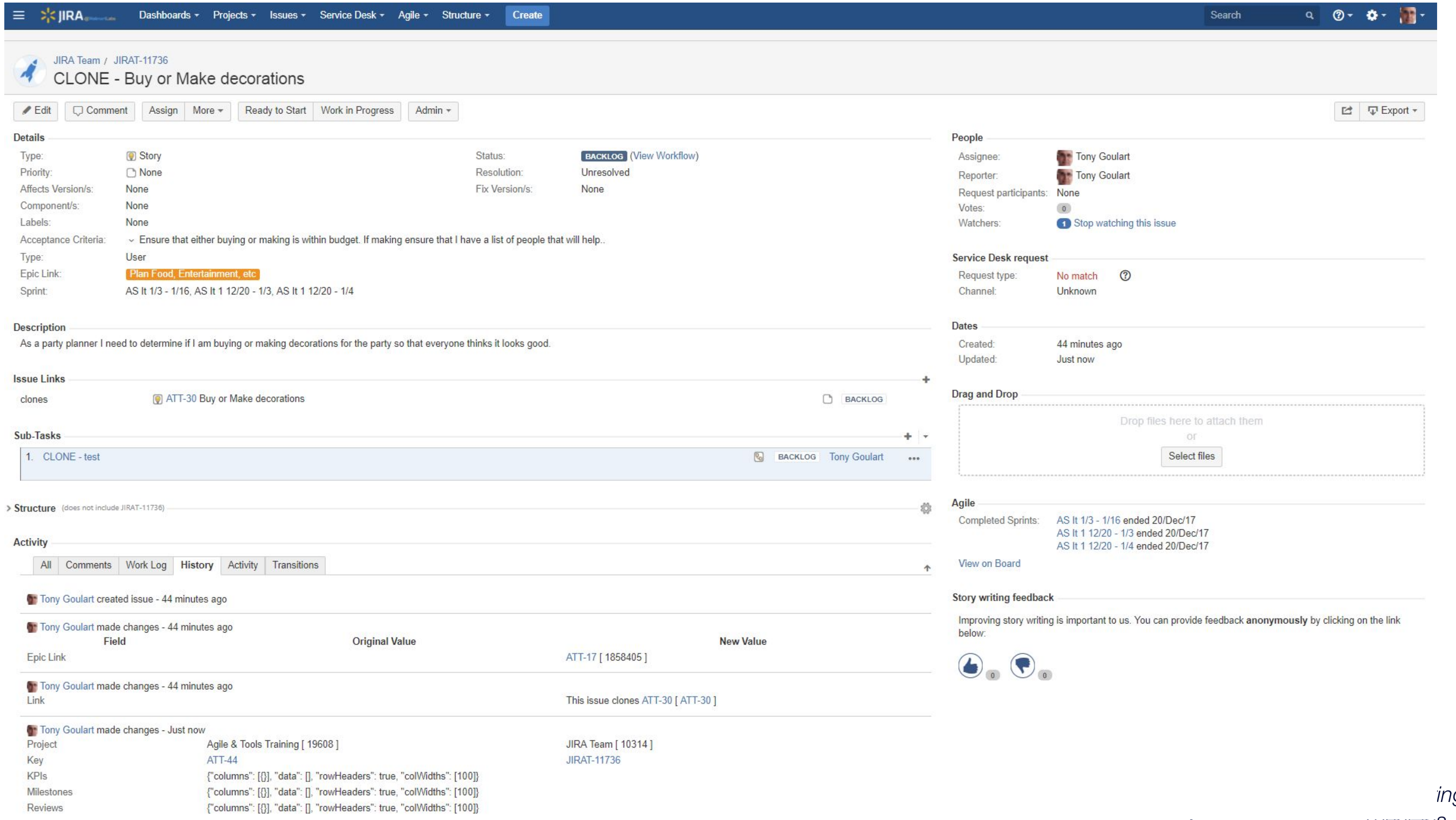


ering
bs

Process to Move an Issue (continued)

- The next screen will let you know that the items are migrating and click to Acknowledge

- The next screen will show you the issue that was moved to your new board



Similarities between Scrum and Kanban

Both are Lean and Agile
Using the Pull System

Both limit WIP

Both use transparency to drive process improvement

Both focus on delivering releasable software early and often

Both are based on self-organizing teams

Both require breaking the work into pieces

In both cases the release plan is continuously optimized based on empirical data (velocity and lead-time)

Differences between Scrum and Kanban

Roles

Scrum	Kanban
Cross-functional teams prescribed	Cross-functional teams optional. Specialist teams allowed.
Prescribe 3 roles (PO/SM/Team)	Doesn't prescribe any roles

Backlog

Scrum	Kanban
Items must be broken down so they can be completed within 1 sprint	No particular item sized is prescribed
A sprint backlog is owned by one specific team	A Kanban Board may be shared by multiple teams or individuals
Prescribes a prioritized product backlog	Prioritization is optional

In Process

Scrum	Kanban
Time boxed iterations prescribed	Time boxed iterations optional. Can have separate cadences for planning, release and process improvement. Can be event driven instead of time boxed.
Team commits to a specific amount of work for this iteration.	Commitment optional
Cannot add items to ongoing iteration	Can add new items whenever capacity is available
A scrum board is reset between each sprint	A Kanban board is persistent

Reporting

Scrum	Kanban
Uses Velocity as a default metric for planning and process improvement	Uses Lead Time as default metric for planning and process improvement
Burndown chart prescribed	No particular type of diagram is prescribed
WIP limited indirectly (per sprint)	WIP limited directly (per workflow state)
Estimation prescribed	Estimation optional