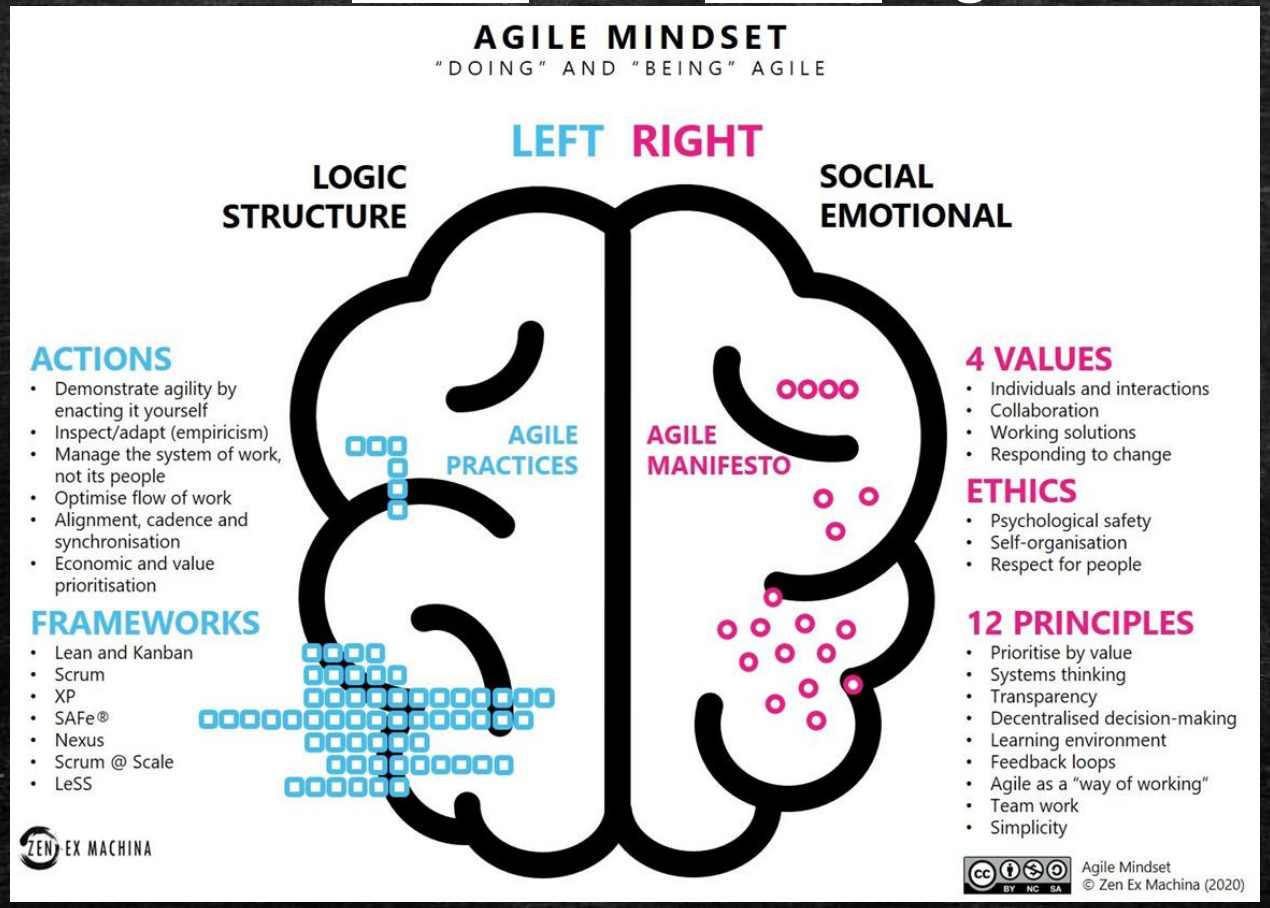
Equanimity at Work

Agile Playbook

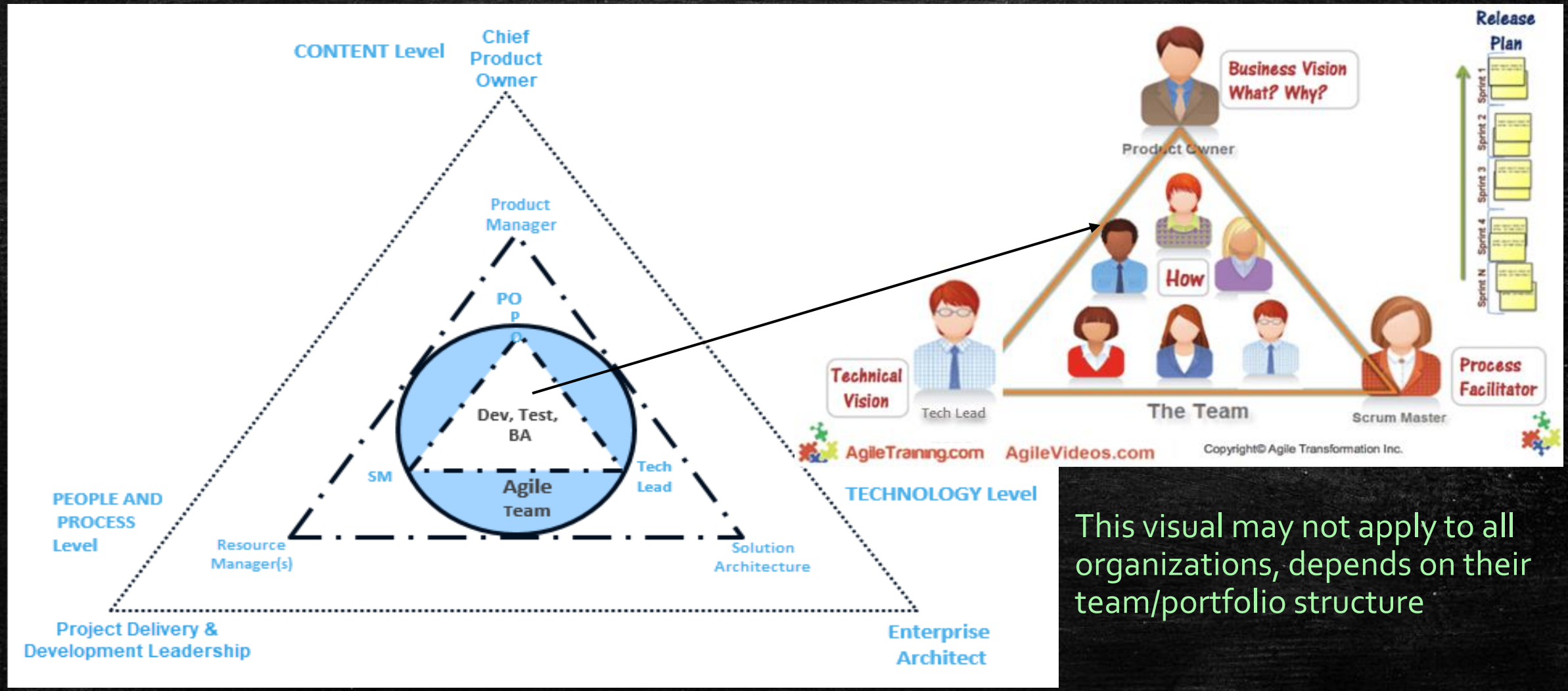
Table of Contents

Agile Mindset

Doing and *Being* Agile



Team Formation



This visual may not apply to all organizations, depends on their team/portfolio structure

Team Formation - Roles & Accountabilities

Product Owner

- Understands product vision
- Help create a product roadmap
- Creates & maintains a prioritized backlog representing Stakeholder & Customer needs as related to the product roadmap
- Ensures the team is working on the highest priority items
- Ensures that the team understands the needs of each backlog item being worked
- Creates & approves the content of Acceptance Criteria for each backlog item
- Verifies that related developed functionality meets the needs of the customers and accepts each item as "Done"
- Provides demonstrations of functionality to the customer for feedback
- Assist in the creation of Risk Mitigation plans
- Orchestrates the removal of escalated impediments

Provides: Vision & Goals (Sprint & Release), Customer's Voice, Product Functional Direction & Guidance, Enablement, Empowerment, Clear Business Requirements, Mentoring

Scrum Master Accountabilities

- Manages the Agile process
- Supports backlog creation and maintenance
- Ensures team ceremonies are facilitated effectively and efficiently
- Guides the team to stay true to the Agile Manifesto and follow the Agile Principles
- Protects the team from distractions
- Orchestrates the removal of impediments
- Ensures that all is in place for a smooth demonstration of functionality to the stakeholders and customers
- Radiates the team's progress to interested parties
- Addresses relationships using appropriate conflict resolution, partnering with resource managers and ensuring good relationships between the team, Product Owner, and Stakeholders and customers

Provides: Transparency, Coaching, Teaching, Mentoring, Empowerment, Encouragement

Technical Lead Accountabilities

- Help identify, and provide input to, new items
- Provides insight to prioritization & technical dependencies
- Help identify items that are no longer needed
- Focuses themselves and team on the highest priority items
- Help create a high-quality product and helps ensure that the acceptance criteria is met for each related user story
- Participates in facilitating demonstrations of the functionality to stakeholders & customers, in order to get their approval and valuable feedback
- Takes the lead in designing the technical solution
- Accountable to the Technical Debt backlog
  - Creates and prioritizes
  - Works with PO to include items in the product roadmap

Provides: Technical Vision, Direction & Guidance, Coaching, Mentoring, Enablement, Empowerment

Team Formation - Roles & Accountabilities

Business Analyst

- Elicits specific needs and requirements for each user story
- Achieves a common understanding of business requirements for each user story
- Selects and completes the highest priority backlog items
- Completes functional testing and validation of the solution
- Demonstrates the solution and functionality to the stakeholders and customers
- Works ahead to elicit details of requirements for upcoming user stories
- Creates the content of Acceptance Criteria for each story backlog item
- Creates & maintains a prioritized backlog of features and stories representing Stakeholder & Customer needs as related to the product roadmap

Provides: Business and Technical Translation, Customer's Voice, Clear Business Requirements, Mentoring, Enablement, Empowerment

Developers

- Identify, and provide input to, new items
- Provide insight to prioritization and technical dependencies
- Identify items that are no longer needed
- Select and complete the highest priority items
- Develop a sound and effective technical solution
- Create a high-quality product
- Ensure that the acceptance criteria is met for each related user story
- Identify and create technical debt backlog items
- Regularly prepare and facilitate demonstrations of functionality to internal stakeholders and customers, in order to get their approval and valuable feedback

