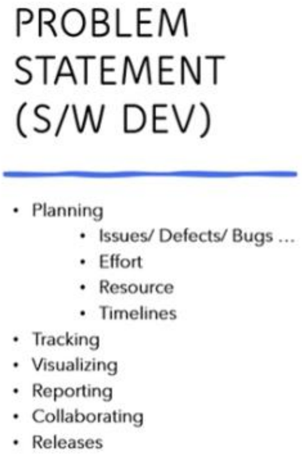
Agile Methodologies

Thursday, March 24, 2022 5:12 PM

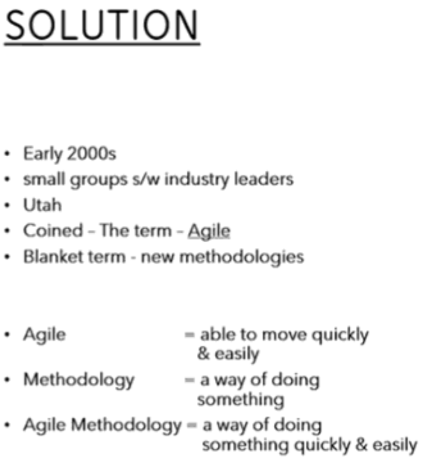
- Day to day process of software development.



PROBLEM STATEMENT (S/W DEV)

- Planning
  - Issues/ Defects/ Bugs ...
  - Effort
  - Resource
  - Timelines
- Tracking
- Visualizing
- Reporting
- Collaborating
- Releases

- These problems leads to introducing agile development.

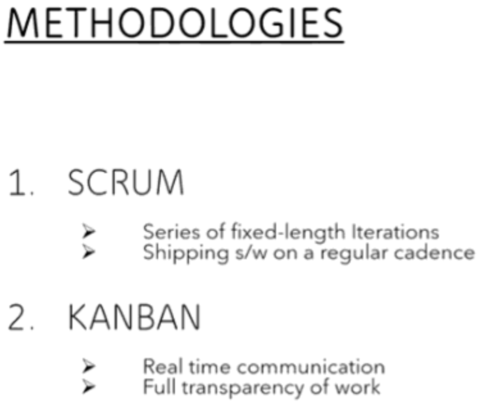


SOLUTION

- Early 2000s
- small groups s/w industry leaders
- Utah
- Coined - The term - Agile
- Blanket term - new methodologies

- Agile = able to move quickly & easily
- Methodology = a way of doing something
- Agile Methodology = a way of doing something quickly & easily

During breaks we need to review our performance and get back with the new updated plan.

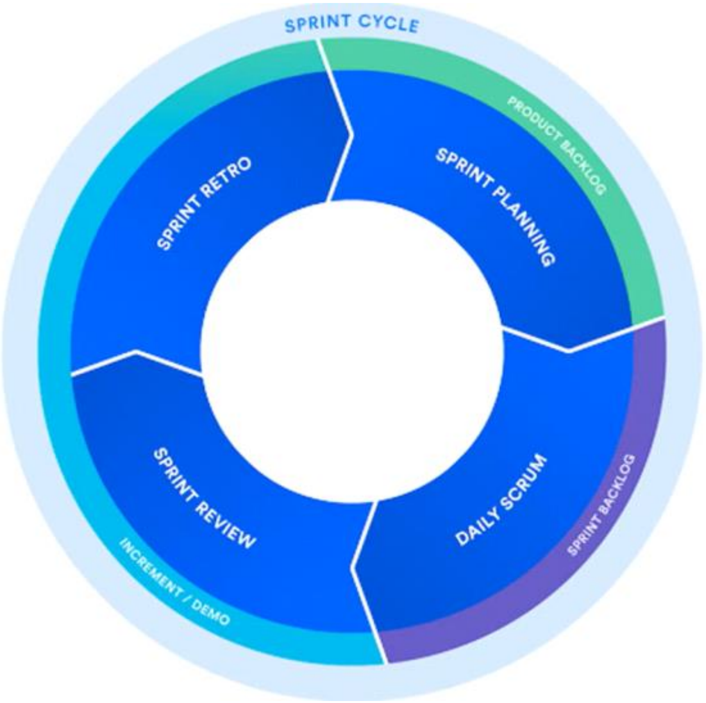


METHODOLOGIES

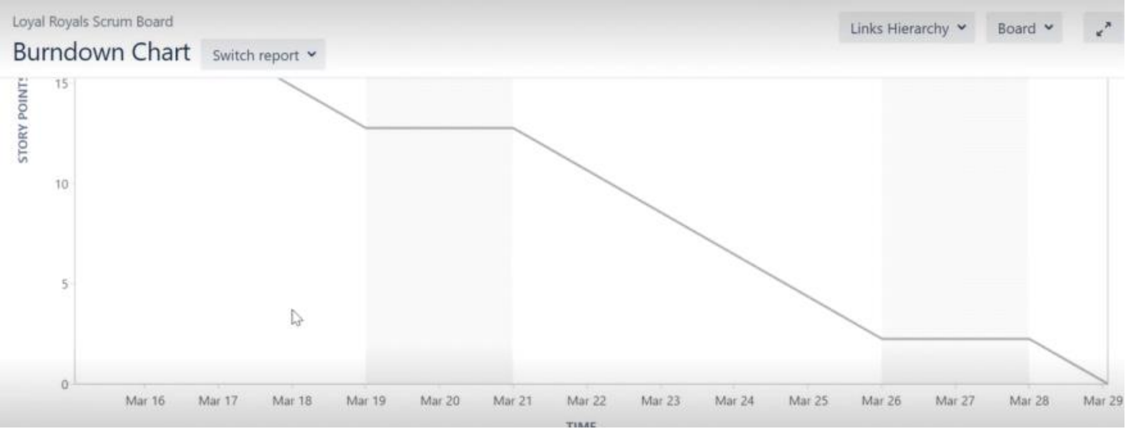
1. SCRUM
   - ➢ Series of fixed-length Iterations
   - ➢ Shipping s/w on a regular cadence
2. KANBAN
   - ➢ Real time communication
   - ➢ Full transparency of work

- Motivation among the team is necessary at regular intervals. Even we failed sometimes we need to set back with better plan.
- We must have always a backup plan.
- We need a cross functional team, if a person falls then other person should replace him immediately. That person may be some person who already existing in the company.
- SCRUM Master: must do certification on Agile Scrum
- KANBAN: no need to have any certification. He must be work practically.
- Agile is all about appreciation and motivation.
- In agile even we motivate someone for their work , but during success everyone needs to be appreciated because everyone will have their part in that success. Which develops team spirit.

SCRUM CYCLE

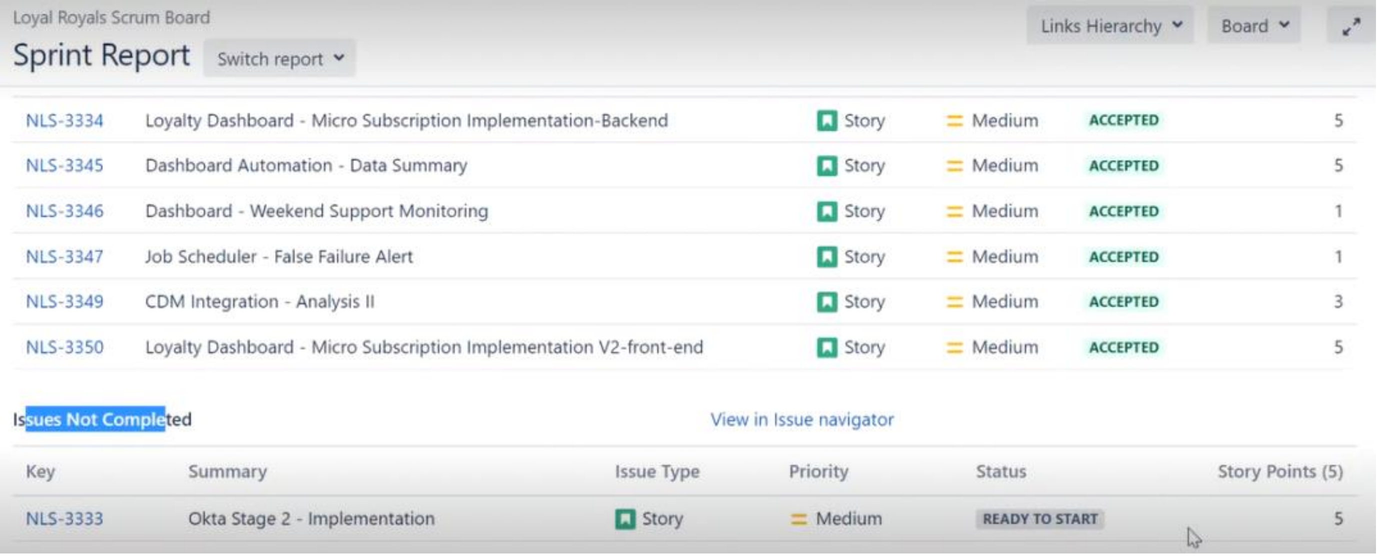


- For every three months PI Planning will process, it is nothing but performance index planning. In PI planning all the dependent teams will come together either developer, tester, business analyst .
- We call SPRINT cycle as two weeks. For every two weeks we will have sprint planning.
- Backlog; an ordered list of prioritized features that has been created keeping the product vision in mind.
- For every two weeks one backlog is added into sprint planning.
- For scrum master we need to tell, what we are doing right now, what are we planning to do and what we have done up to now.
- In daily scrum , if we have no work to do. If scrum master ask us, we need to tell openly if we are not having any tasks to do. So , he will give another important task from the product backlog.
- Every Monday we will be having a sprint review, to check on the week's progress.
- The Sprint Retrospective is an opportunity for the Scrum Team to inspect itself and create a plan for improvements to be enacted during the next Sprint. ... The Sprint Retrospective occurs after the Sprint Review and prior to the next Sprint Planning
- **KANBAN:** Continuous improvement
- **SCRUM:** Monthly or weekly reports
- Active sprint mean the task that must be do now as part of sprint plan.
- **Burndown chart** let's scrum master to learn how many tickets or points that developer is closing. Here Burndown chart graph done for each day.



- The contents that we added as we done is called as story

SPRINT REPORT: done according to the sprint timeline



Sprint Report

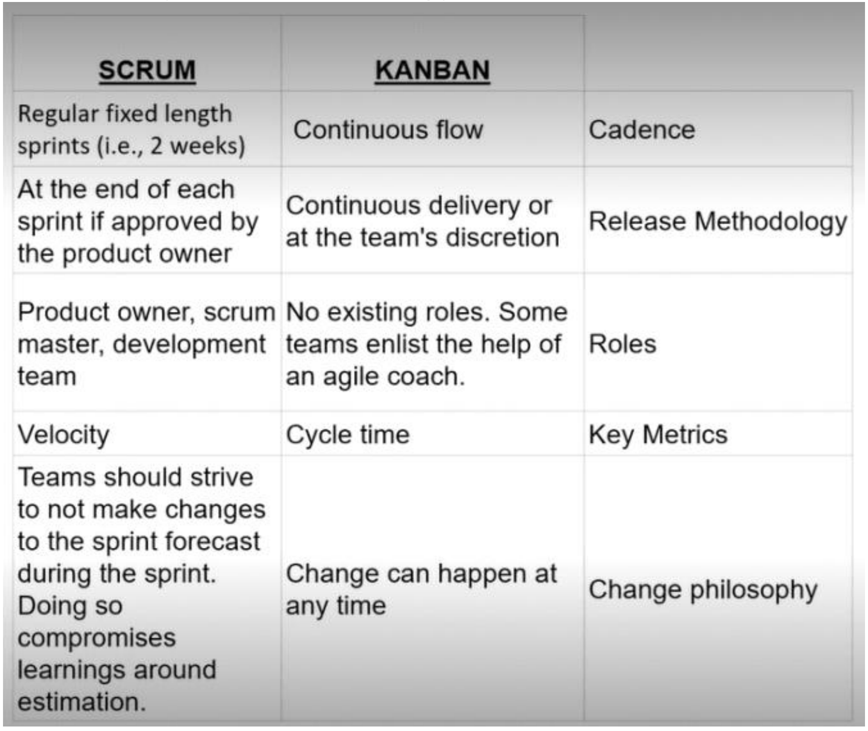
NLS-3334	Loyalty Dashboard - Micro Subscription Implementation-Backend	Story	Medium	ACCEPTED	5
NLS-3345	Dashboard Automation - Data Summary	Story	Medium	ACCEPTED	5
NLS-3346	Dashboard - Weekend Support Monitoring	Story	Medium	ACCEPTED	1
NLS-3347	Job Scheduler - False Failure Alert	Story	Medium	ACCEPTED	1
NLS-3349	CDM Integration - Analysis II	Story	Medium	ACCEPTED	3
NLS-3350	Loyalty Dashboard - Micro Subscription Implementation V2-front-end	Story	Medium	ACCEPTED	5

Issues Not Completed

View in Issue navigator

Key	Summary	Issue Type	Priority	Status	Story Points (5)
NLS-3333	Okta Stage 2 - Implementation	Story	Medium	READY TO START	5

- In Nielsen we are having command data model and hence we are having monthly two releases on the first week and the third week.
- Story point takes 2 days and while task point takes almost 15 to 30 mins
- A tester can raise a ticket as an issue to the developer using agile process. So it will be more effective process between defining priorities.

