

Lesson 5

Exploring, Executing and Releasing Value

- 1. Introducing the Scaled Agile Framework
- 2. Embracing a Lean-Agile Mindset
- 3. Understanding SAFe Principles
- 4. Experiencing PI Planning
- 5. Exploring, Executing, and Releasing Value
- 6. Leading the Lean-Agile Enterprise
- 7. Empowering a Lean Portfolio
- 8. Building Large Solutions

SAFe® Course Attending this course gives students access to the SAFe® Lean-Agile Leader exam and related preparation materials.

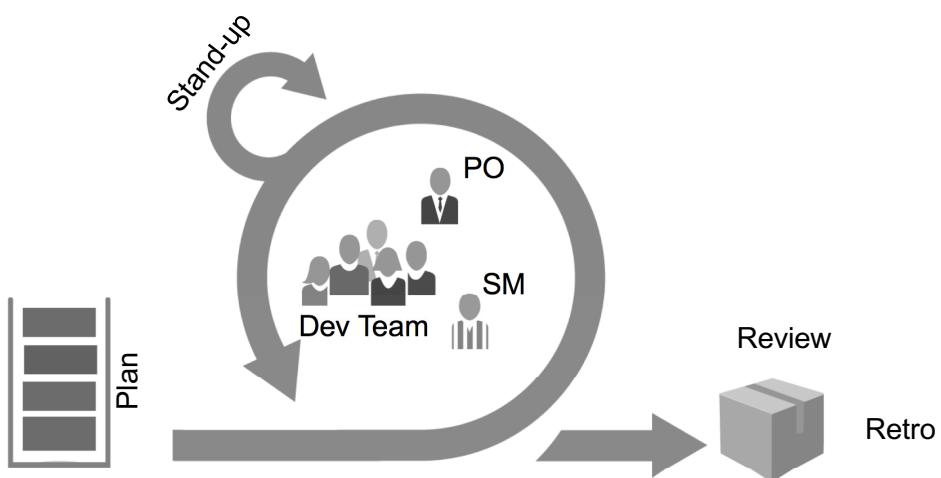
Learning objectives

- 5.1 Continuously deliver value with Agile Release Trains
- 5.2 Continuously explore customer needs
- 5.3 Continuously integrate
- 5.4 Continuously deploy with DevOps
- 5.5 Release on demand
- 5.6 Relentlessly improve results

5.1 Continuously deliver value with Agile Release Trains

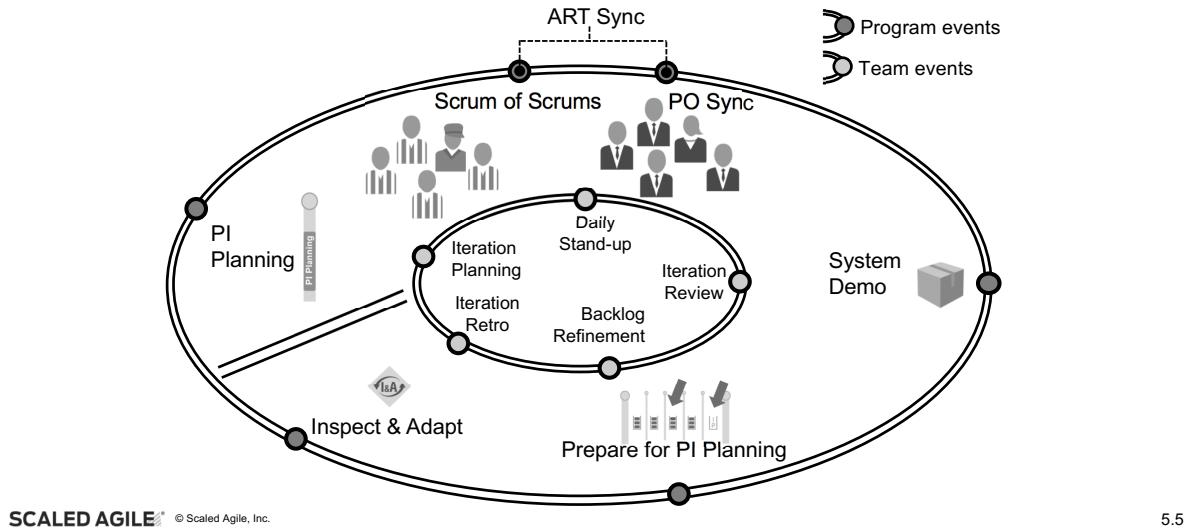
Teams execute iterations with Scrum

Scrum is built on transparency, inspection, and adaptation.



Program events drive the train

Program events create a closed loop system to keep the train on the tracks.



ART Sync is used to coordinate progress

Programs coordinate dependencies through sync meetings.