Provides: Sound & Effective Technical Solutions, Teaching, Mentoring

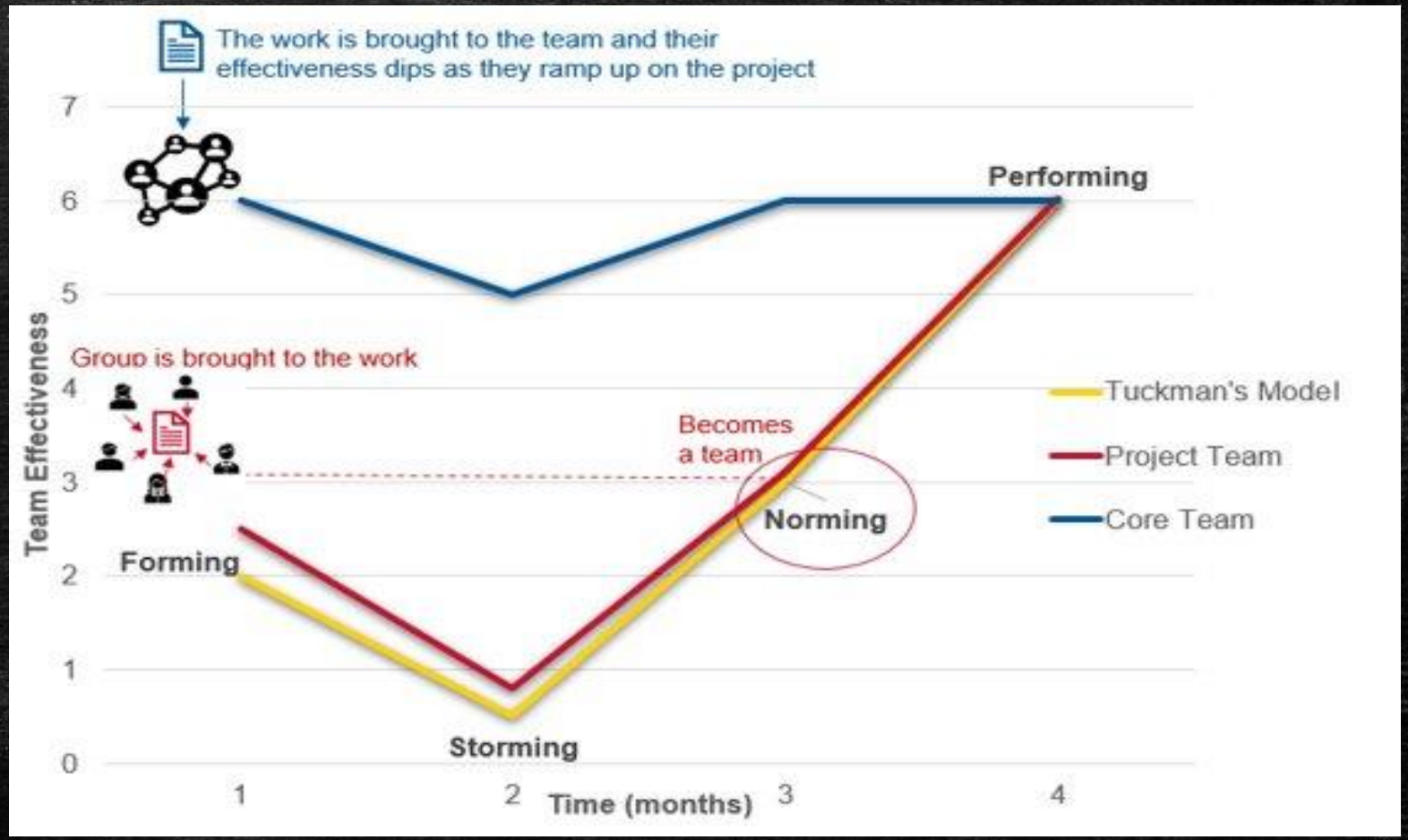
Quality Analyst

- Provide insight to prioritization
- Identify items that are no longer needed
- Create, clarify, and finalize acceptance criteria for each backlog item
- Create test cases from Acceptance Criteria so that the solution can be tested and verified as a high-quality product
- Execute test cases and verify results to help produce a high-quality product
- Complete multiple levels of testing
- Identify, create and execute multiple types of automated tests
- To prepare and facilitate demonstration of the functionality to internal Stakeholders and Customers

Provides: High Quality Solutions, Maintenance of a High-Quality Product, Teaching, Mentoring

Team Formation – Stable vs. Project Teams

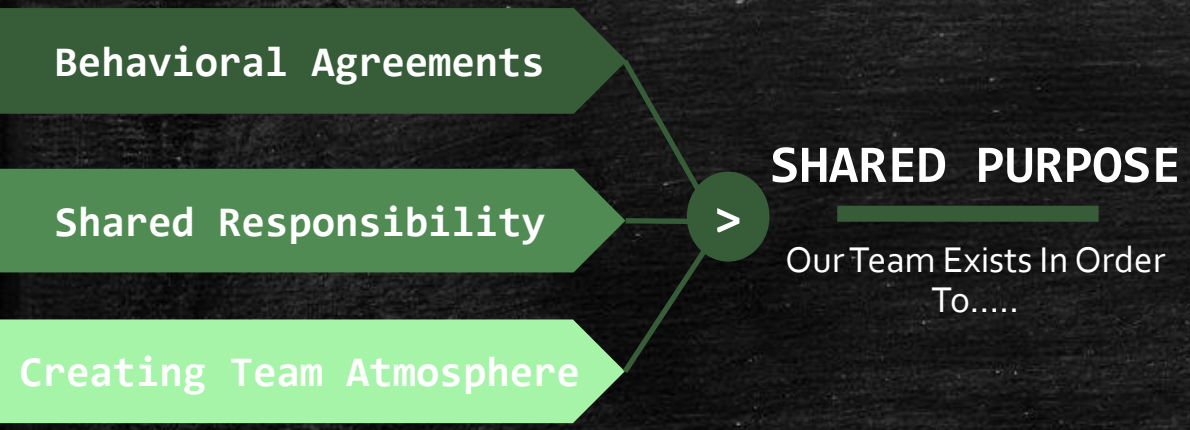
Tuckman's model of group development visualizes how taking a group of individuals to the work causes risks to project delivery due to the team's lack of effectiveness and predictability of working together as they become a team



Bruce Tuckman's studies indicate that it takes an average of 3 months for a group to become a "team" by reaching the Norming phase.

Team Formation - Charters & Working Agreements

A Team Charter captures the team's agreements on how they will work together. This traditionally includes Team Norms, Behaviors, Definitions of Ready, Definitions of Done, Role Accountabilities. Depending on the team, it can be helpful to also include items such as Board Column Exit Criteria, Team Exceptions to Project/Product Management Governances, Deployment Processes, Escalation Process, Testing Coordination and Communication Plans. A team should review its working agreement regularly such as when a new member is added to the team or quarterly at a retrospective.



Questions to propose to the team to get them started

What team policies or standards are useful to add to our charter?

- What would help improve communication for distributed people?
- How can we make information and decisions radically transparent?
- How can we increase knowledge and cross-training/

What behaviors do we value as a team?

- What are 3-5 phrases that team feel describe them?
- Examples may include: Tell it like it is, Cross-training, Respect each other, raise issues and concerns sincerely, have fun, start focus finish,

How do we want to be if someone violates our Team Charter?

- What emotions do we want to act from, vs what our immediate internal thought might be?
- Will we try to come from a place of curiosity or a place of frustration, disbelief, or judgement?
- As a team, how will we react?

What do we want to do if someone ignores our Team Charter?

- Based on where we want to be, what will we do?
- Examples:
  - Anyone may ask, is our Team Charter still serving us?
  - Anyone may ask, where are we in *The Responsibility Process?
  - Use *COIN

Team Formation – Team's Definitions

Features – Source of truth that encompasses feature priority within a project, progress and links or attachments of related documentation (IE: business requirements, technical designs, UAT testing validation, etc.)

Definition of Ready

Description of the business ask and acceptance criteria is documented
Architecture and/or technical design linked
Prioritized by sequence & business value
Points of contact are identified (UAT testers, individuals to keep informed, etc.)
High level dependencies identified (IE: Automated testing, UX/UI, Infrastructure Ops, etc.)
Applicable target deployment dates **documented**

Definition of Done

Successful deployment(s) into production
Adherence to Change Management Controls is documented
Stakeholder communication has been issued
All stories are closed with PO approval
Demonstration to Stakeholders has been completed
Fully tested including UAT completed & documented
Documentation has been updated as required

Board Column	Requirements Gathering	Refinement	Execution	Ready to Deploy	Deployed
Exit Criteria	•Feature meets team's Definition of Ready •Next in priority order to be refined	•Known stories meet team's definition of ready •Testing plan is linked if required(UAT, regression, etc.) •Have refined Acceptance Criteria •Stories are estimated & mapped to a sprint •Dependency alignment is confirmed •UI design (ideation, prototype, finalized) is documented •Automated testing needs are defined •Target deployment date documented if applicable	•Stories are in a closed status with PO approval •Bugs have been resolved & closed •Test cases/plans are linked to the stories •Business approval to deploy to production is documented or linked •Demo has been given for Stakeholder signoff	•Applicable documentation is updated or noted if none •Outstanding Defects have documented business acceptance of being in production with a story in the backlog to be prioritized to resolve it •Documentation of following the Change Management process is noted or linked (ie: CAB approval, ServiceNow RFC ticket number)	•Feature meets team's Definition of Done •Code has successfully deployed •Associated SNOW tickets have been resolved •Knowledge Transfer completed with Support Teams with presentation linked •Release notes and Stakeholder communication has been sent

Team Formation – Team's Definitions

Stories – source of truth that encompasses story requirements, priority, progress, test cases, early risk indicators and sprint commitment with velocity (team and individual representation to align work items)

Definition of Ready

Prioritized by sequence and business value
Acceptance Criteria is clear with few to no unknowns
Dependencies are aligned or resolved
Technical solution has been defined and clarified
Exceptions to process norms have been identified/documented (IE: Spike stories, Bugs, etc.)
Spikes – to be pointed but no test requirements

Definition of Done

Validation/Demo for PO approval/sign off
Development has been completed
Test cases are completed and linked
Pull request process has been completed
Applicable documentation is updated (IE: I&O changes, Database Schema, Call changes, User Guide)
Exceptions to process norms are documented
Bugs are categorized correctly
Adherence to the Change Management Controls documented

	New	Ready for Dev	Blocked	In Dev	Ready for PR	Ready for QA	In QA	Ready for Acceptance
Exit Criteria	• Prioritized Sprint backlog • Next prioritized work item • Meets Team's Definition of Ready	• Technical solution has been defined and documented/linked • External and/or internal dependencies have been aligned ➢ Internal Example – Story sequence ➢ External Example – UX/UI or automated testing needs	• Blocker/Impediment has been removed or resolved and story has been documented	• Development has been completed per story Acceptance Criteria and technical design	• Peer review has been completed on development work	• Next prioritized work item that is ready to be tested	• Test cases are linked to the story with documented results • Bugs related to the story have been identified and resolved or added to the backlog for prioritization	• Story meets Story Definition of Done • Validation testing/demo has been completed • Related bugs have been resolved and documented with aligned reason selections

Frameworks

Most common frameworks include….