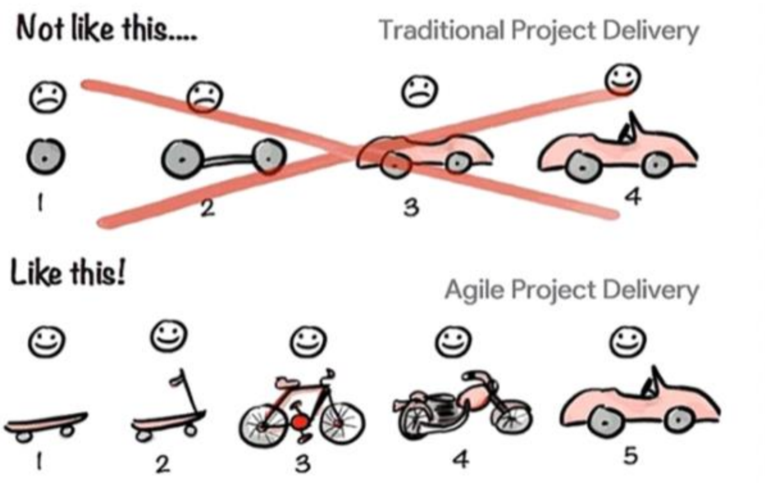


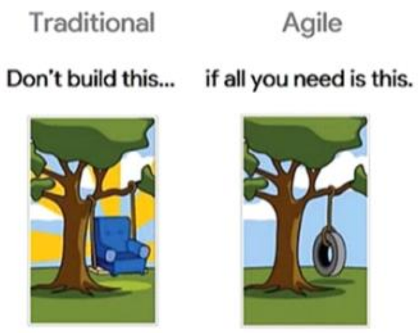
SCRUM	KANBAN	
Regular fixed length sprints (i.e., 2 weeks)	Continuous flow	Cadence
At the end of each sprint if approved by the product owner	Continuous delivery or at the team's discretion	Release Methodology
Product owner, scrum master, development team	No existing roles. Some teams enlist the help of an agile coach.	Roles
Velocity	Cycle time	Key Metrics
Teams should strive to not make changes to the sprint forecast during the sprint. Doing so compromises learnings around estimation.	Change can happen at any time	Change philosophy

- During sprint planning we define story points using some games using Fibonacci series.

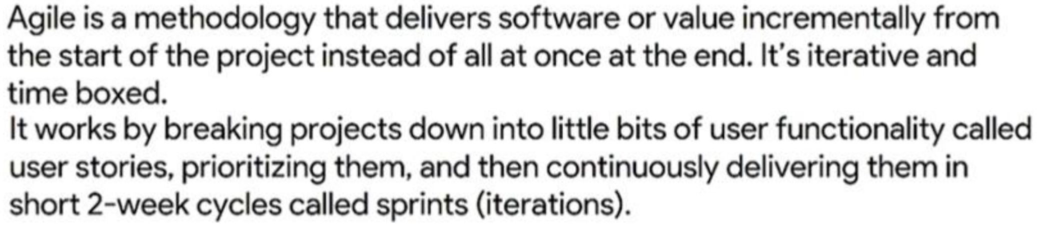
Agile

Agile methods or Agile processes generally promote and encourage frequent **inspection and adaptation,**
a leadership philosophy that encourages **teamwork**, **self-organization and accountability,**
a set of engineering best practices intended to allow for **rapid delivery,**
and a business approach that aligns development with **customer needs and company goals.**



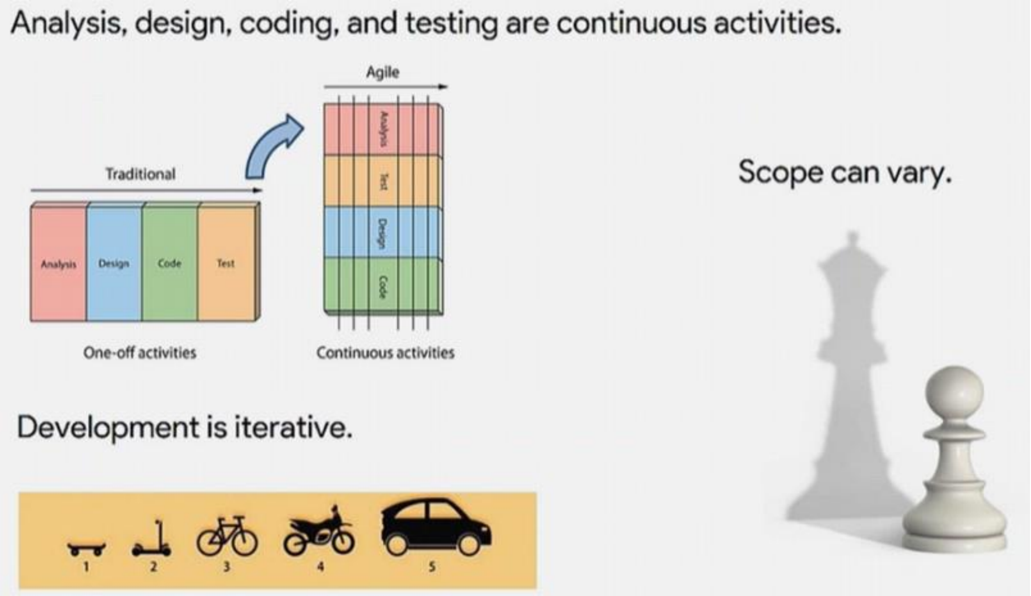


Agile is a Incremental product development approach where each step adds a value


Agile is a methodology that delivers software or value incrementally from the start of the project instead of all at once at the end. It's iterative and time boxed.

It works by breaking projects down into little bits of user functionality called user stories, prioritizing them, and then continuously delivering them in short 2-week cycles called sprints (iterations).

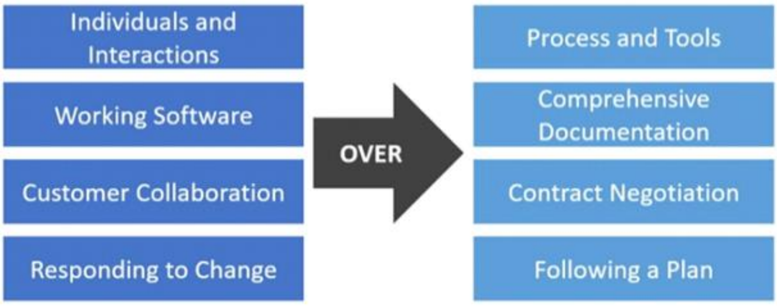
Sprint is also called as iteration cycle.


Analysis, design, coding, and testing are continuous activities.

Scope can vary.

Development is iterative.

- In agile, user can ask adding a feature depends on the sprint cycle


Individuals and Interactions

Working Software

Customer Collaboration

Responding to Change

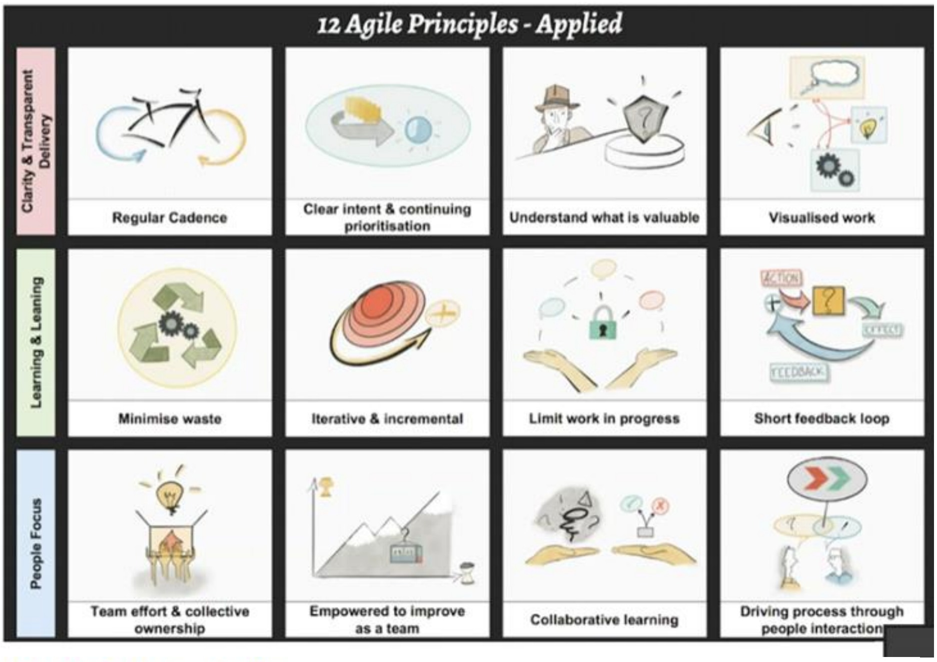
OVER

Process and Tools

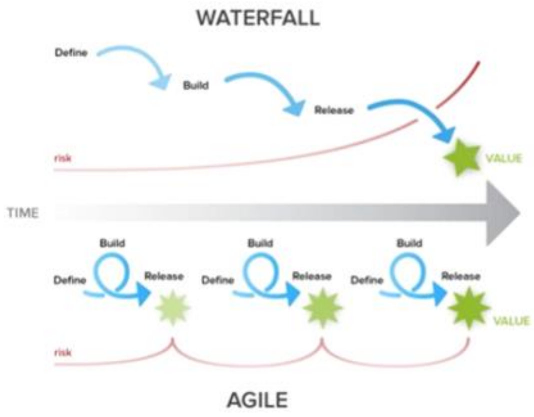
Comprehensive Documentation

Contract Negotiation

Following a Plan


12 Agile Principles - Applied
Clarity & Transparent Delivery
Regular Cadence
Clear intent & continuing prioritisation
Understand what is valuable
Visualised work
Learning & Leaning
Minimise waste
Iterative & incremental
Limit work in progress
Short feedback loop
ACTION
EFFECT
FEEDBACK
People Focus
Team effort & collective ownership
Empowered to improve as a team
Collaborative learning
Driving process through people interaction

Waterfall vs. Agile

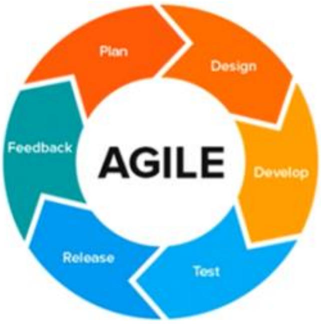


Project execution using Waterfall:

Each phase (Define/Design, Build, Test) of a project must be completed before the next phase can begin. Value is only delivered (released) when the entire project is complete.

Project execution using Agile:

A project is broken down and prioritized into a series of smaller Features that undergo iterative Define/Design, Build, Test and Release cycles. Value is delivered incrementally to the customer until the project is complete.



Key Agile characteristics:

- Just in time (JIT) design
- Superior quality
- Value to the customer
- Faster time to market

User Stories

As a <user> (who) I want... <what> so that... <why>
+ story points + acceptance criteria. Should be independent, negotiable, valuable, estimable, sized appropriately, testable (INVEST).

Story points

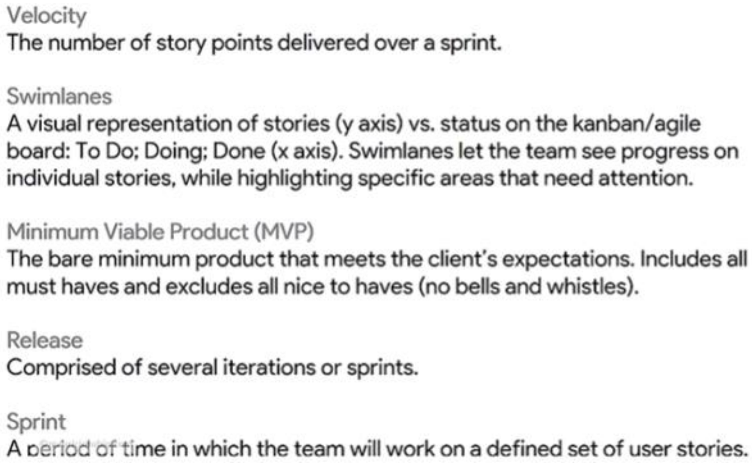
Story points are a relative unit of measure to assess whether a user story is big or small. You can use different scales for story points (e.g. 1, 3 or 5 ; t-shirt sizes s, m, l, xl, xxl).

Product Backlog

An ordered list of everything that might be needed in the product (also known as Master Story List).

Sprint Backlog

The set of Backlog items selected to be delivered over a period of time.



Velocity

The number of story points delivered over a sprint.

Swimlanes

A visual representation of stories (y axis) vs. status on the kanban/agile board: To Do; Doing; Done (x axis). Swimlanes let the team see progress on individual stories, while highlighting specific areas that need attention.

Minimum Viable Product (MVP)

The bare minimum product that meets the client's expectations. Includes all must haves and excludes all nice to haves (no bells and whistles).

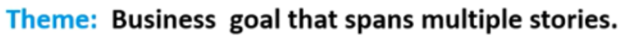
Release

Comprised of several iterations or sprints.