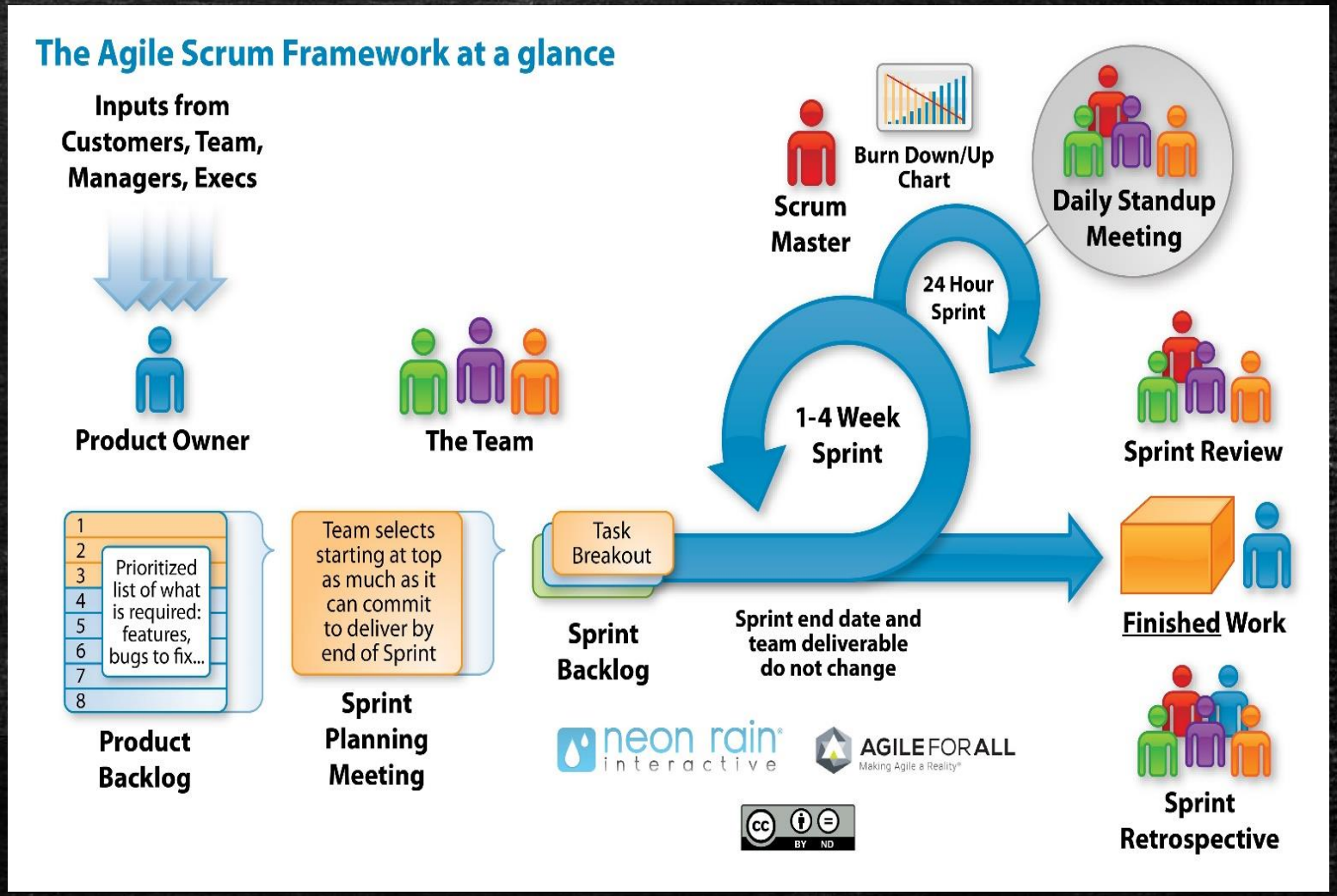
- Scrum
- ScrumXP
- Kanban
- Scrumban

	Scrum/Scrum XP	Kanban	Scrumban
Roles	• Product Owner • Scrum Master • Team	• Team + Needed Roles	• Team + Needed Roles
Ceremonies	• Daily Stand-Up • Sprint Planning • Sprint Review • Sprint Retrospective	• Daily Stand-Up • Review • Retrospective	• Daily Stand-Up • Planning • Review • Retrospective
Iterations	Sprints (1-4 weeks)	Optional	Continuous flow
Work In Progress	Controlled by Sprint content	Controlled by Workflow state (team sets limits)	Controlled by Workflow state (team sets limits)
New Work	Should wait for next sprint	Added based on priority	Added based on priority
Backlog	Target is 3 sprints worth of work being refined	Target is one-month worth of work being refined	Target is one-month worth of work being refined

*Teams working in an environment that requires release planning will also target to have high level estimates for features and/or stories prior to release planning/commitment

Frameworks - Scrum

A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value. *Scrum* itself is a simple framework for effective team collaboration on complex products. (https://www.scrum.org/resources/what-is-scrum)



Frameworks – Scrum Ceremonies

Backlog Refinement

Purpose – collaborate on Product Backlog items (Design/Solution/Acceptance Criteria)
Attendees – PO, SM, Team

Recommended Cadence
1-2 hours per week

Agenda Items
- Refine Epic/Feature/Story acceptance criteria
- Slice stories using *INVEST
- Point stories

Best Practices – maintain up to 3 sprints of stories that are refined and pointed

Sprint Planning

Purpose – collaborate on the sprint commitment based on PO Sprint Goal and prioritized product backlog

Attendees – PO, SM &Team

Recommended Cadence – First day of the Sprint

Agenda Items
- Confirm team capacity & velocity
- PO reviews Sprint Goal
- Review Sprint candidate stories for clear acceptance criteria, dependencies, and point value
- Stories are assigned to team members
- SM calls for group consensus from team and PO on sprint commitment

Best Practices – start the session by accounting for all out of office time during the sprint so that the velocity target can be adjusted if needed.

Daily Stand Up/Scrum

Purpose – collaborate on a plan or play for the day on what they are going to deliver
Attendees – SM, Team and PO as a resource
Recommended Cadence – daily for 15 minutes
Agenda Items
- What did you accomplish yesterday?
- What is your plan today?
- Do you have any impediments?

Best Practices – have a post stand up board to capture items that require further discussion.

Sprint Review

Purpose – Demo was accomplished during the sprint and gain feedback on what is being delivered

Attendees – SM, Team, PO, Stakeholders

Recommended Cadence – Last day of the Sprint

Agenda Items
- PO recaps the sprint goal or focus
- Review sprint planned stories, what the acceptance criteria/scope was and provide a demo of what they completed
- Review carry over stories (committed to at planning but not completed) and provide reason why they were carried over
- Review commitment for next sprint
- Collect Stakeholder feedback

Best Practices
Categorize Stakeholder feedback as either backlog items to be prioritized (nice to have vs MVP – minimal viable product) or requirement changes/clarification for a follow up discussion

Retrospective

Purpose – for the team to celebrate their successes in working together and to reflect and collaborate on how they can make iterative improvements

Attendees – SM and Team

Recommended Cadence – Last day of the Sprint

Agenda Items.
- What is going well
- What could be better
- Vote on items to solution
- Identify and commit as a team on one item to commit to during the sprint

Best Practices – Research, reach out to find different ways to keep the retro fun for the team.
Review the backlog of items at the next retro to assess changes and/or progress.

Frameworks – Scrum Best Practices

- Estimate the implementation effort of the product backlog together – Product Owner, Customer, Scrum Team (Scrum.org offers a tool for high level project estimation)
- Clarify dependencies between product backlog items and other teams
- Invite the customer to team ceremonies or meetings to discuss or review the solution design and acceptance criteria
- Focus on the goals of the Scrum Team
- Promote peer-to-peer collaboration
- Continually improve performance in Scrum events
- Create transparent Scrum artifacts for inspection and adaption
- Use road mapping tools to enable the organization to…
  - Plan for incremental product delivery
  - Establish the business value and priority of each increment
  - Estimate how long the product will take to develop in iterative cycles