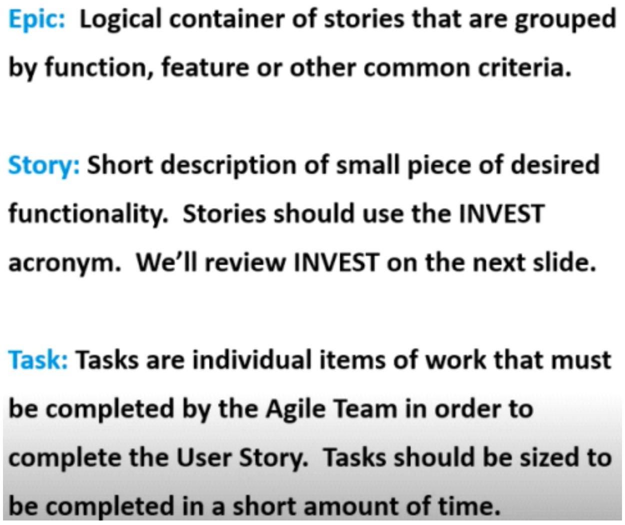
Sprint

A period of time in which the team will work on a defined set of user stories.

User stories tell all the issues , this is the basic block.
Story points defines the difficulty or the work size of the project.
Product backlog mean a list of tasks that we have to do.
Sprint Backlog mean which are needed to deliver in a short period of time.
Velocity is used for predictability of how long the work going to complete.
Swim lanes are some flags on user stories which tells their status of the story.
Minimum viable product is like the base product that customer will accept at least.
Release is comprised of several iterations or sprints.
Velocity is arranged depends on the team. If we are having experts in it, we will have high velocity.
Or else it will be low if we are having a limited expertise people.



Theme: **Business goal that spans multiple stories.**



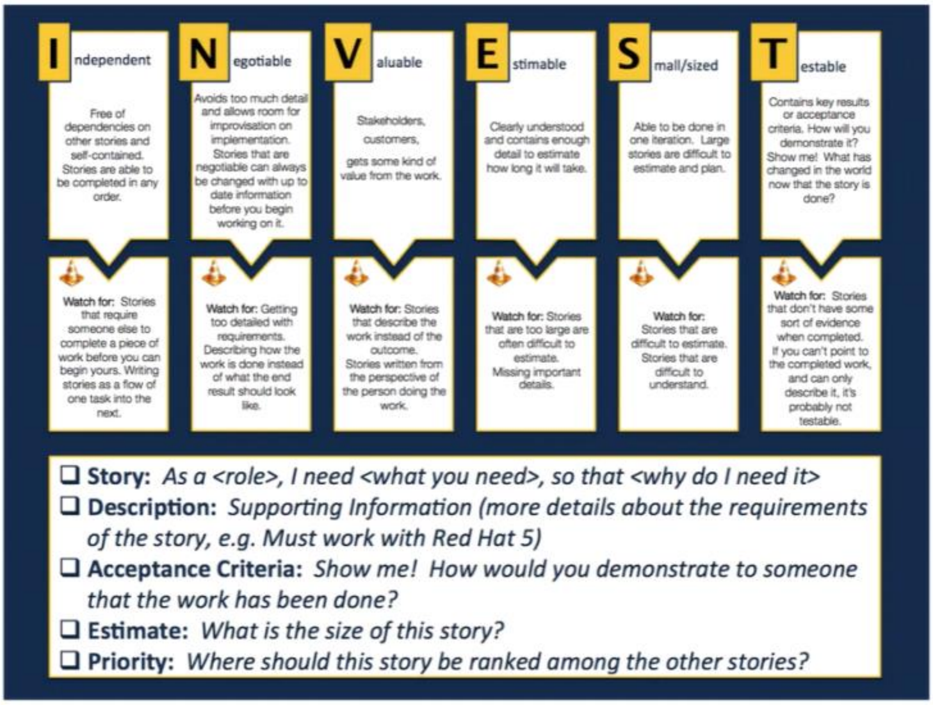
Epic: **Logical container of stories that are grouped by function, feature or other common criteria.**

Story: **Short description of small piece of desired functionality. Stories should use the INVEST acronym. We'll review INVEST on the next slide.**

Task: **Tasks are individual items of work that must be completed by the Agile Team in order to complete the User Story. Tasks should be sized to be completed in a short amount of time.**

- We must take the user stories which doesn't have dependencies because it slows the sprint cycle production level.

User Stories - INVEST model



Independent

Free of dependencies on other stories and self-contained. Stories are able to be completed in any order.

Watch for: Stories that require someone else to complete a piece of work before you can begin yours. Writing stories as a flow of one task into the next.

Negotiable

Avoids too much detail and allows room for improvisation on implementation. Stories that are negotiable can always be changed with up to date information before you begin working on it.

Watch for: Getting too detailed with requirements. Describing how the work is done instead of what the end result should look like.

Valuable

Stakeholders, customers, gets some kind of value from the work.

Watch for: Stories that describe the work instead of the outcome. Stories written from the perspective of the person doing the work.

Estimable

Clearly understood and contains enough detail to estimate how long it will take.

Watch for: Stories that are too large are often difficult to estimate. Missing important details.

Small/sized

Able to be done in one iteration. Large stories are difficult to estimate and plan.

Watch for: Stories that are difficult to estimate. Stories that are difficult to understand.

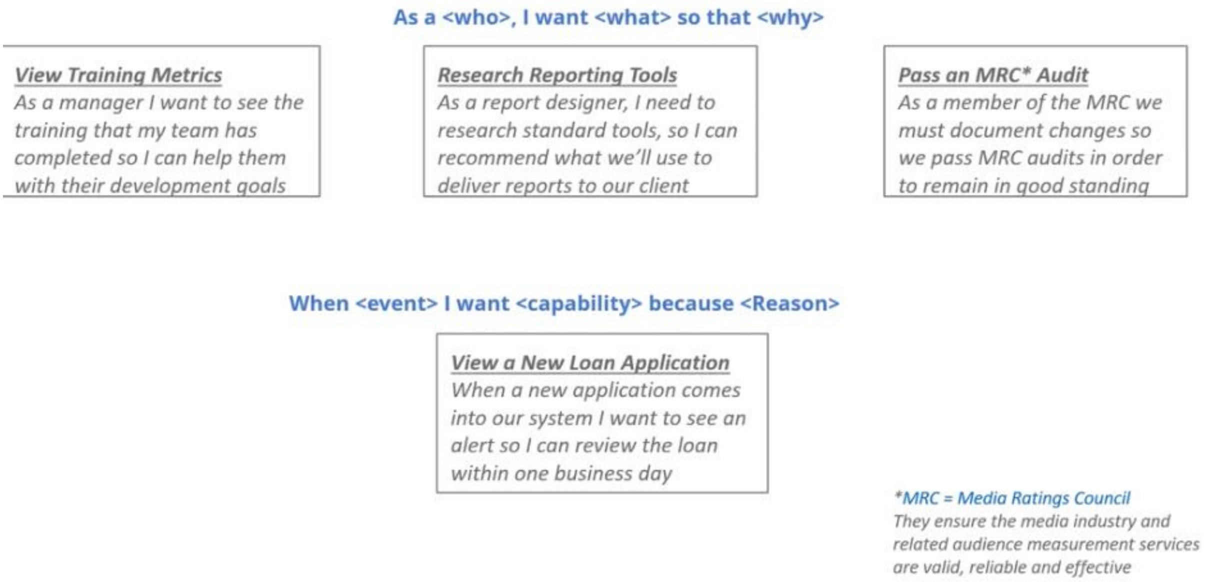
Testable

Contains key results or acceptance criteria. How will you demonstrate it? Show me! What has changed in the world now that the story is done?

Watch for: Stories that don't have some sort of evidence when completed. If you can't point to the completed work, and can only describe it, it's probably not testable.

- ❑ **Story:** *As a <role>, I need <what you need>, so that <why do I need it>*
- ❑ **Description:** *Supporting Information (more details about the requirements of the story, e.g. Must work with Red Hat 5)*
- ❑ **Acceptance Criteria:** *Show me! How would you demonstrate to someone that the work has been done?*
- ❑ **Estimate:** *What is the size of this story?*
- ❑ **Priority:** *Where should this story be ranked among the other stories?*

User Story examples



As a <who>, I want <what> so that <why>

View Training Metrics
As a manager I want to see the training that my team has completed so I can help them with their development goals

Research Reporting Tools
As a report designer, I need to research standard tools, so I can recommend what we'll use to deliver reports to our client

Pass an MRC* Audit
As a member of the MRC we must document changes so we pass MRC audits in order to remain in good standing

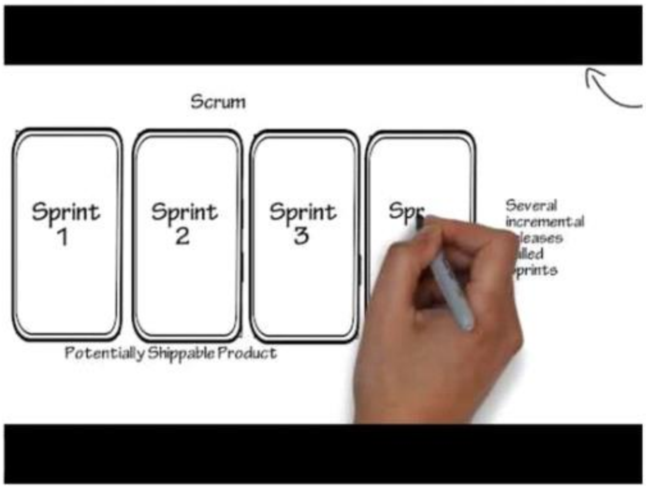
When <event> I want <capability> because <Reason>

View a New Loan Application
When a new application comes into our system I want to see an alert so I can review the loan within one business day

**MRC = Media Ratings Council*
They ensure the media industry and related audience measurement services are valid, reliable and effective

Common Agile Methodologies: Scrum, Kanban, Extreme Programming (XP)

Agile practices (including Scrum and Kanban methods), can be used for any type of work execution. They do not have to be limited to development projects and technical teams.



What is Scrum?

Scrum (n): **A widely used framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value. Scrum is a subset of Agile**.

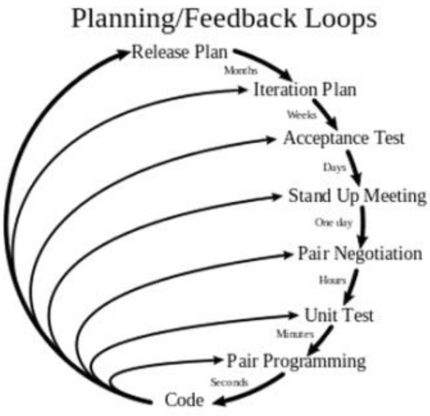
Scrum places emphasis on software development and is especially effective when work can be prioritized in batches/iterations.

Sprints are the basic unit of work execution (timebox). Finished portions of a product are created in each sprint

XP Basics

XP is a disciplined approach to delivering high-quality software quickly and continuously.

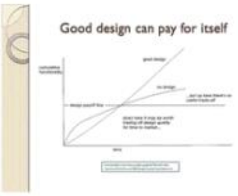
- It promotes high customer involvement, rapid feedback loops, continuous testing, continuous planning, and close teamwork to deliver working software at very frequent intervals, typically every 1-3 weeks.