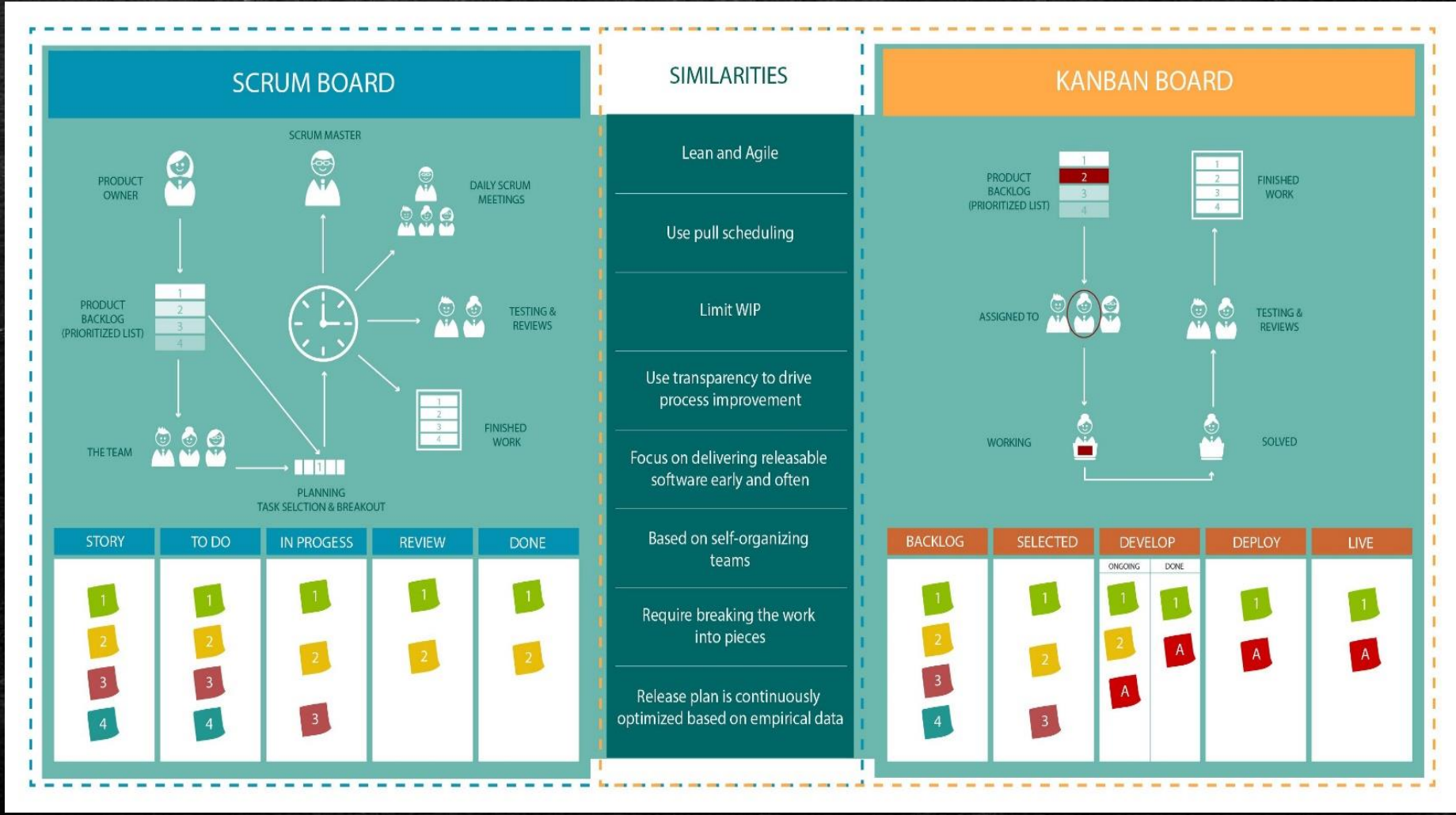
Frameworks - ScrumXP

Scrum is a ***project management*** framework and guides how to manage a project, while **XP** contains specific ***engineering practices*** that enable the delivery of potentially shippable software. Marrying the two yields the best results for teams that have a solid Scrum foundation.

Extreme Programming Practices in Scrum	
Group	**Practice**
Feedback	✓ Test-Driven Development • Writing requirements like test cases ✓ Pair Programming • Two programmers work jointly on the same code with one focused on writing and the other reviews the code, suggesting improvements and fixes mistakes along the way
Continual Process	✓ Continuous Integration • Commit code multiple times a day ✓ Code Refactoring • Continuously improving code by removing redundancy, eliminating unnecessary functions, increasing code coherency and at the same time decoupling elements ✓ Small Releases • Release the first version quickly and further developing the product by making small and incremental updates
Code Understanding	✓ Simple Design • The right design should pass all tests, have no duplicate code and contain the fewest possible methods and classes ✓ Collective Code Ownership • The whole team is responsible for the design of a system. Each team member can review and update code ✓ Coding Standards • Have a common sets of coding practices, using the same formats and styles for code writing
Work Conditions	✓ 40-Hour Week and Consistently work at 85% output • Teams & Individuals have modes of Crunch Mode (overloaded with work), Recovery Mode (not productive) and Optimal→ Increasing the frequency of Crunch Mode, the longer the Recovery Mode becomes

Frameworks - Kanban

- A visual system for managing work as it moves through a process. Kanban visualizes both the process (the workflow) and the actual work passing through that process
- Goal of Kanban is to identify potential bottlenecks in the process and fix them so work can flow through it cost-effectively at an optimal speed or throughput



Frameworks – Kanban Ceremonies

Backlog Refinement

Purpose – collaborate on Product Backlog items (Design/Solution/Acceptance Criteria)
Attendees – PO, SM, Team

Recommended Cadence 1-2 hours per week

Agenda Items
- Refine Epic/Feature/Story acceptance criteria
- Slice stories using *INVEST
- Estimate Stories

Daily Stand Up

Purpose – collaborate on a plan or play for the day on what they are going to deliver
Attendees – SM, Team and PO as a resource
Recommended Cadence – daily for 15 minutes
Agenda Items
- What did you accomplish yesterday?
- What is your plan today?
- Do you have any impediments?

Best Practices – have a post stand up board to capture items that require further discussion.

Review

Purpose – Demo was accomplished during the sprint and gain feedback on what is being delivered

Attendees – SM, Team, PO, Stakeholders

Recommended Cadence – Ad-hoc as needed

Agenda Items
- Review story acceptance criteria/scope and provide a demo of what was completed
- Collect Stakeholder feedback

Best Practices
Categorize Stakeholder feedback as either backlog items to be prioritized (nice to have vs MVP – minimal viable product) or requirement changes/clarification for a follow up discussion

Retrospective

Purpose – for the team to celebrate their successes in working together and to reflect and collaborate on how they can make iterative improvements

Attendees – SM and Team

Recommended Cadence – Ad-hoc as needed or determined by the team

Agenda Items.
- What is going well
- What could be better
- Vote on items to solution
- Identify and commit as a team on one item to focus on as an iterative improvement

Best Practices – Research, reach out to find different ways to keep the retro fun for the team.
Review the backlog of items at the next retro to assess changes and/or progress.

Frameworks – Kanban Best Practices

Focus on quality

a. 90% of a team's effort is spent fixing defects
b. Encourage high initial quality - big impact to productivity and throughput of a team

Reduce Work-in-Progress (WIP)

a. No more than 5-10 items (number dependent upon team size) in-progress at any given time
    I. Increases speed to market and customer feedback loop
    II. Increases quality; the Scrum Master should closely manage WIP as a means to control quality

Deliver Often

a. Reducing WIP limits shortens lead time; shorter lead time enables releasing working code more often
b. More frequent releases build trust with your customers

Balance Demand against Throughput

a. Set the rate at which the team will accept new requirements into the software development pipeline to correspond with the rate at which we can deliver working code
    I. This ensures that the team will effectively fix WIP to a given size

Prioritize

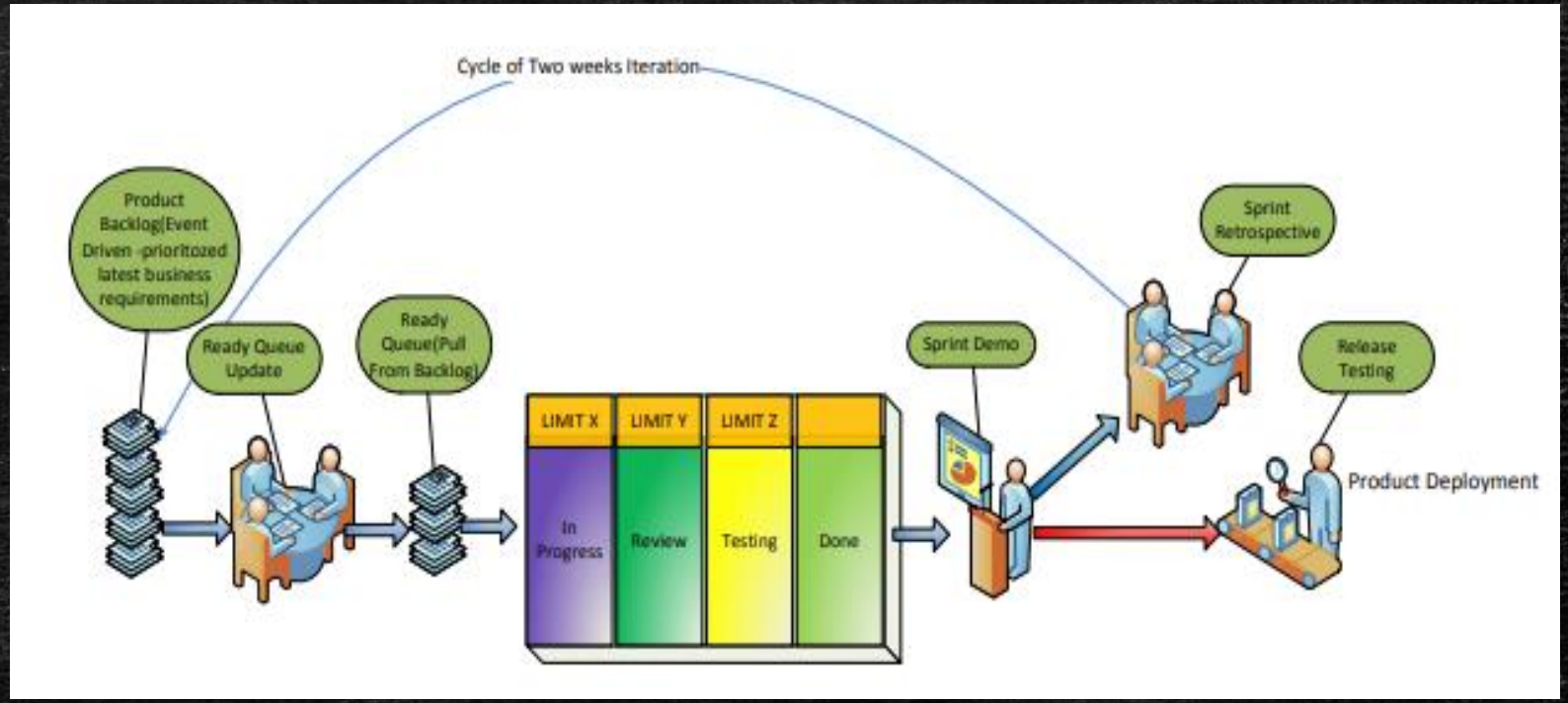
a. Order the priority of the input in approximately the order requested until consistency is achieved in both delivery and predictability
b. New work should be pulled into development only as capacity is freed up on completion of existing work

Attack Sources of Variability to Improve Predictability

a. Variability impacts flow of work through a value stream
    I. Utilize root cause analysis techniques to determine cause of variability
    II. Implement solutions to reduce variance, increase consistency and predictability

Frameworks - Scrumban

A hybrid agile methodology that uses the structure of Scrum (backlog and ceremonies) with the flexibility of Kanban’s flow-based method of WIP limits.



Frameworks – Scrumban Ceremonies

Backlog Refinement

Purpose – collaborate on Product Backlog items (Design/Solution/Acceptance Criteria)
Attendees – PO, SM, Team

Recommended Cadence
1-2 hours per week

Agenda Items
- Refine Epic/Feature/Story acceptance criteria
- Slice stories using *INVEST
- Estimate Stories

Planning

Purpose – Identify what in flight work can be completed within a specified timeframe

Attendees – PO, SM &Team

Recommended Cadence – Determined by the team & work type

Agenda Items
- Confirm team capacity/velocity
- Review candidate stories for clear acceptance criteria, dependencies, and estimate
- Stories are assigned to team members
- SM calls for group consensus from team and PO on commitment

Best Practices – start the session by accounting for all out of office time during the time frame

Daily Stand Up

Purpose – collaborate on a plan or play for the day on what they are going to deliver
Attendees – SM, Team and PO as a resource
Recommended Cadence – daily for 15 minutes
Agenda Items
- What did you accomplish yesterday?
- What is your plan today?
- Do you have any impediments?

Best Practices – have a post stand up board to capture items that require further discussion.

Review

Purpose – Demo was accomplished during the sprint and gain feedback on what is being delivered

Attendees – SM, Team, PO, Stakeholders

Recommended Cadence – Ad-hoc as needed

Agenda Items
- Review story acceptance criteria/scope and provide a demo of what was completed
- Collect Stakeholder feedback

Best Practices
Categorize Stakeholder feedback as either backlog items to be prioritized (nice to have vs MVP – minimal viable product) or requirement changes/clarification for a follow up discussion

Retrospective

Purpose – for the team to celebrate their successes in working together and to reflect and collaborate on how they can make iterative improvements

Attendees – SM and Team

Recommended Cadence – Ad-hoc as needed or determined by the team

Agenda Items.
- What is going well
- What could be better
- Vote on items to solution
- Identify and commit as a team on one item to focus on as an iterative improvement

Best Practices – Research, reach out to find different ways to keep the retro fun for the team.
Review the backlog of items at the next retro to assess changes and/or progress

Frameworks – Scrumban Best Practices

As in Scrum

1. Ensure your product backlog item meets the team's definition of *ready* prior to starting development.
2. Apply WIP limit even to the backlog to reduce possible waste of estimating how much can be done in one iteration.

As in Kanban

1. Use ‘ready’ queue (between backlog and doing) to organize
2. Start with what you do now
3. Agree to pursue incremental, evolutionary change
4. Visualize the workflow
5. Limit Work-in-Process (WIP)
6. Balance the flow
7. Make management and process policies explicit
8. Deliver more often
9. Pull new work into development only when capacity is freed up on completion of existing

Frameworks – Other Practices

This may not be applicable, depending on what the organization supports

- Agile @ Scale
- SAFe
- Crystal
- Lean

Progressive Elaboration

With each project comes the cone of uncertainty as requirements are clarified and solutions are designed. From project ideation to stable teams taking in the work for execution is a practice of continuous collaboration and refinement of the solution(s) and feature deliveries. Below are some phases to enable collaboration to work through uncertainties, ambiguities and risks. *Names of meetings and roles involved can vary depending on the organization*.

Elaboration Begins → **Project Execution**

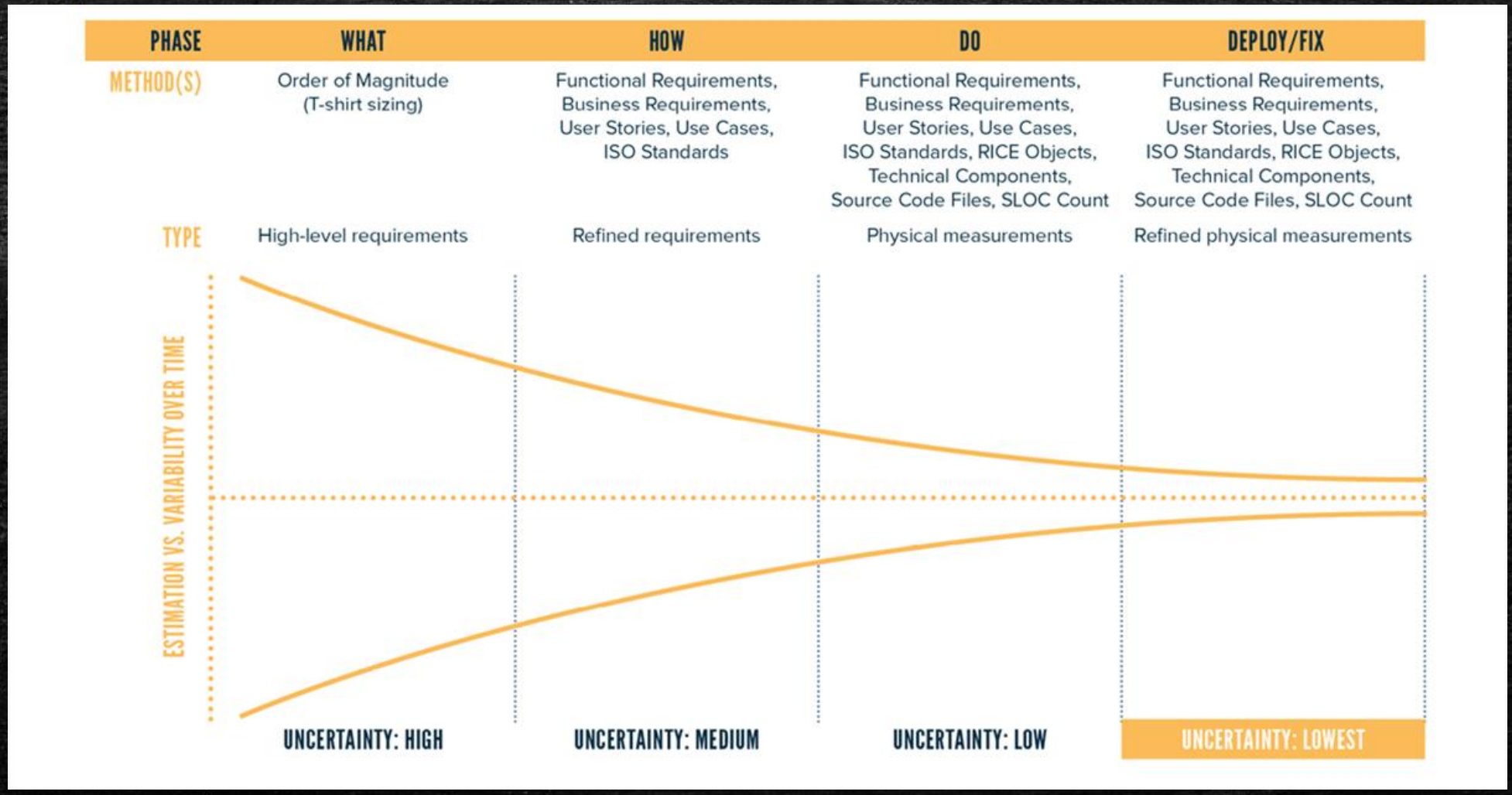
Customer, Program Team(s) and Stable Team(s) members Product Owner, Scrum Master, Business/Systems Analyst, Tech Lead meet to review project details and ask any clarifying questions
All teams being impacted attend

Teams that would have work for the project would draft backlog items (Epics, Features, High Level Stories) and obtain high level estimate from the Stable Team

Program Team(s) and Stable Team(s) members (Product Owner, Scrum Master, Business/Systems Analyst, Tech Lead) meet again with high level work identified to create a process map, align dependency work items, create road map of project deliveries across teams and identify any risks

Stable Teams refine the backlog items, provide estimates and map the stories to sprints based on priority and velocity. Anything discovered that would impact dependency alignment or the project road map is communicated to the Program Team(s)

Progressive Elaboration – Cone of Uncertainty



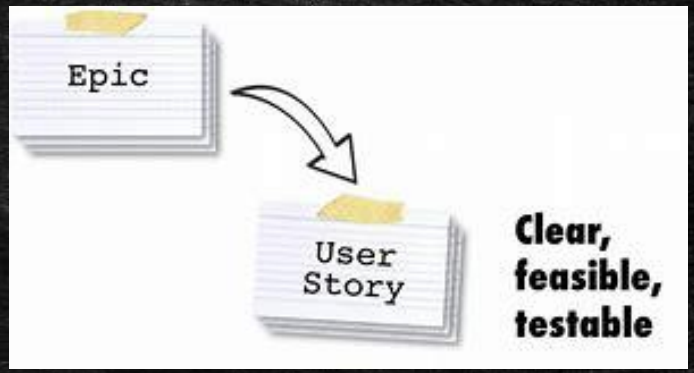
Progressive Elaboration – Defining Done

Epic

- Comprised of Features and/or Stories
- Acceptance Criteria or Definition of Done (DoD) is defined by the Product Owner and Customer
- Value Add, ROI and Cost of Delay assists with prioritizing the backlog.

Best Practice – Create a template to gather & document requirements

- Customer ask (what are they trying to solve)
- Current work around if applicable
- Value/ROI/cost of delay
- Target completion date if applicable
- Requirements



Feature

- Chunk of functionality that delivers business value
- Additions or changes to existing functionality
- Acceptance Criteria of a feature is written from the user perspective
- Acceptance Criteria compasses the User Stories for an over arching definition of done

Story

- Short, simple descriptions of a feature told from user perspective
- Stories contain multiple types of work (e.g., programming, testing, database design, user interface design, analysis, etc.)
- Written using **INVEST** (**I**ndependent, **N**egotiable, **V**aluable, **E**stimable, **S**mall, **T**estable) model

Progressive Elaboration – Story Writing

INVEST (**I**ndependent, **N**egotiable, **V**aluable, **E**stimable, **S**mall, **T**estable)

Story Templates

User Story

As a < type of user >, **I want** < some goal > **so that** < some reason >.

Technical Story

In order to <some goal>, <system/application> **needs to** <some requirement>, **so that** <some reason/value>.

Story Acceptance Criteria

Given <some pre-requisite>
When <some process is executed>
Then <some outcome/output/result will occur>

EPIC

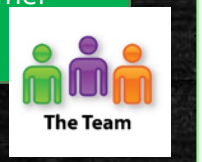
As a Customer, I want to be able to have Wishlist so that I can come back to buy products later

Feature

(usually multiple per Epic)

As a customer, I want to be able to save a product in my Wishlist so that I can view it again later.