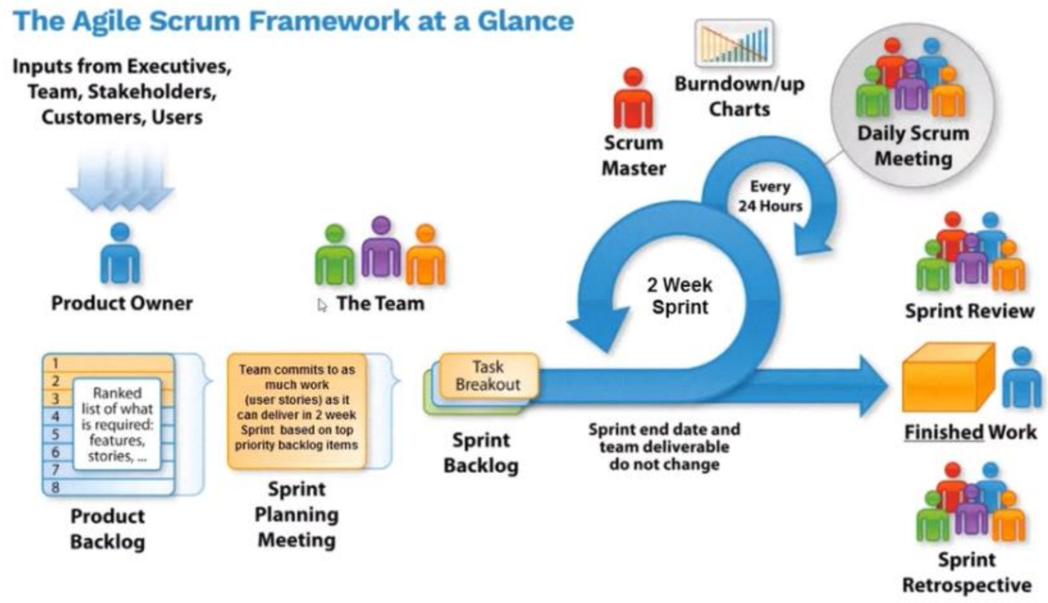
What you'll see when working with XP

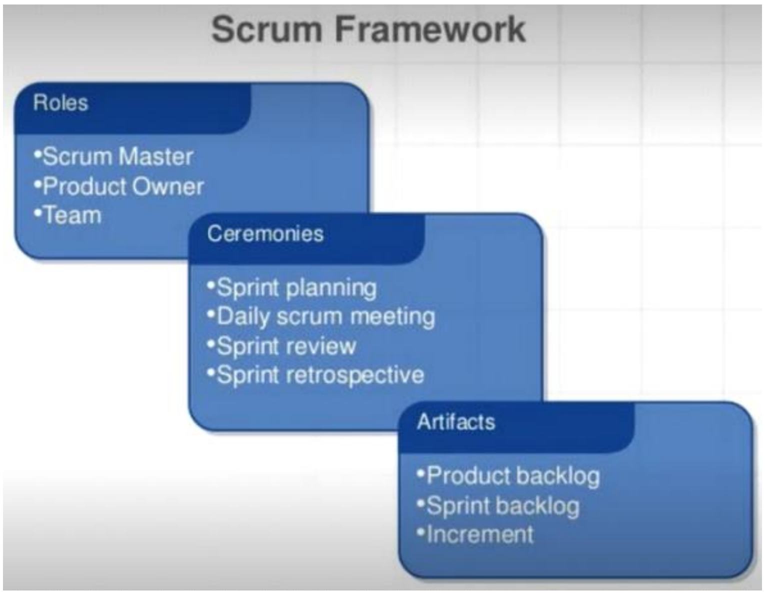
- The whole team starts together. The product owner is embedded throughout the entire project
- The team embraces change
- Paired programming (strictly an XP practice)
- Continuous integration; small releases
- Coding standards
- Refactoring - improving code without changing the features it implements.

Pros and cons of XP:
(video ~ 7 minutes)



Scrum Basics


The Agile Scrum Framework at a Glance
Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Scrum Master
Burndown/up Charts
Daily Scrum Meeting
Every 24 Hours
2 Week Sprint
Sprint Review
1
2
3
4
5
6
7
8
Ranked list of what is required: features, stories, ...
Product Backlog
Team commits to as much work (user stories) as it can deliver in 2 week Sprint based on top priority backlog items
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Sprint Retrospective



Scrum Framework

Roles

- Scrum Master
- Product Owner
- Team

Ceremonies

- Sprint planning
- Daily scrum meeting
- Sprint review
- Sprint retrospective

Artifacts

- Product backlog
- Sprint backlog
- Increment

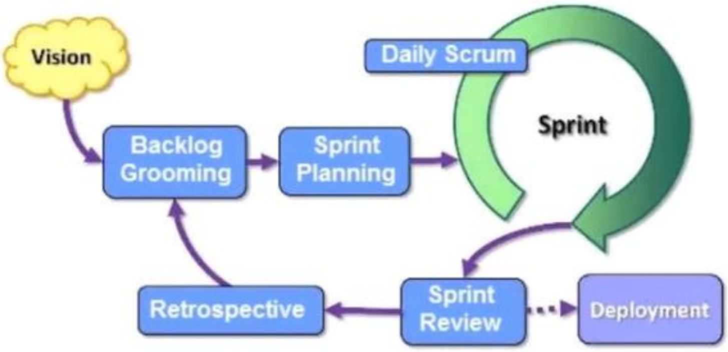
Scrum master is typically a servant leader where he manages the team, and checks are the tasks are being achieved or not. And keep track of the progress of the team. And product owner can be a person who can be from the customer's company who has idea about what are the requirements they need as a product to design. Scrum master helps us to prioritize the tasks in designing a completing product. SCRUM Master have connections with different team , which helps us in intergroup communication.

Sprint Planning - Scrum Team takes top priority items from larger "**Product Backlog**" and plans a Sprint iteration (e.g. 2-week period of time when work is executed upon). The "**Sprint Backlog**" is created at this time.

Daily Scrum - Scrum Team meets for 15 minutes each day (aka "Daily Stand Up") to review progress, align as a team, and also discuss any impediments encountered.

Sprint Review - Scrum Team meets to review (demo) completed work to Product Owner for PO Acceptance (per Acceptance Criteria defined in each Story). Completed stories contribute to a "**Product Increment**".

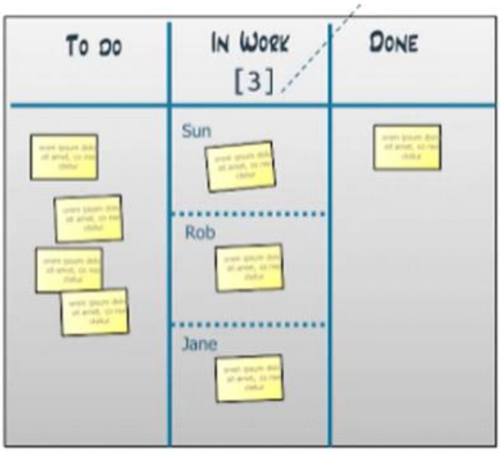
Sprint Retrospective - Scrum Team discusses the Agile "process" and determines what the team should continue doing, stop doing and start doing in order to "inspect and adapt" their agile practices and improve upon them.



Backlog Grooming - This meeting and process doesn't occur at a prescribed time. It is periodically performed (as needed) by Product Owner (w/input from team) to refine and prioritize the Product Backlog (Epics/Features, Stories). Development team estimates size of top priority Stories in the backlog (once they are refined).

Kanban Basics

Emphasis is on continuous delivery



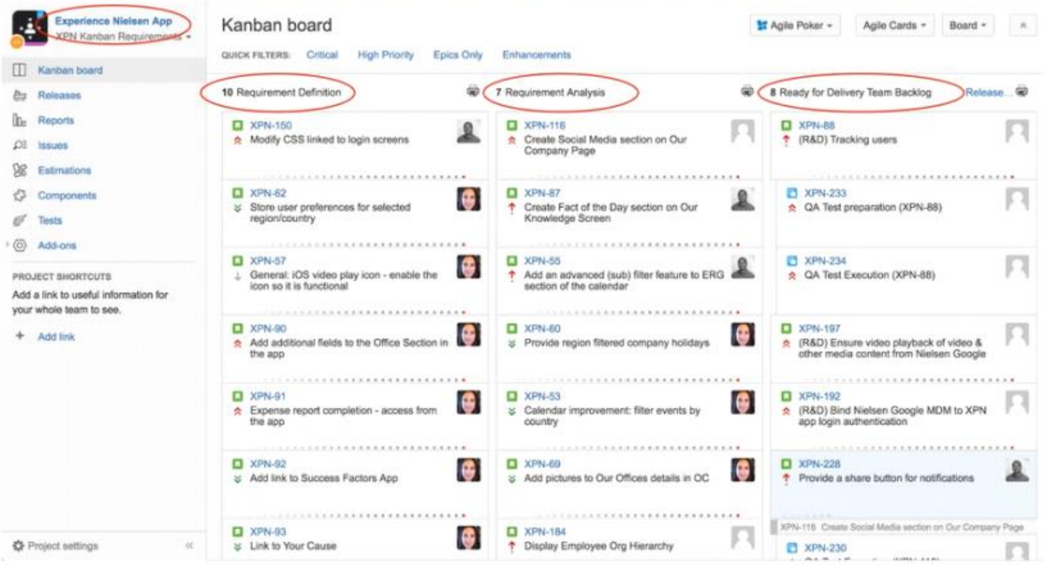
Three Key Kanban Principles:

1. Visualize what you are doing by moving between phases of work
2. Concentrate on finishing work not starting work.
3. Inspect and adapt to improve the flow of work. Identify potential bottlenecks in your process and resolve them so work can flow through the phases cost-effectively and at optimal speed.

Kanban can be used in conjunction with another methodology like Scrum, or it can be used on its own

Kanban Boards - JIRA Software Tool

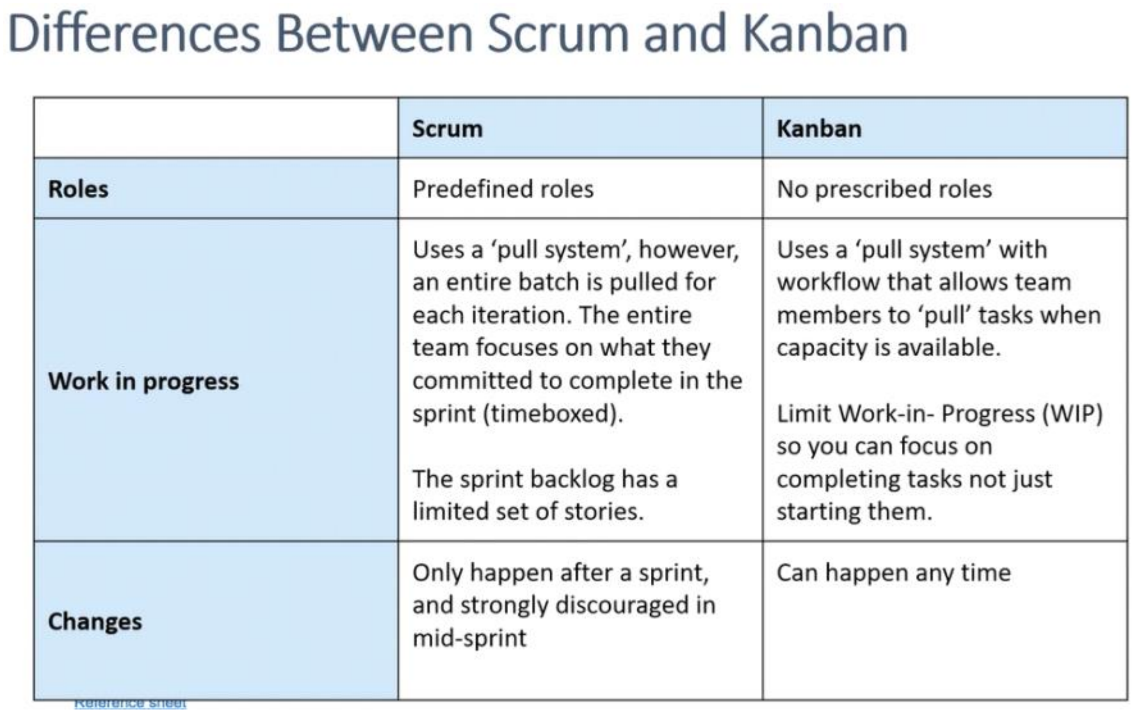
Jira's Kanban Board with lanes to represent different stages/phases of software development



Kanban board

10 Requirement Definition	7 Requirement Analysis	8 Ready for Delivery Team Backlog
XPN-150 Modify CSS linked to login screens	XPN-116 Create Social Media section on Our Company Page	XPN-88 (R&D) Tracking users
XPN-62 Store user preferences for selected region/country	XPN-87 Create Fact of the Day section on Our Knowledge Screen	XPN-233 QA Test preparation (XPN-88)
XPN-57 General: iOS video play icon - enable the icon so it is functional	XPN-55 Add an advanced (sub) filter feature to ERG section of the calendar	XPN-234 QA Test Execution (XPN-88)
XPN-90 Add additional fields to the Office Section in the app	XPN-60 Provide region filtered company holidays	XPN-197 (R&D) Ensure video playback of video & other media content from Nielsen Google
XPN-91 Expense report completion - access from the app	XPN-53 Calendar improvement: filter events by country	XPN-192 (R&D) Bind Nielsen Google MDM to XPN app login authentication
XPN-92 Add link to Success Factors App	XPN-69 Add pictures to Our Offices details in OC	XPN-228 Provide a share button for notifications
XPN-93 Link to Your Cause	XPN-184 Display Employee Org Hierarchy	XPN-116 Create Social Media section on Our Company Page XPN-230

This is a Kanban board used with Scrum by a software development team (Nielsen Enterprise IT), created in Jira

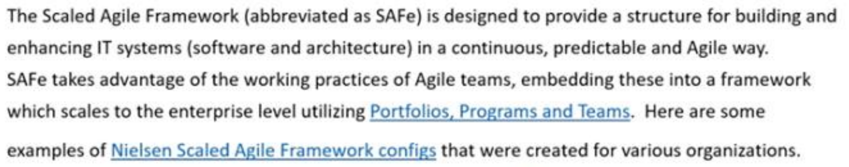


Differences Between Scrum and Kanban

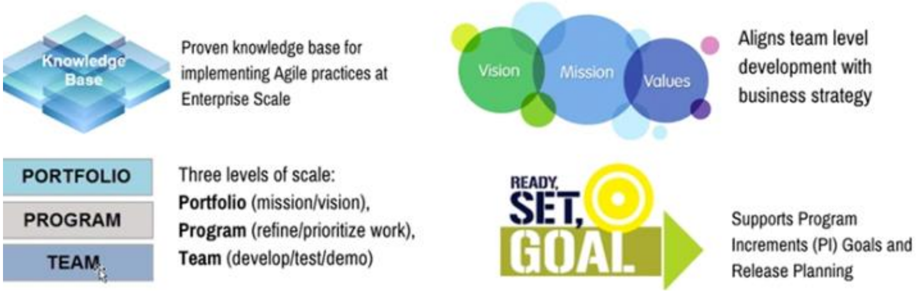
	Scrum	Kanban
Roles	Predefined roles	No prescribed roles
Work in progress	Uses a 'pull system', however, an entire batch is pulled for each iteration. The entire team focuses on what they committed to complete in the sprint (timeboxed). The sprint backlog has a limited set of stories.	Uses a 'pull system' with workflow that allows team members to 'pull' tasks when capacity is available. Limit Work-in- Progress (WIP) so you can focus on completing tasks not just starting them.
Changes	Only happen after a sprint, and strongly discouraged in mid-sprint	Can happen any time

Reference sheet

Scaled Agile Framework



The Scaled Agile Framework (abbreviated as SAFe) is designed to provide a structure for building and enhancing IT systems (software and architecture) in a continuous, predictable and Agile way. SAFe takes advantage of the working practices of Agile teams, embedding these into a framework which scales to the enterprise level utilizing Portfolios, Programs and Teams. Here are some examples of Nielsen Scaled Agile Framework configs that were created for various organizations.