Story writing involves all of the team members and can also include the PO and Customer



User Story (example)

Description

As a customer, **I want** to be able to save a product in my Wishlist **so that** I can view it again later

Acceptance Criteria

Given a tile has been created,
When a product is viewed,
Then a save to Wishlist option will appear
Then the option to save to Wishlist will be disabled from the screen

Technical Story (example)

Description

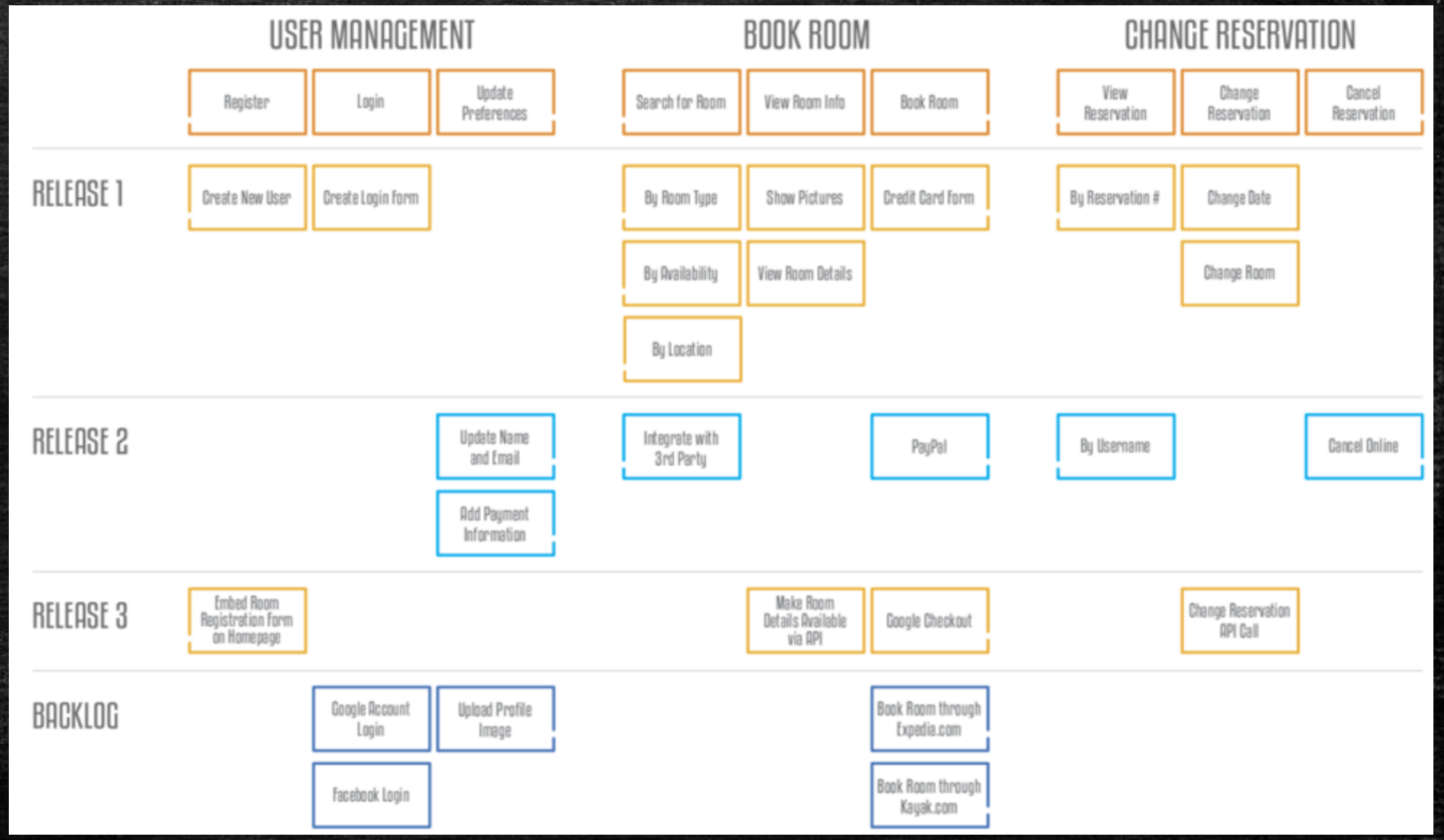
In order to have the 'add to Wishlist' option appear, another table **needs to** be added in the database **so that** the Wishlist data of the customer is stored

Acceptance Criteria

Given a table is added in the database to collect the Wishlist data,
When the add to Wishlist tile is clicked,
Then the products selected will appear in the table

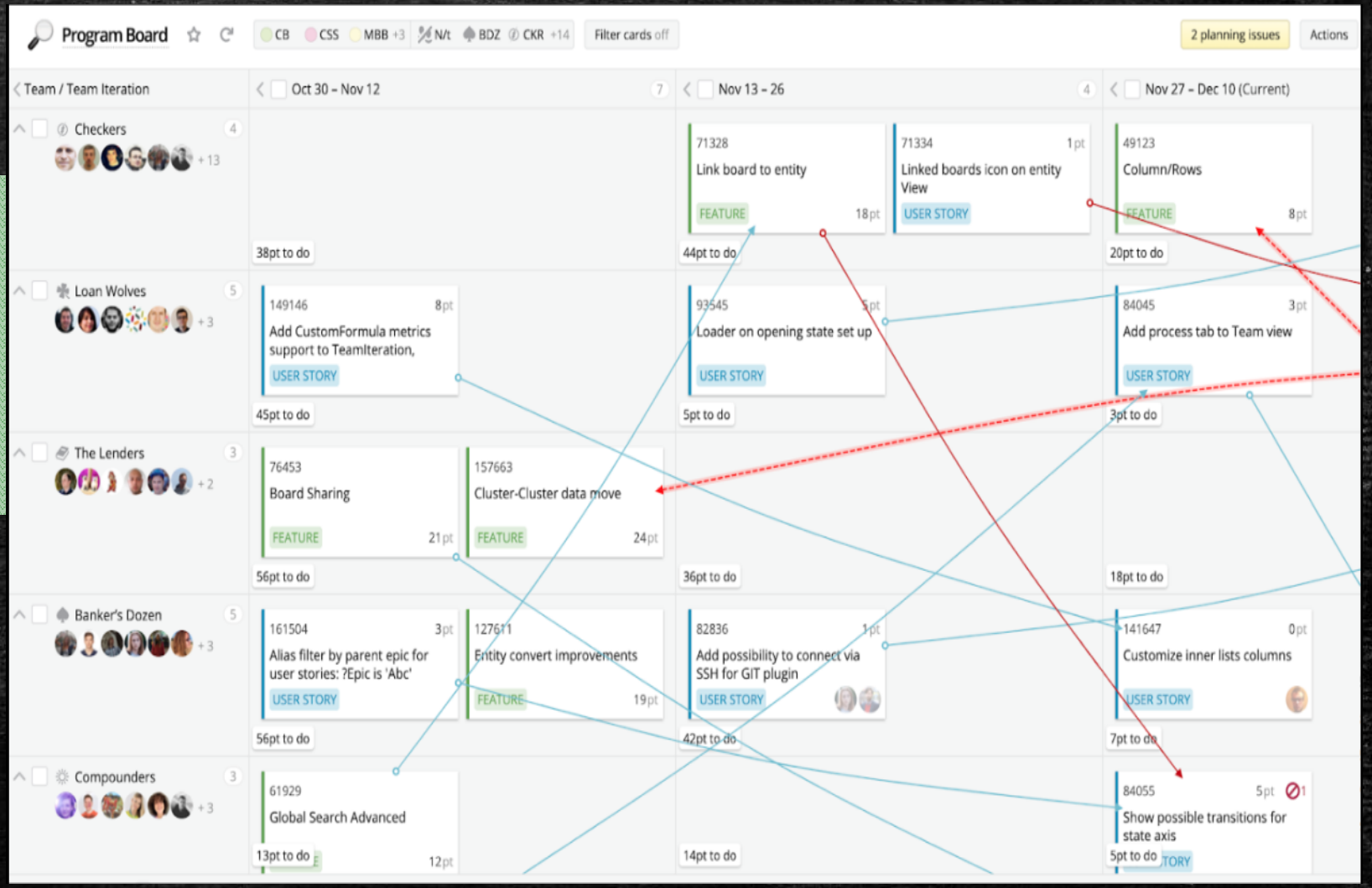
Progressive Elaboration – Story Mapping

A method for arranging user stories to create a more holistic view of how they fit into the overall user experience. Arranged on a horizontal axis, the fundamental steps of the customer journey (sometimes labeled as epics, sometimes not) are arranged in chronological order based on when a user would perform the particular task relative to their overall workflow with the product.



Progressive Elaboration – Dependency Alignment

A program board such as this can be used during a release planning session to map out and align cross team dependencies between sprints within the Release increment



Progressive Elaboration – Dependency Alignment

Once in execution, a project board, either physical or electronic can be used to maintain dependency alignment across teams for larger efforts.

If the dependency on another team is smaller scale, there are options in most tools to tag the stories to create a visual of the stories that have dependencies on another team's work or is a dependency for another team's work. To maintain alignment, communication expectations between teams would need to be established.

Teams with many cross-team dependencies that may span multiple projects or efforts, the Scrum Masters within the teams often have regular touch bases to discuss alignment and risks

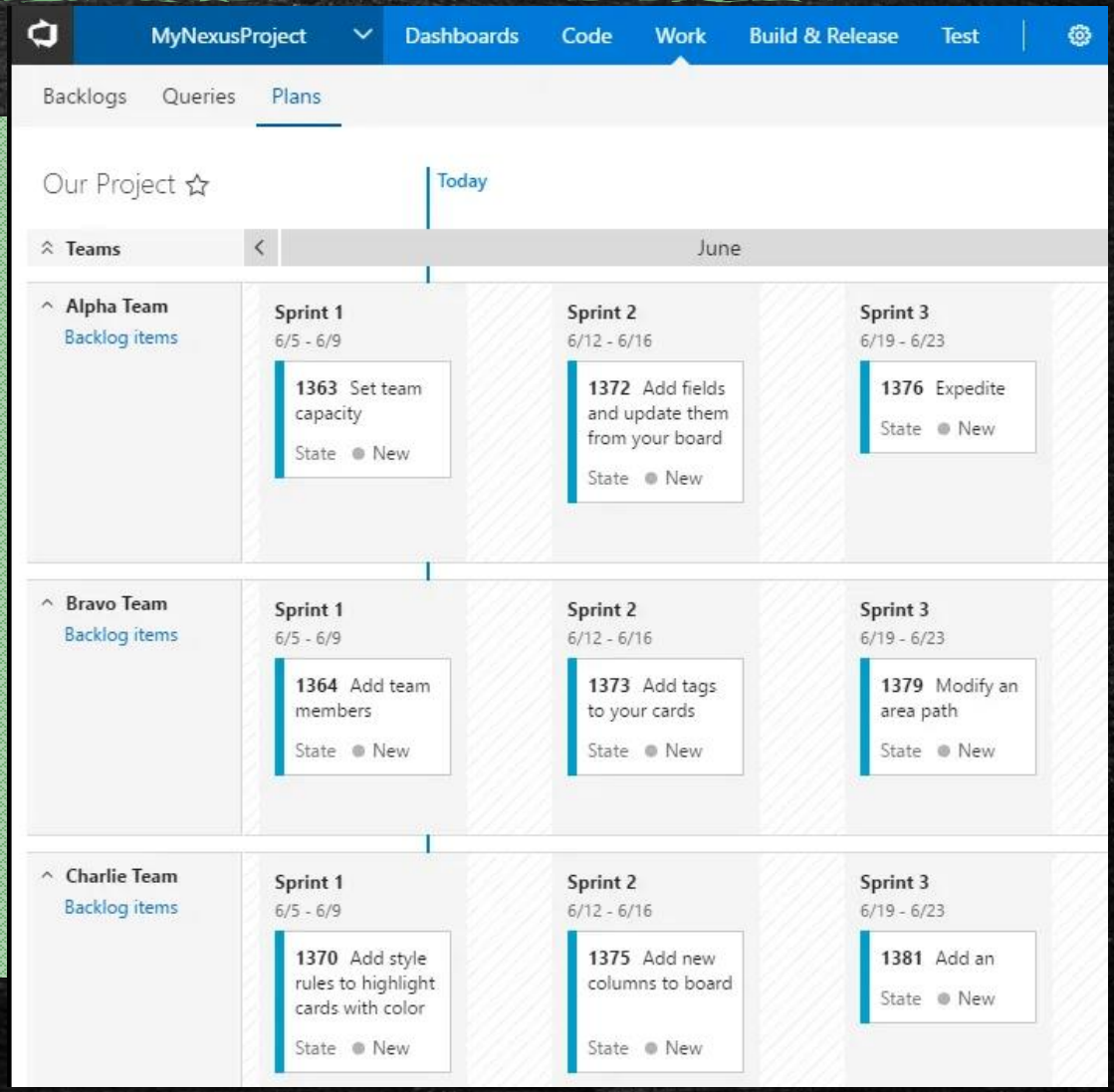
Example:

Alpha Team has a dependency on the UX/UI team to create a design mockup

Alpha Team adds a tag to the story that has the dependency on the UX/UI work so that it is visible which stories they need to align with that team.

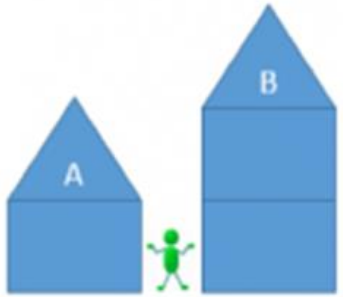
Alpha Team engages the UX/UI team to map out the stories to sprints and align on a tentative target sprint that their work would be completed by.

They discuss their preferred form of communication (attending stand up, frequent email updates, separate sync up, etc.) and they choose to have someone from the UX/UI team attend daily stand up once their work is in execution.



Estimating - Fibonacci

Why Use Fibonacci Numbers?



1 story vs 2 story

Could you tell which house is taller if you were standing on a street in front of them?

100 story vs 101 story

What about these?
Which building is taller?
If you were on the street?

60 story vs 100 story

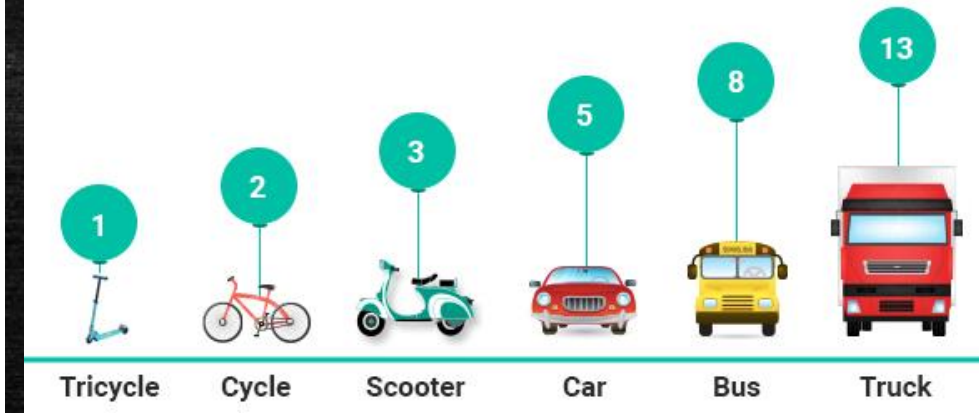
Fibonacci numbers get farther apart as the numbers get bigger making estimation easier
1, 2, 3, 5, 8, 13, 20, 40, 60, 100...

...and you can add them up! Unlike t-shirt sizes. What's a small plus an extra-large equal?

Agile Estimating Best Practices

Size is *estimated*, **velocity** is *measured*, **duration** is *derived*, **cost** is *calculated*

- **Estimate Size**
  - Fibonacci Number Sequence
    - Anchor Story
    - Planning Poker
    - Wall Technique
- **Measure Velocity**
  - 4-6 Sprints to determine Velocity
- **Inspect and Adapt/Recalibrate**



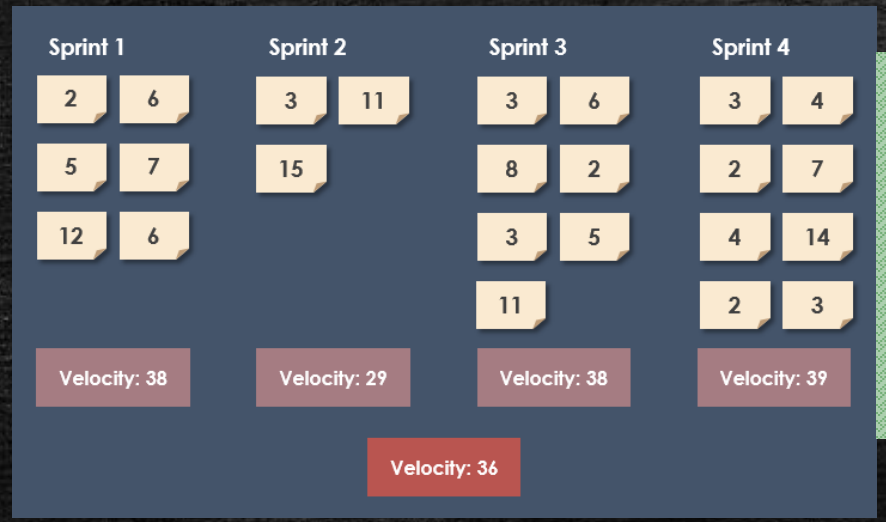
Estimating - Fibonacci

Points	Things to consider to get started pointing
1	Quick to deliver & minimal complexity – Know exactly what needs to be done (*IE: Add field to a form*)
2	Quick to deliver, some complexity – I mostly know what needs to be done (*IE: Add a parameter to a form, validation, storage*)
3	Moderate complexity, some uncertainty/unknowns – Good idea of what needs to be done (*IE: Migrate somewhat complex static CSS into CSS pre-processor)*
5	Longer time to deliver/high complexity, likely unknowns – High level understanding of what needs to be done (*IE: Integrate with third-party API for pushing/pulling data and link to user profiles in platform*)
8	Long time to deliver, high complexity, critical unknowns – Understand the concept and goals of the work, may need to be broken down into a spike to remove some uncertainty (*IE: Overhaul the layout/HTML/CSS/JS of a web application*)
13	Long time to deliver, high complexity, many critical unknowns – (*IE: Migrate application from outdated data store to new DB technology and ORM*)

User Story Estimates		Build Complexity		
		Low	Med	High
Time Intensive	Low	1	3	5
	Med	3	5	8
	High	5	8	13

Pointing Matrix – A team that is **NEW** to pointing can use this to assist in transitioning from Time based estimates *to Complexity based estimates*

Estimating – Velocity & Capacity



Velocity is based on estimating stories using **pointing** . Average velocity range is calculated based on the points a team has completed over a span of 3 to 4 sprints or iterations. Using a team's average velocity range, estimated stories can be mapped to a sprint up to a team's target velocity

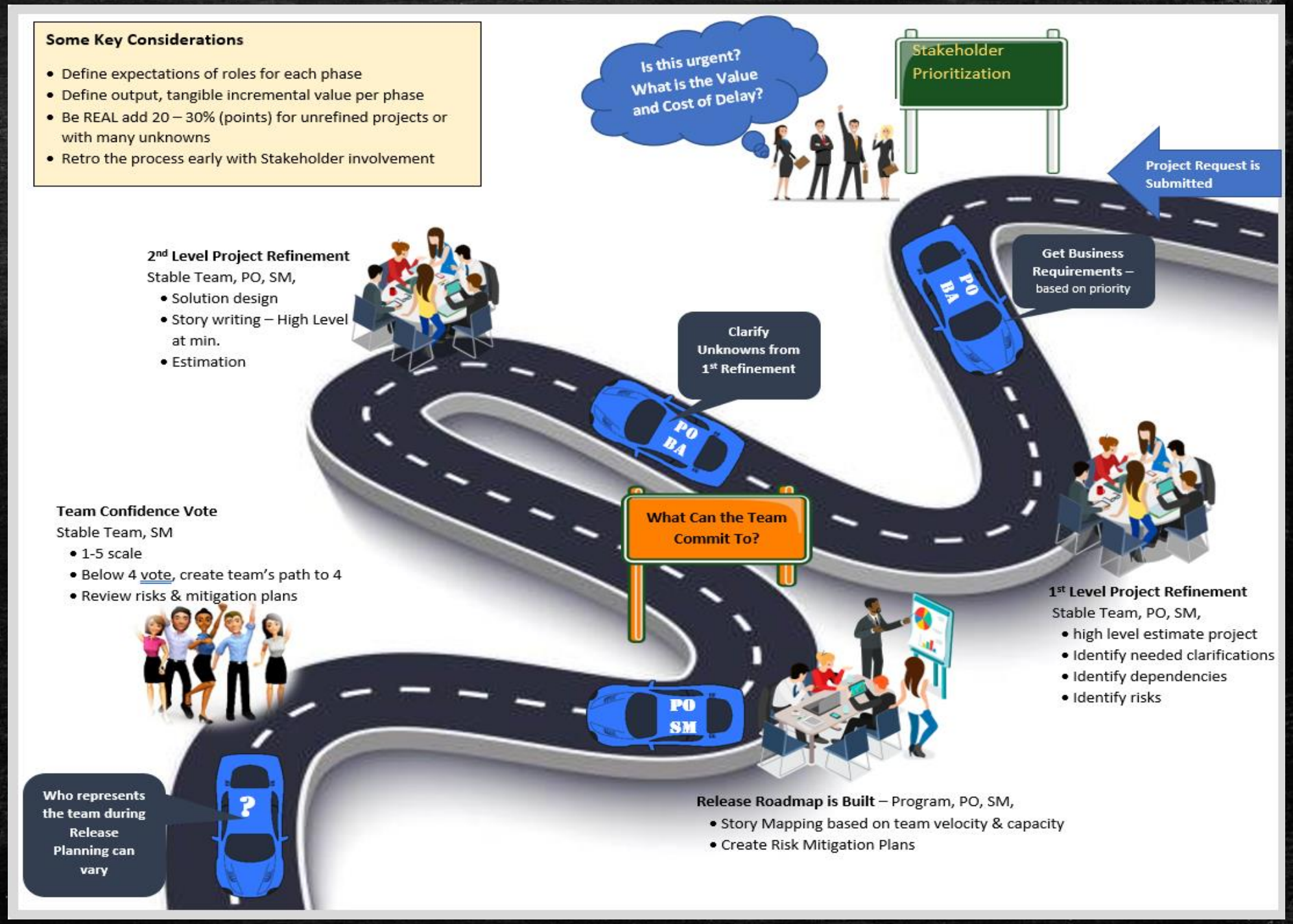
Using Capacity planning using day increments vs hours during sprint and release planning provides indicators to sprints where target velocity would need to be adjusted.