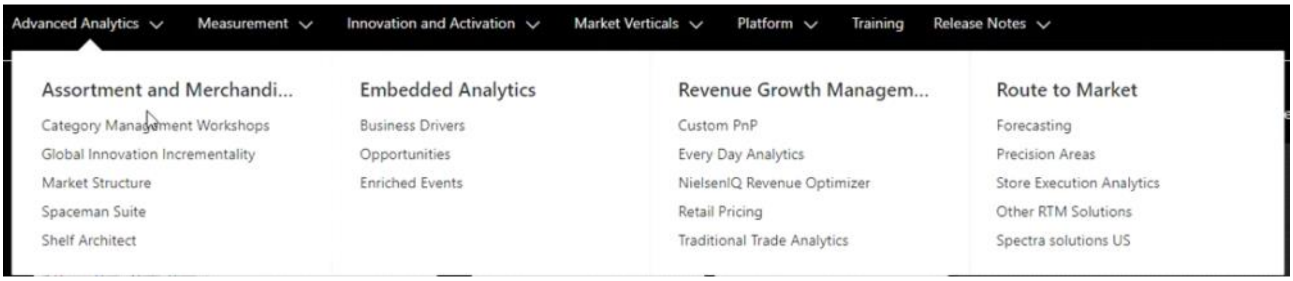
Proven knowledge base for implementing Agile practices at Enterprise Scale

Aligns team level development with business strategy

Three levels of scale:
Portfolio (mission/vision),
Program (refine/prioritize work),
Team (develop/test/demo)

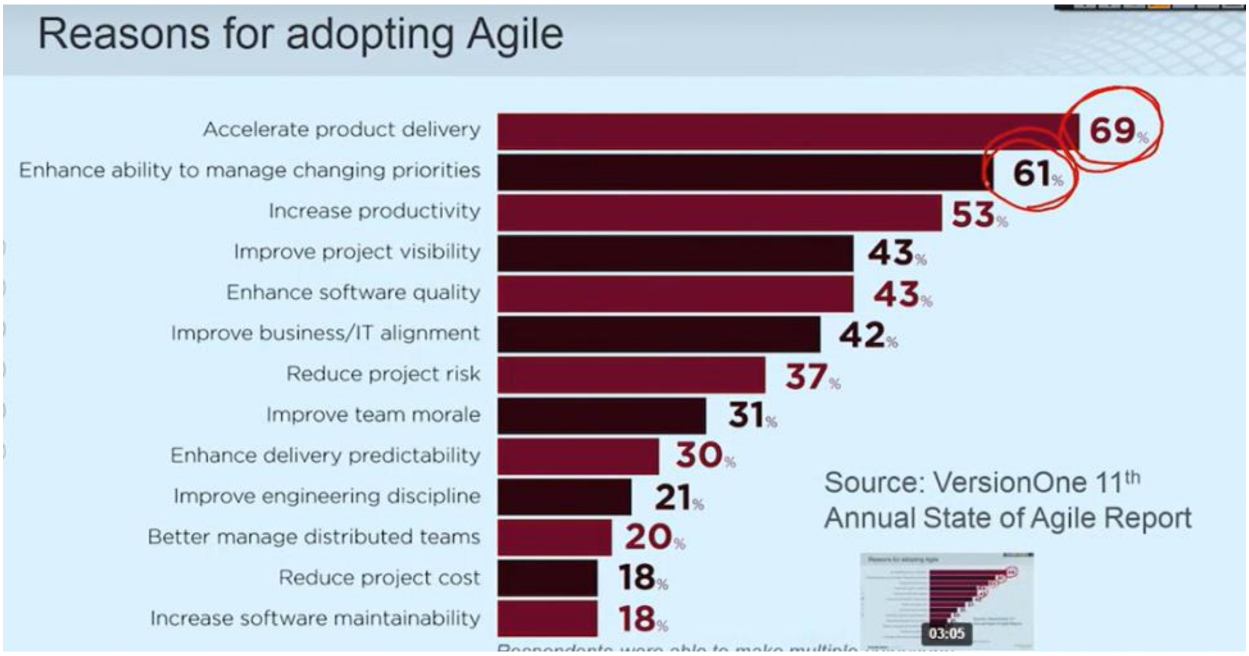
Supports Program Increments (PI) Goals and Release Planning

Program is a cross functional and do changes on different products.
Portfolio is the higher version of program. Where we can only see the abstract version of programs.
Example: all the products that serves retailer is a portfolio, all the products that serves consumer is a portfolio
Safe agile is a large scale agile process where agile is not condensed to the teams it will work till the CEO level.



For example : Market Structure is a team , Assortment and Merchandizing is program and Advanced analytics is a portfolio.
CEO checks the growth of portfolio's.

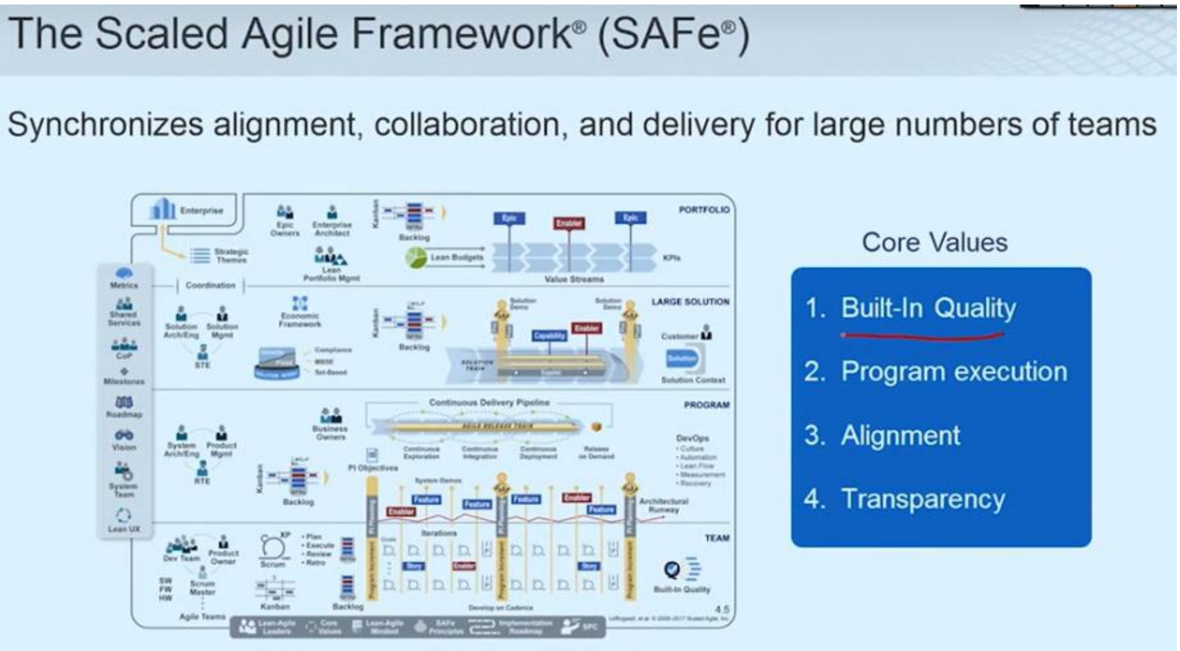
SAFe Agile Structure: (SAFe 5 for Lean Enterprises (scaledagileframework.com))



Reasons for adopting Agile

Source: VersionOne 11th Annual State of Agile Report

Respondents were able to make multiple



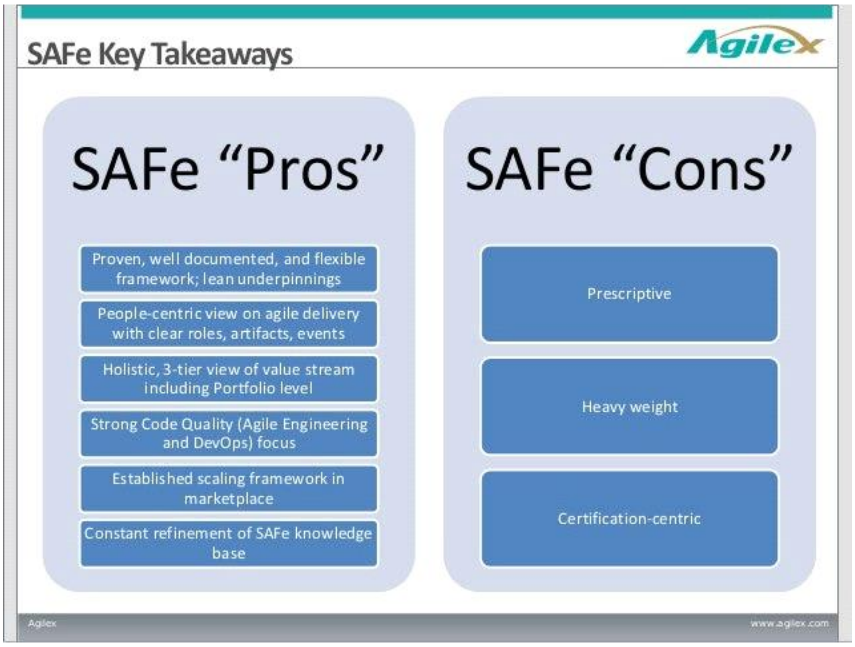
The Scaled Agile Framework® (SAFe®)

Synchronizes alignment, collaboration, and delivery for large numbers of teams

Core Values

1. Built-In Quality
2. Program execution
3. Alignment
4. Transparency

	Agile	**SAFe Agile**
Framework	Agile frameworks are built on Agile principles	SAFe is a scaled-up framework of Agile principles
Role of management	Management plays only a supporting role in everyday functions. They are critical during the transition to Agile.	There are management roles. For example, Portfolio managers are responsible for the enterprise's financial, technical, and business aspects. The Product Manager at the program level is responsible for prioritizing features of the product and making sure they are well-understood.
Team Organisation	Agile teams have no sub-teams or hierarchies. They focus on completing small goals one at a time towards creating the end-product.	SAFe frameworks have ART – Agile Release Trains, which are virtual organizations of Agile teams that plan and work together on the same timeline to achieve the Value Stream goals.
Decision making	Independent decision making is encouraged.	Decisions can be made either at the team level or at the enterprise level, depending on the scope of the issue.
Planning	Planning is done not for the entire project but for shorter durations. The process is not complicated or long	Planning and the development cycle are long processes because of the focus on the 'big picture.'
Ability to change	Agile methods are planned around flexibility and change. Products can be improved after each iteration.	Making changes in a SAFe process is possible but not as easy to execute because of the large scale.
Role of team members	Agile relies on the commitment and expertise of individual team members to work with dedication. It is not always a practical expectation, and project deadlines may be missed due to some weak links in the team.	Since there are established leadership roles, large-scale projects can be completed with the development teams' involvement and top management.
Alignment with business goals	Since this approach is more bottom-up, developers and testers focussed on their tasks can lose sight of business objectives.	The top-down alignment and management involvement in making strategic decisions ensure that business objectives are met.



SAFe Key Takeaways

SAFe "Pros"

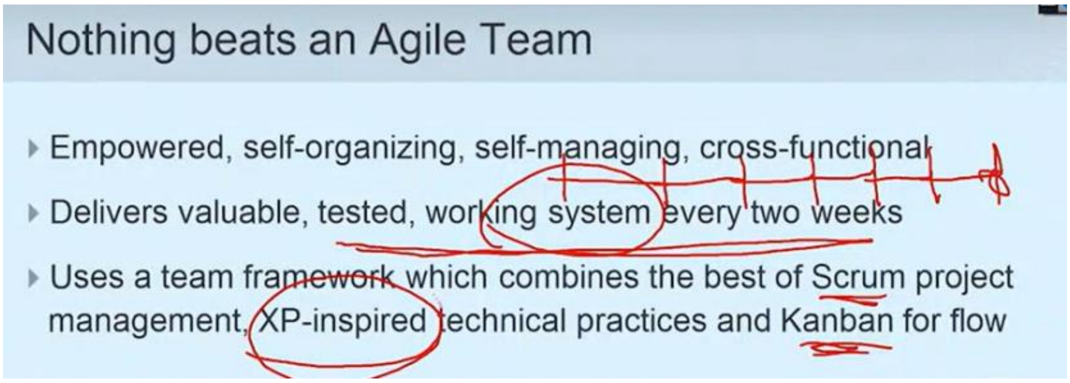
- Proven, well documented, and flexible framework; lean underpinnings
- People-centric view on agile delivery with clear roles, artifacts, events
- Holistic, 3-tier view of value stream including Portfolio level
- Strong Code Quality (Agile Engineering and DevOps) focus
- Established scaling framework in marketplace
- Constant refinement of SAFe knowledge base

SAFe "Cons"

- Prescriptive
- Heavy weight
- Certification-centric

Agilex www.agilex.com

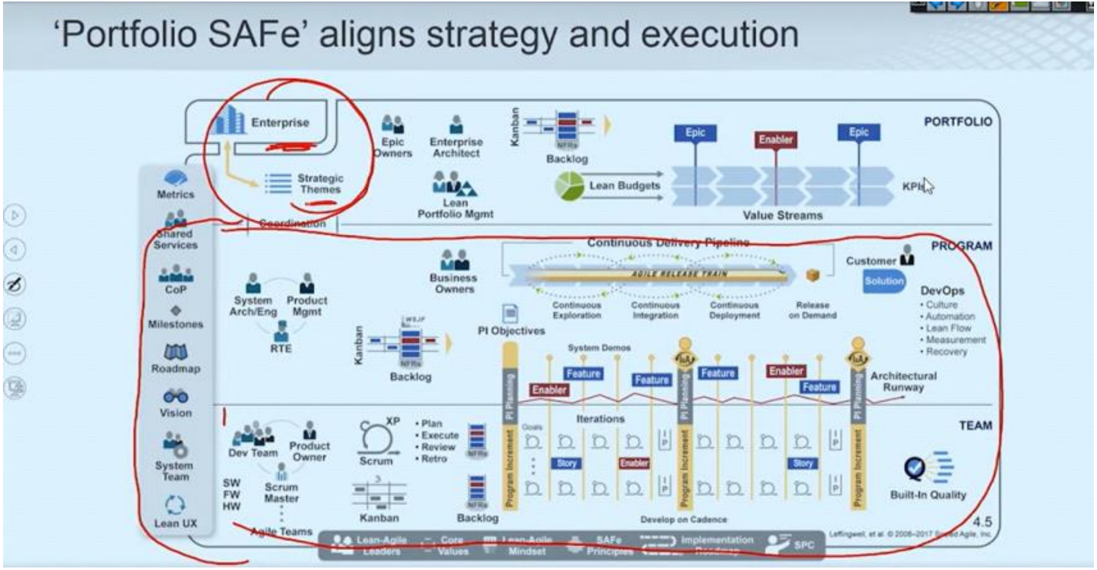
SAFe application:



Nothing beats an Agile Team

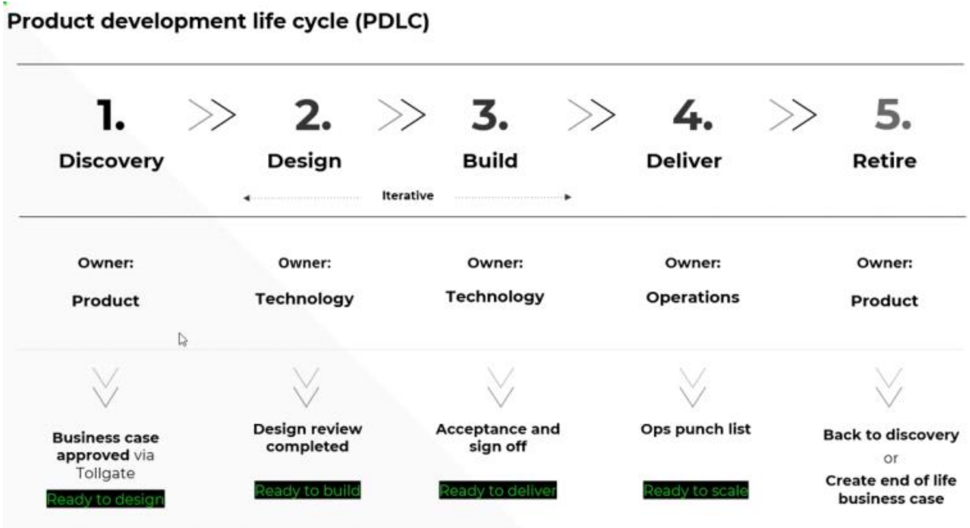
- Empowered, self-organizing, self-managing, cross-functional
- Delivers valuable, tested, working system every two weeks
- Uses a team framework which combines the best of Scrum project management, XP-inspired technical practices and Kanban for flow

- This is the three level program that we are talking (Team , Program , Portfolio)
- Portfolio is observed by the enterprise and other respective layers will be observed by their respective managers and leaders that are mentioned at left of the level of diagram.



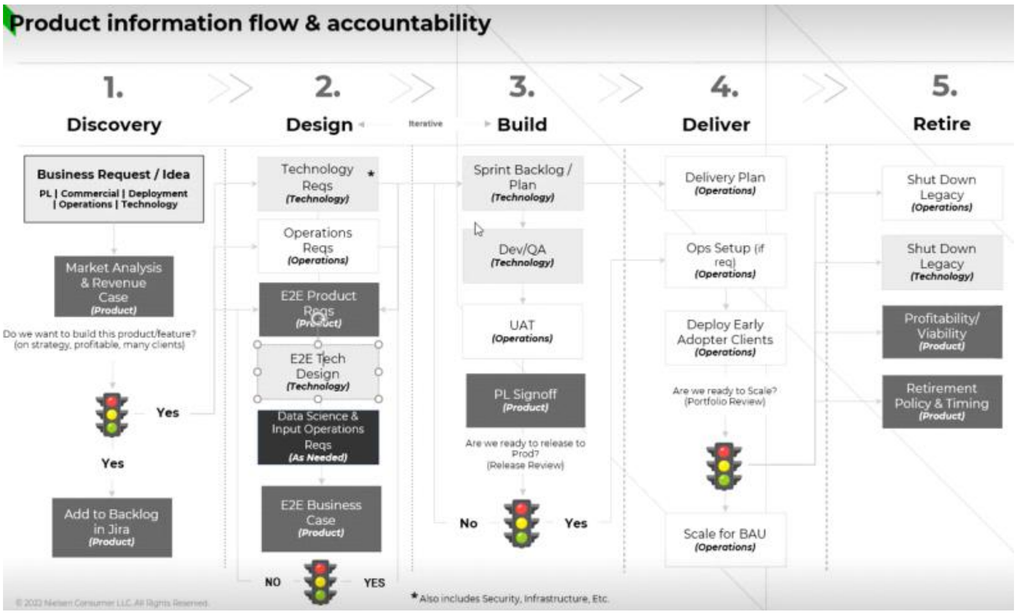
‘Portfolio SAFe’ aligns strategy and execution

Product development life cycle.



Product development life cycle (PDLC)

1. Discovery	2. Design	3. Build	4. Deliver	5. Retire
	◀ Iterative ▶			
Owner: Product	Owner: Technology	Owner: Technology	Owner: Operations	Owner: Product
Business case approved via Tollgate Ready to design	Design review completed Ready to build	Acceptance and sign off Ready to deliver	Ops punch list Ready to scale	Back to discovery or Create end of life business case

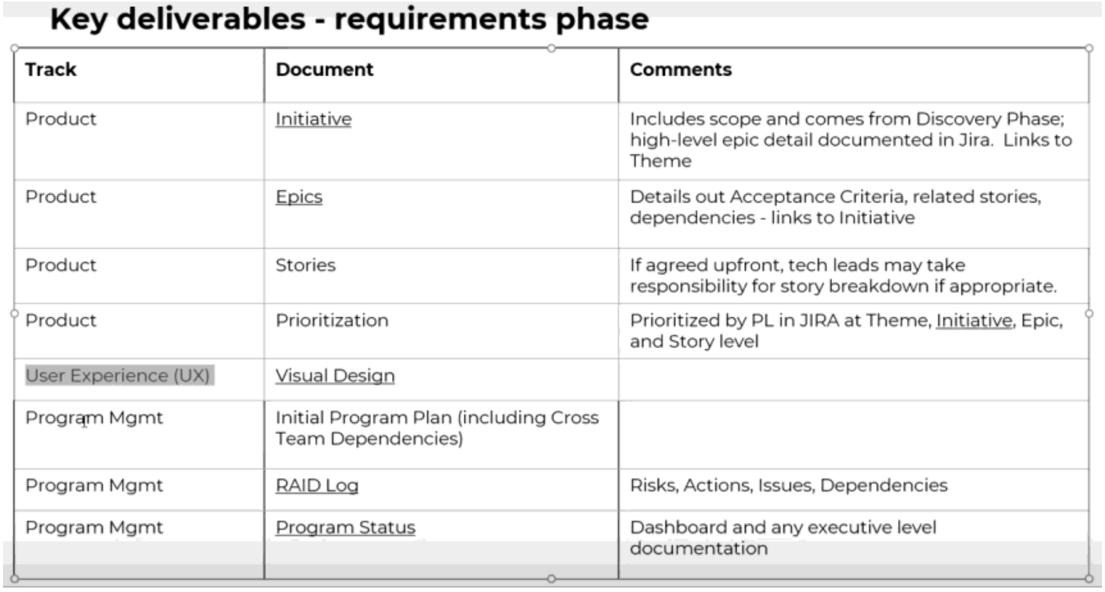


Product information flow & accountability

1. Discovery
2. Design
3. Build
4. Deliver
5. Retire

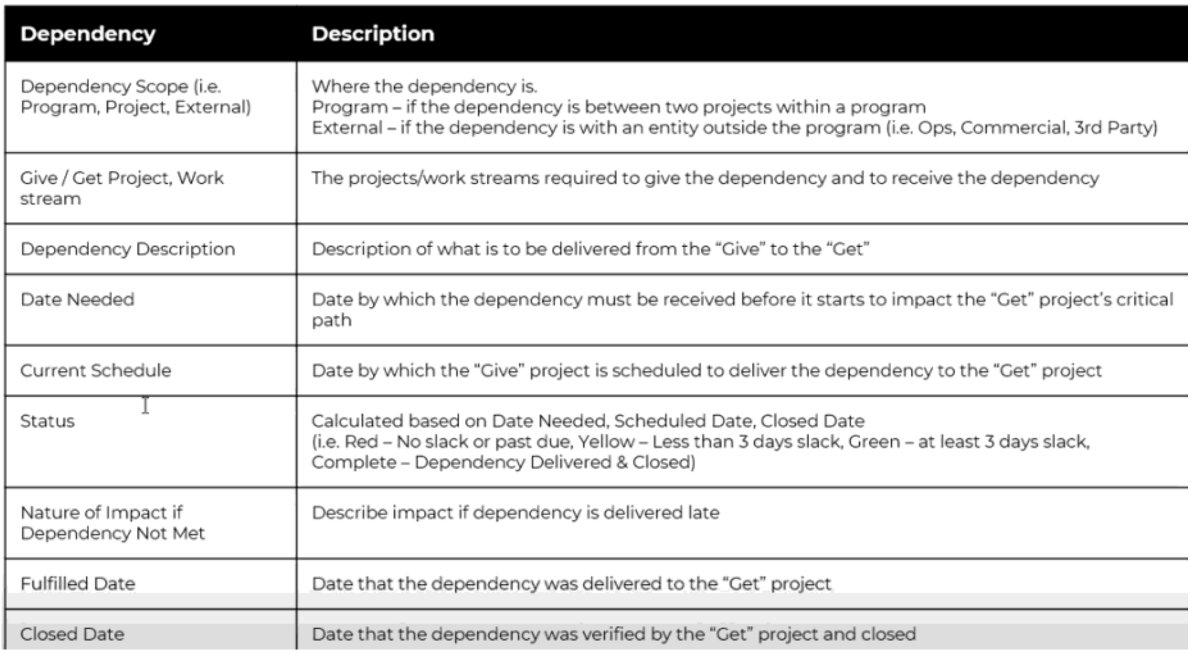
Iterative
Business Request / Idea
PL | Commercial | Deployment | Operations | Technology
Market Analysis & Revenue Case (Product)
Do we want to build this product/feature? (on strategy, profitable, many clients)
Yes
Yes
Add to Backlog in Jira (Product)
Technology Reqs (Technology) *
Operations Reqs (Operations)
E2E Product Reqs (Product)
E2E Tech Design (Technology)
Data Science & Input Operations Reqs (As Needed)
E2E Business Case (Product)
NO
YES
Sprint Backlog / Plan (Technology)
Dev/QA (Technology)
UAT (Operations)
PL Signoff (Product)
Are we ready to release to Prod? (Release Review)
No
Yes
Delivery Plan (Operations)
Ops Setup (if req) (Operations)
Deploy Early Adopter Clients (Operations)
Are we ready to Scale? (Portfolio Review)
Scale for BAU (Operations)
Shut Down Legacy (Operations)
Shut Down Legacy (Technology)
Profitability/ Viability (Product)
Retirement Policy & Timing (Product)

* Also includes Security, Infrastructure, Etc.