Example: the image below indicates 10 days in an iteration for each person (45 days) and Iteration 32 shows the team has 7 out of office days total. An Average Velocity of 90 points for this team would equate to approximately 2 points/day. For Iteration 32, a target velocity should be reduced by 2 points for each out of office day, making the adjusted velocity 76 points

Capacity is based on estimating stories or tasks using the number of **hours** it would take to complete. Using a team's estimated capacity per iteration/month or Release, work can be mapped to align with dependencies and create a Release Plan

Team Details				Iteration 31		Iteration 32		Iteration 33		Iteration 34		Iteration 35		Release 4	
Team Members	Allocation	Hrs/ Day	Hrs/ Sprint	OOO Hours	Avail Hours	OOO Hours	Avail Hours	OOO Hours	Avail Hours	OOO Hours	Avail Hours	OOO Hours	Avail Hours	OOO Hours	Avail Hours
Developer 1	100%	6	60	12	48	18	30	0	54	0	60	0	54	30	246
Developer 2	100%	6	60	0	60	6	42	0	54	6	54	0	54	12	264
Developer 3	100%	6	60	0	60	0	48	6	48	3	57	0	54	9	267
Tester 1	100%	6	60	3	57	12	36	0	54	0	60	0	54	15	261
Tester 2	50%	3	30	0	30	6	18	0	27	0	30	3	24	9	129
Whole Team OOO Hours →				0		54 (12 x 4 + 6)		27 (6 x 4 + 3)		0		27 (6 x 4 + 3)		108	
Total		27	270	15	255	96	174	33	237	9	261	30	240	183	1167

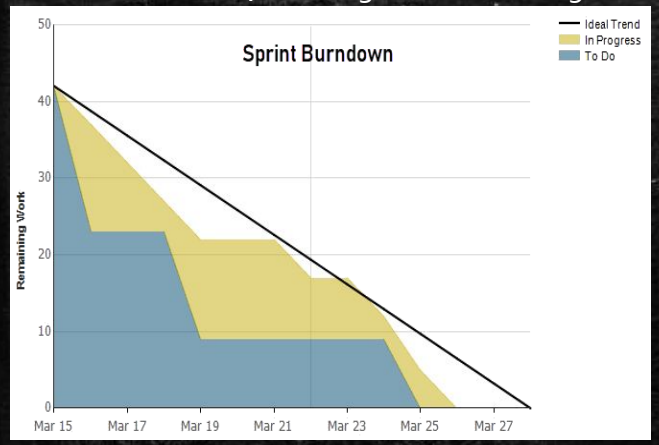
Roadmaps – Preparing a Release Plan

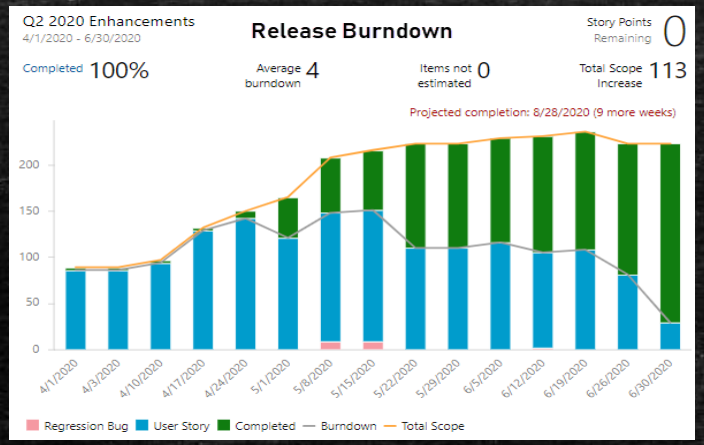


Metrics – Team Focus

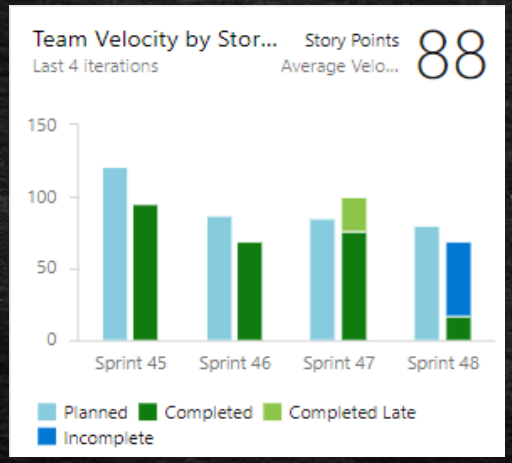
The most important thing stakeholders need to know about your project is whether it is on track. The following metrics help communicate this, and explain deviations from the expected project path

Sprint and release burndown—Gives stakeholders a view of your progress. Starting with the initial scope of the focus, it visualizes the progress of either completion of the work, increase in the scope of the work or a "flatline" or no progress in the work. Visualizing these things provide an early indicator of risks, allowing for earlier mitigation and awareness.

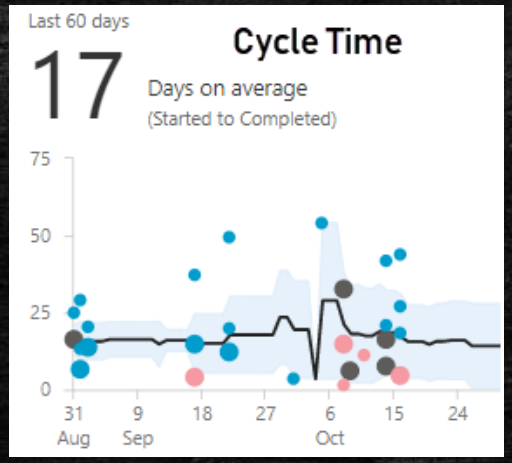




Team capacity—How many developers are on the team full time? Has work capacity been affected by vacations or sick leave? Developers pulled off to side projects? Does the target velocity of the Sprint or Release need to be adjusted?



Sprint velocity—A historic review of how much value you have been delivering. This metric helps determine the predictability of the team's delivery of planned work. Predictability enables a more accurate road mapping and planning



Cycle Time – Used to assess the current process for any "bottle necks" of where the progress of delivering the work is slowing down or could be more efficient. If it jumps up or drops in an area, it would prompt conversations to understand the cause and effect.

Metrics – Enterprise Focus

This slide will vary, depending on the organization. It could be a combination of Project, Financial and Team Health metrics with risks and risk mitigation plans identified by project

Allocated Budget Across Portfolios	Used Budget Across Portfolios	Used Budget Percentage	Estimated Cost Across Portfolios	Actual Cost Across Portfolios
26.27 M	10.36 M	39 %	11.17 M	7.99 M

Portfolio Summary

Portfolio Name	Portfolio owner	Health	Budget	Used Budget	# Projects
Facilities Upgrade	Thomas Béddov		1.50 M	975.00 K	2
Human Capital Initiatives	Thomas Béddov		1.52 M	984.00 K	3
Insurance Biz	Thomas Béddov		4.85 M	2.10 M	4
Internal Activities Portfolio	Thomas Béddov		2.75 M	1.05 M	4
	Béddov		3.54 M	565.50 K	3
	Béddov		4.50 M	1.06 M	5
	Béddov		5.10 M	1.88 M	14
	Béddov		2.50 M	1.75 M	1

PPM PORTFOLIO SUMMARY

PORTFOLIO: All | PROJECT TYPE: All | PROJECT OWNER: All | PROJECT MANAGER: All

OVERALL STATUS

OFF TRACK	OVERDUE ISSUES	BEHIND SCHEDULE	OVER BUDGET	ACTIVE RISKS
4	6	22	17	8

Portfolio	Project Name	Workspace URL	Overall	Schedule	Work	Resource	Cost	Issue	State	Due Date	% Complete	Duration Variance	Cost Variance
Application	Media Research Website								3) In Progress	6/13/2018	27%	480	$294,000
	UltraViolet								3) In Progress	7/24/2018	15%	0	$0
Create Competitive Advantages	Collections Managment System								3) In Progress	6/27/2019	30%	0	($1,684,000)
	Evaluate Strategic Merger or Acquisition								3) In Progress	9/22/2017	41%	0	$0
	Server Upgrades								1) Requested	8/26/2019	0%	0	$0
	Social Networking Integration								1) Requested	10/4/2019	0%	-1,032	$0
Cut Costs	Health Assessment Reporting Tool								3) In Progress	5/18/2018	31%	784	$73,200
	Hub Upgrade								3) In Progress	4/12/2018	5%	8	$600
	Marketing Campaign Planning								3) In Progress	8/3/2018	20%	1,416	$11,200
	Partner Help Desk								3) In Progress	7/23/2024	59%	13,875	$37,900

Resources

Websites

- www.agilealliance.org
- https://www.mountaingoatsoftware.com/
- https://agileforall.com
- https://www.scrumalliance.org
- https://www.isixsigma.com/
- https://goleansixsigma.com/
- https://www.collaborationsuperpowers.com
- Reto ideas:
  - retromat.org/
  - https://funretro.io/templates/elvis-retrospective-always-on-my-mind-little-less-conversation

Books

- Coaching Agile Teams by Lyssa Adkins
- Scrum Field Guide
- Agile Estimating and Planning by Mike Cohn
- Scrum The Art of Doing Twice the Work in Half the Time by Jeff Sutherland
- Succeeding with Agile by Mike Cohn
- Agile Product Management with Scrum by Roman Pichler
- User Stories Applied by Mike Cohn
- Extreme Programing Explained by Kent Bock
- Inventors Tool Kit by David Silverstein
- Work Together Anywhere Successfully Individuals