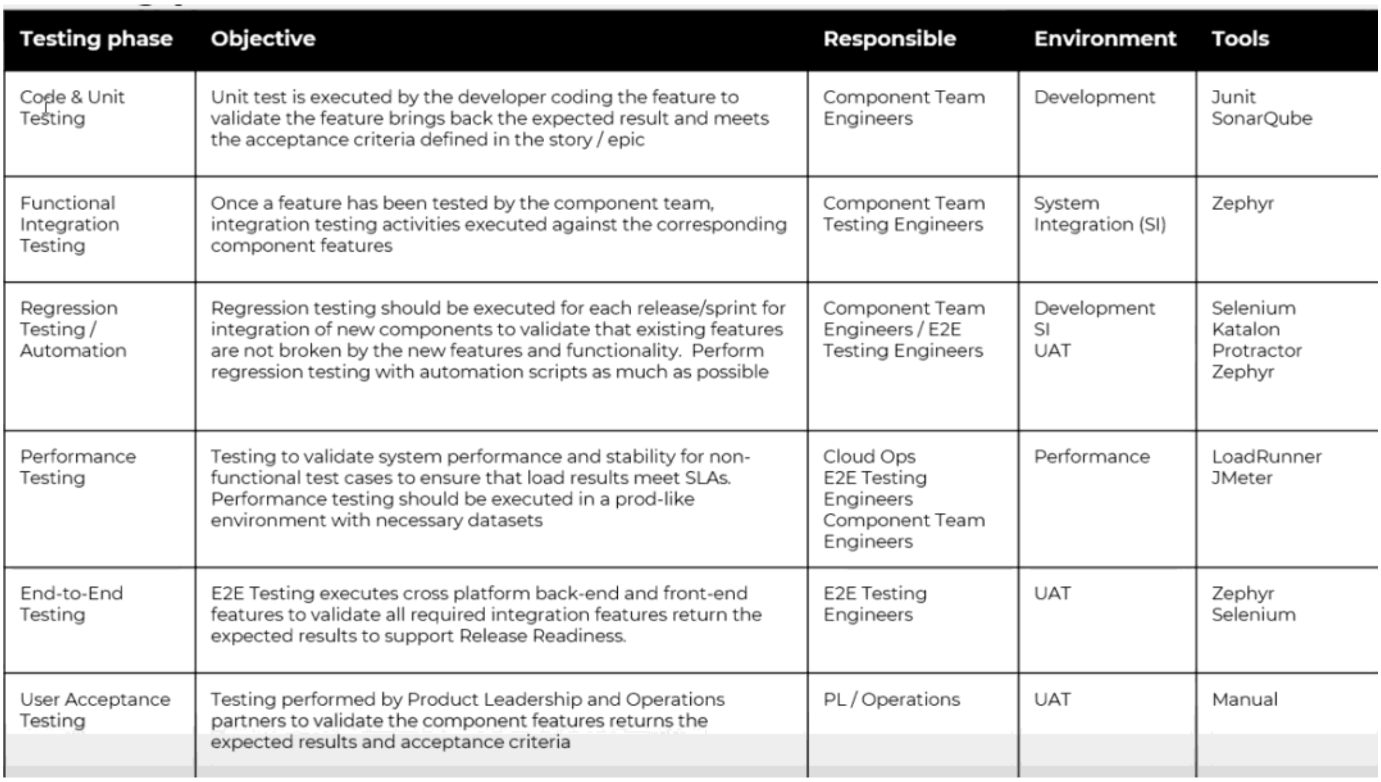


Key deliverables - requirements phase

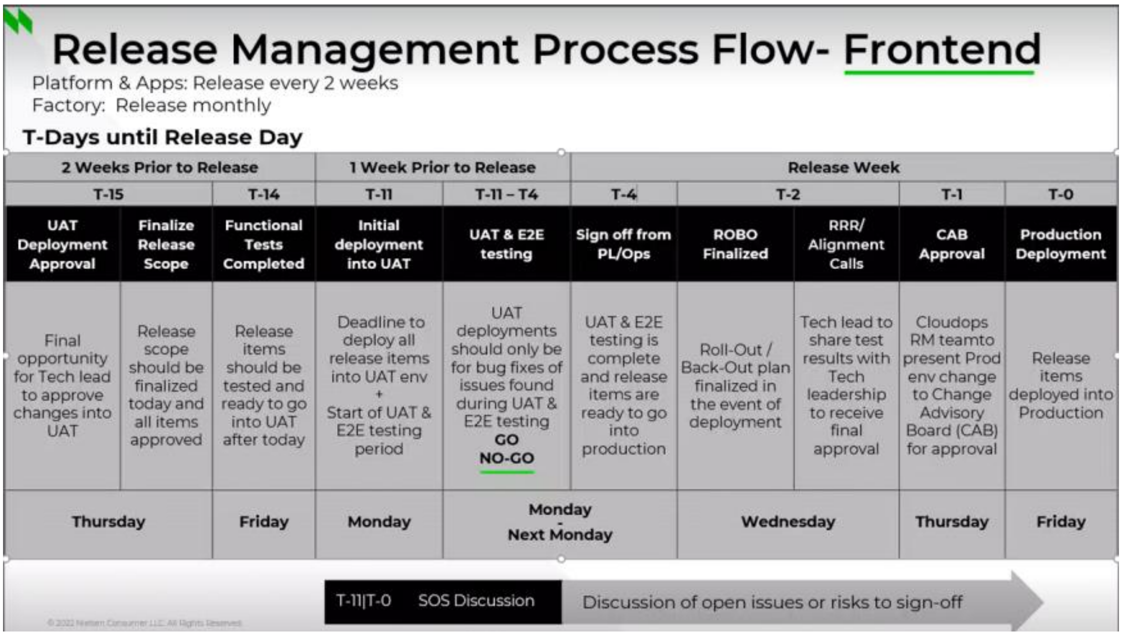
Track	Document	Comments
Product	Initiative	Includes scope and comes from Discovery Phase; high-level epic detail documented in Jira. Links to Theme
Product	Epics	Details out Acceptance Criteria, related stories, dependencies - links to Initiative
Product	Stories	If agreed upfront, tech leads may take responsibility for story breakdown if appropriate.
Product	Prioritization	Prioritized by PL in JIRA at Theme, Initiative, Epic, and Story level
User Experience (UX)	Visual Design	
Program Mgmt	Initial Program Plan (including Cross Team Dependencies)	
Program Mgmt	RAID Log	Risks, Actions, Issues, Dependencies
Program Mgmt	Program Status	Dashboard and any executive level documentation



Dependency	Description
Dependency Scope (i.e. Program, Project, External)	Where the dependency is. Program – if the dependency is between two projects within a program External – if the dependency is with an entity outside the program (i.e. Ops, Commercial, 3rd Party)
Give / Get Project, Work stream	The projects/work streams required to give the dependency and to receive the dependency
Dependency Description	Description of what is to be delivered from the "Give" to the "Get"
Date Needed	Date by which the dependency must be received before it starts to impact the "Get" project's critical path
Current Schedule	Date by which the "Give" project is scheduled to deliver the dependency to the "Get" project
Status	Calculated based on Date Needed, Scheduled Date, Closed Date (i.e. Red – No slack or past due, Yellow – Less than 3 days slack, Green – at least 3 days slack, Complete – Dependency Delivered & Closed)
Nature of Impact if Dependency Not Met	Describe impact if dependency is delivered late
Fulfilled Date	Date that the dependency was delivered to the "Get" project
Closed Date	Date that the dependency was verified by the "Get" project and closed



Testing phase	Objective	Responsible	Environment	Tools
Code & Unit Testing	Unit test is executed by the developer coding the feature to validate the feature brings back the expected result and meets the acceptance criteria defined in the story / epic	Component Team Engineers	Development	Junit SonarQube
Functional Integration Testing	Once a feature has been tested by the component team, integration testing activities executed against the corresponding component features	Component Team Testing Engineers	System Integration (SI)	Zephyr
Regression Testing / Automation	Regression testing should be executed for each release/sprint for integration of new components to validate that existing features are not broken by the new features and functionality. Perform regression testing with automation scripts as much as possible	Component Team Engineers / E2E Testing Engineers	Development SI UAT	Selenium Katalon Protractor Zephyr
Performance Testing	Testing to validate system performance and stability for non-functional test cases to ensure that load results meet SLAs. Performance testing should be executed in a prod-like environment with necessary datasets	Cloud Ops E2E Testing Engineers Component Team Engineers	Performance	LoadRunner JMeter
End-to-End Testing	E2E Testing executes cross platform back-end and front-end features to validate all required integration features return the expected results to support Release Readiness.	E2E Testing Engineers	UAT	Zephyr Selenium
User Acceptance Testing	Testing performed by Product Leadership and Operations partners to validate the component features returns the expected results and acceptance criteria	PL / Operations	UAT	Manual



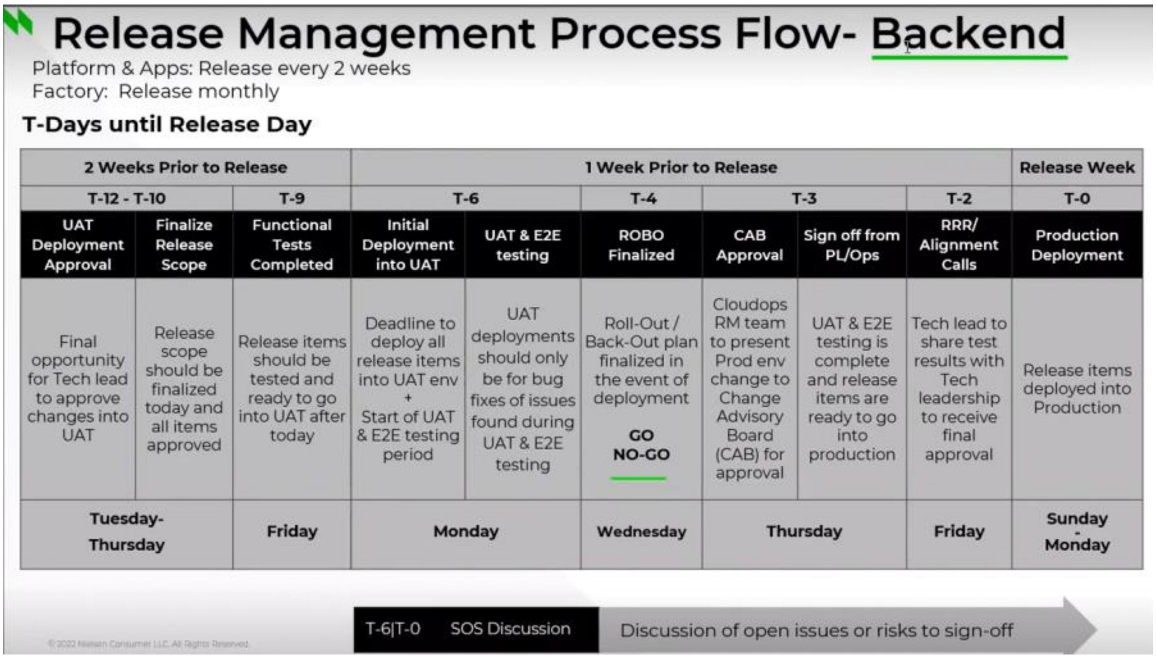
Release Management Process Flow- Frontend

Platform & Apps: Release every 2 weeks
Factory: Release monthly

T-Days until Release Day

2 Weeks Prior to Release			1 Week Prior to Release		Release Week				
T-15		T-14	T-11	T-11 – T4	T-4	T-2		T-1	T-0
UAT Deployment Approval	**Finalize Release Scope**	**Functional Tests Completed**	**Initial deployment into UAT**	**UAT & E2E testing**	**Sign off from PL/Ops**	**ROBO Finalized**	**RRR/ Alignment Calls**	**CAB Approval**	**Production Deployment**
Final opportunity for Tech lead to approve changes into UAT	Release scope should be finalized today and all items approved	Release items should be tested and ready to go into UAT after today	Deadline to deploy all release items into UAT env + Start of UAT & E2E testing period	UAT deployments should only be for bug fixes of issues found during UAT & E2E testing **GO** **NO-GO**	UAT & E2E testing is complete and release items are ready to go into production	Roll-Out / Back-Out plan finalized in the event of deployment	Tech lead to share test results with Tech leadership to receive final approval	Cloudops RM teamto present Prod env change to Change Advisory Board (CAB) for approval	Release items deployed into Production
Thursday		**Friday**	**Monday**	**Monday - Next Monday**		**Wednesday**		**Thursday**	**Friday**

T-11|T-0 SOS Discussion — Discussion of open issues or risks to sign-off



Release Management Process Flow- Backend

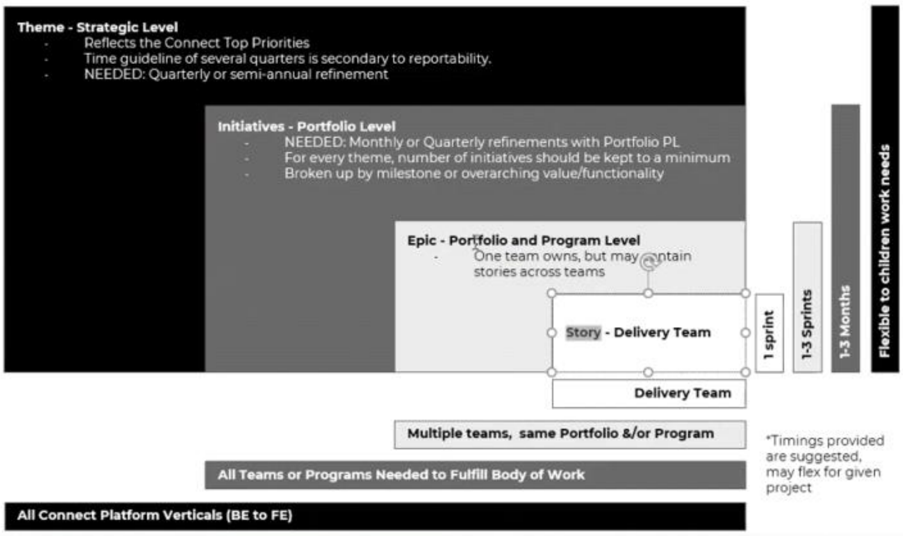
Platform & Apps: Release every 2 weeks
Factory: Release monthly

T-Days until Release Day

2 Weeks Prior to Release			1 Week Prior to Release						Release Week
T-12 - T-10		T-9	T-6		T-4	T-3		T-2	T-0
UAT Deployment Approval	**Finalize Release Scope**	**Functional Tests Completed**	**Initial Deployment into UAT**	**UAT & E2E testing**	**ROBO Finalized**	**CAB Approval**	**Sign off from PL/Ops**	**RRR/ Alignment Calls**	**Production Deployment**
Final opportunity for Tech lead to approve changes into UAT	Release scope should be finalized today and all items approved	Release items should be tested and ready to go into UAT after today	Deadline to deploy all release items into UAT env + Start of UAT & E2E testing period	UAT deployments should only be for bug fixes of issues found during UAT & E2E testing	Roll-Out / Back-Out plan finalized in the event of deployment **GO NO-GO**	Cloudops RM team to present Prod env change to Change Advisory Board (CAB) for approval	UAT & E2E testing is complete and release items are ready to go into production	Tech lead to share test results with Tech leadership to receive final approval	Release items deployed into Production
Tuesday-Thursday		**Friday**	**Monday**		**Wednesday**	**Thursday**		**Friday**	**Sunday - Monday**

T-6|T-0 SOS Discussion — Discussion of open issues or risks to sign-off

Issue Standards



Theme - Strategic Level
- Reflects the Connect Top Priorities
- Time guideline of several quarters is secondary to reportability.
- NEEDED: Quarterly or semi-annual refinement

Initiatives - Portfolio Level
- NEEDED: Monthly or Quarterly refinements with Portfolio PL
- For every theme, number of initiatives should be kept to a minimum
- Broken up by milestone or overarching value/functionality

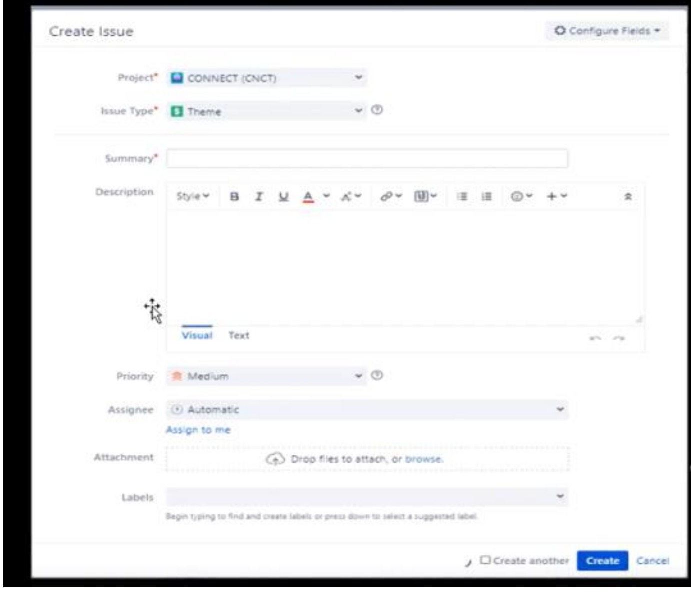
Epic - Portfolio and Program Level
- One team owns, but may contain stories across teams

Story - Delivery Team

1 sprint
1-3 Sprints
1-3 Months
Flexible to children work needs
Delivery Team
Multiple teams, same Portfolio &/or Program
All Teams or Programs Needed to Fulfill Body of Work
All Connect Platform Verticals (BE to FE)

*Timings provided are suggested, may flex for given project

Themes (Created by Senior Leadership)



Create Issue

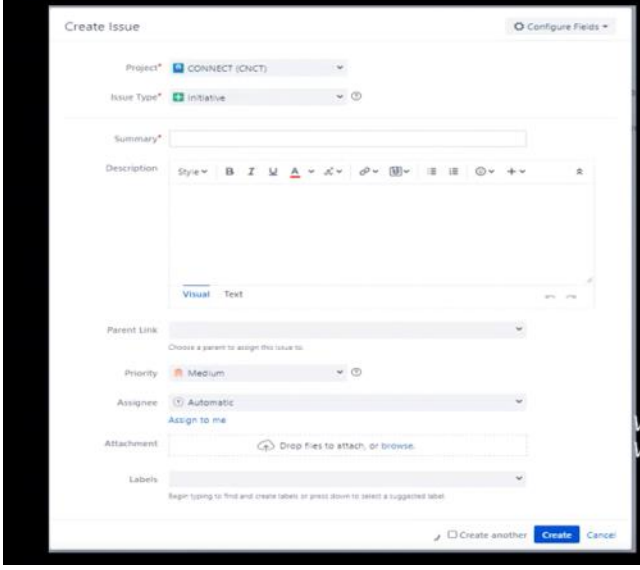
Project* CONNECT (CNCT)
Issue Type* Theme
Summary*
Description
Priority Medium
Assignee Automatic
Attachment Drop files to attach, or browse.
Labels
Begin typing to find and create labels or press down to select a suggested label.

- **Project** - CONNECT
- **Issue Type** - Theme
- **Summary** - provide business/technical description of Theme
- **Description** - brief description of business/technical objective
- **Priority** - select from pulldown menu
- **Assignee** - initially Senior Product Leadership and eventually PL assigned
- **Attachment** - include related documents for the theme such as Initiative Brief and Requirement Documents
- **Labels** - include 'Portfolio_Reportable' if this is to be included on the Leadership Report out board
- Other fields are optional and generally not used

Workflow: (Discovery, Design, Withdrawn, Done)

- Fields above remain the same through the workflow

Initiatives (Created by Product or Technical Leadership)



Project* CONNECT (CNCT)

Issue Type* Initiative

Summary*

Description

Parent Link

Choose a parent to assign this issue to.

Priority Medium

Assignee Automatic

Assign to me

Attachment Drop files to attach, or browse.

Labels

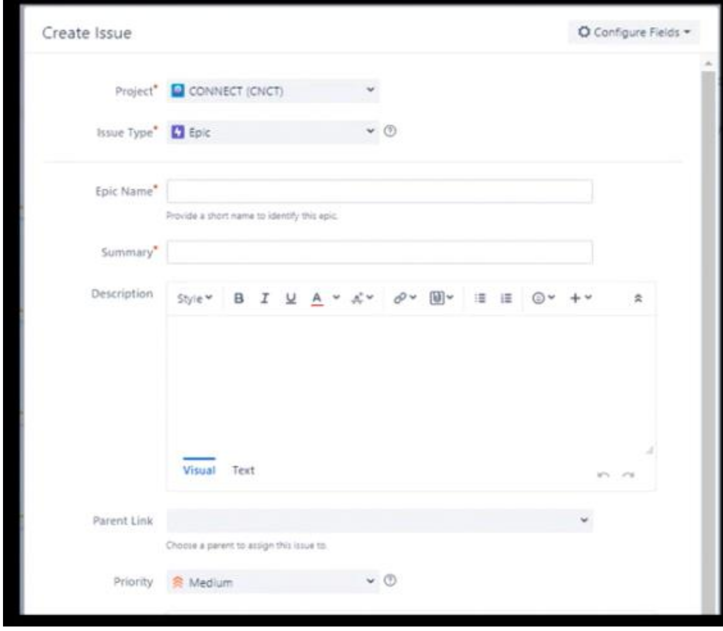
Begin typing to find and create labels or press down to select a suggested label.

- **Project** - CONNECT
- **Issue Type** - Initiative
- **Summary** - provide business or technical description of Initiative
- **Description** - brief description of business or technical objective
- **Parent Link** - select Theme
- **Priority** - select from pulldown menu
- **Assignee** - generally PL
- **Attachment** - include related documents for the theme
- **Labels** - include 'Portfolio_Reportable' if this is to be included on the Leadership Report out board
- Other fields are optional and generally not used

Workflow: (Portfolio Intake, Investment Solution, Withdrawn)

- Fields above remain the same through the workflow

Epics Part 1 (Created by PL, Tech, Ops or PMO)



Project* CONNECT (CNCT)

Issue Type* Epic

Epic Name*

Provide a short name to identify this epic.

Summary*

Description

Parent Link

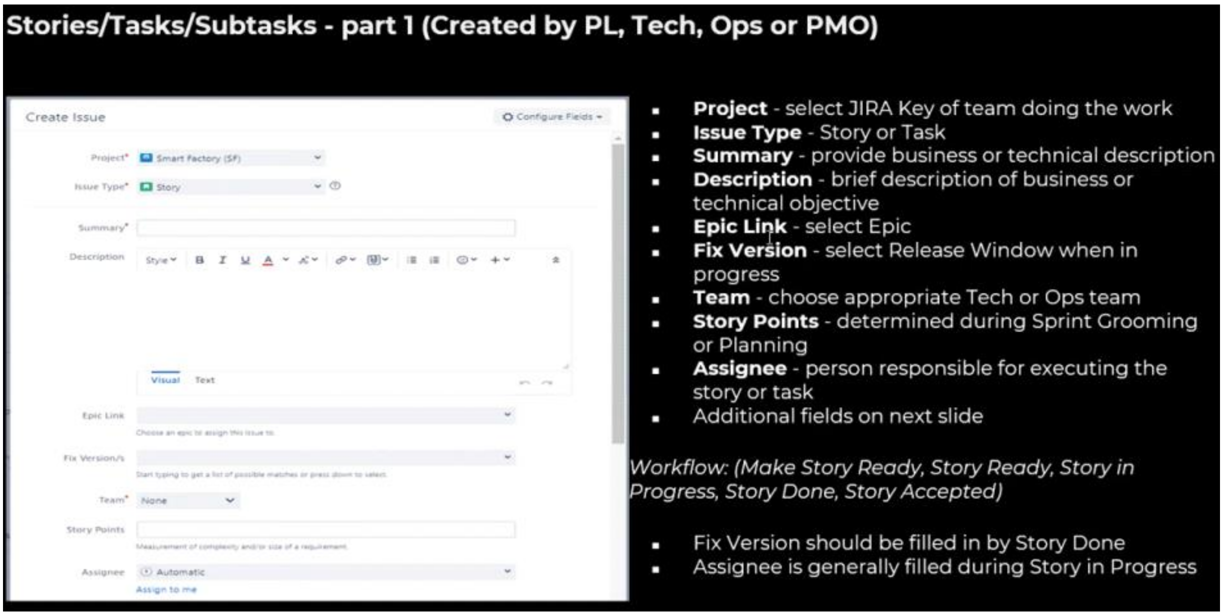
Choose a parent to assign this issue to.

Priority Medium

- **Project** - based on team doing the work
- **Issue Type** - Epic
- **Epic Name** - provide business or technical title
- **Summary** - provide business or technical description
- **Description** - brief description of business or technical objective
- **Parent Link** - select Initiative
- **Priority** - select from pulldown menu
- See next slide for other fields

Workflow: (Epic Ready, Epic In Progress, Epic Validation, Withdrawn)

- Fields on this slide do not change during the workflow



Stories/Tasks/Subtasks - part 1 (Created by PL, Tech, Ops or PMO)

Create Issue

Project*: Smart Factory (SF)

Issue Type*: Story

Summary*:

Description:

Epic Link:

Choose an epic to assign this issue to.

Fix Version/s:

Start typing to get a list of possible matches or press down to select.

Team*: None

Story Points:

Measurement of complexity and/or size of a requirement.

Assignee: Automatic

- **Project** - select JIRA Key of team doing the work
- **Issue Type** - Story or Task
- **Summary** - provide business or technical description
- **Description** - brief description of business or technical objective
- **Epic Link** - select Epic
- **Fix Version** - select Release Window when in progress
- **Team** - choose appropriate Tech or Ops team
- **Story Points** - determined during Sprint Grooming or Planning
- **Assignee** - person responsible for executing the story or task
- Additional fields on next slide

Workflow: (Make Story Ready, Story Ready, Story in Progress, Story Done, Story Accepted)

- Fix Version should be filled in by Story Done
- Assignee is generally filled during Story in Progress

SAFe (SAFe 5 for Lean Enterprises (scaledagileframework.com)**)**

Agile

Monday, April 4, 2022 10:16 PM