Scrum of Scrums

- ▶ Visibility into progress and impediments
- ▶ Facilitated by RTE
- ▶ Participants: Scrum Masters, other select team members, SMEs if necessary
- ▶ Weekly or more frequently, 30 – 60 minutes
- ▶ Timeboxed, and followed by a “meet after”

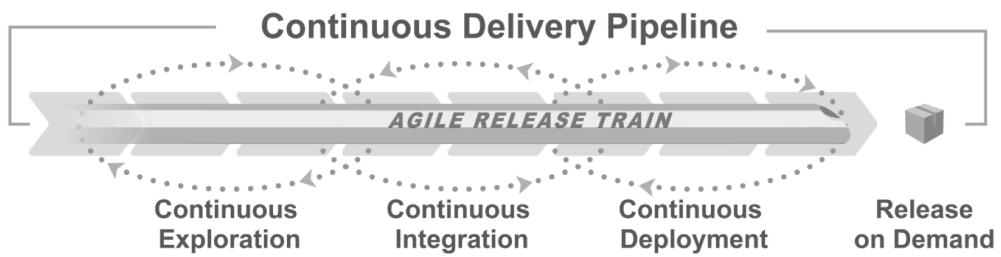
ART Sync



PO Sync

- ▶ Visibility into progress, scope, and priority adjustments
- ▶ Facilitated by RTE or PM
- ▶ Participants: PMs, POs, other stakeholders, and SMEs as necessary
- ▶ Weekly or more frequently, 30 – 60 minutes
- ▶ Timeboxed, and followed by a “meet after”

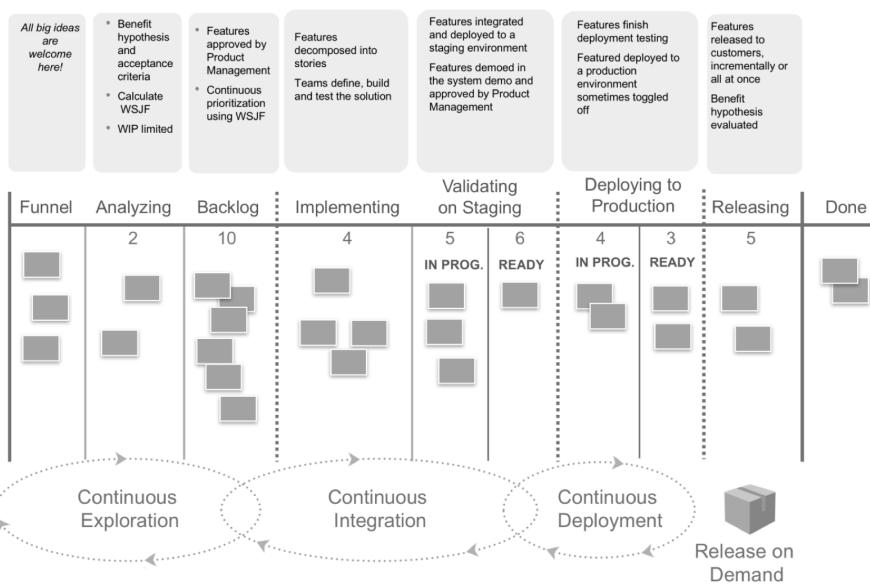
ARTs continuously deliver value



SCALED AGILE® © Scaled Agile, Inc.

5.7

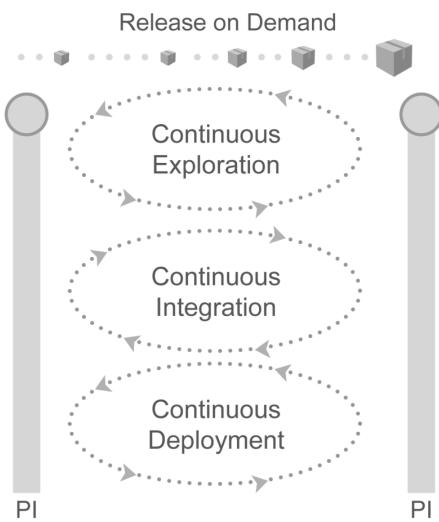
Manage continuous delivery with the Program Kanban



SCALED AGILE® © Scaled Agile, Inc.

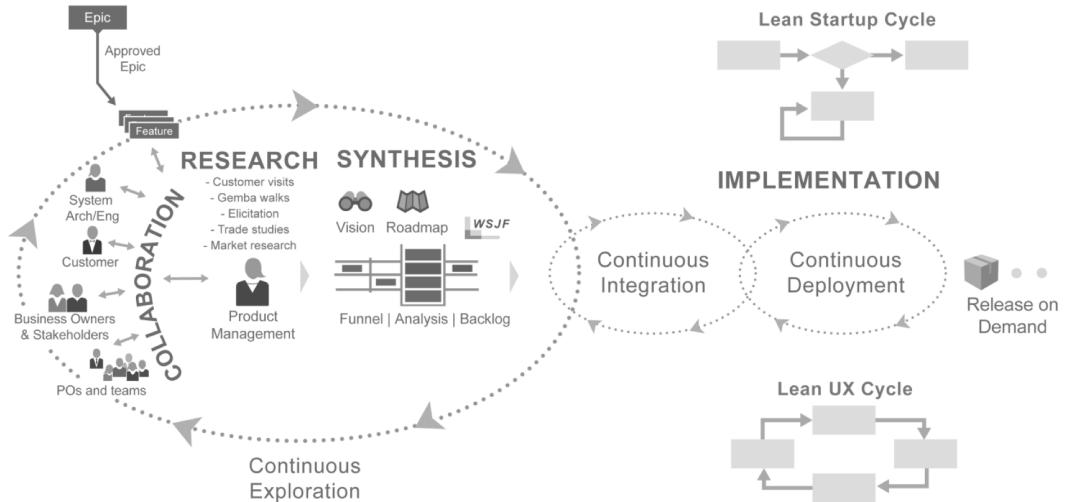
5.8

Continuous delivery happens all the time



5.2 Continuously explore customer needs

Continuous Exploration



SCALED AGILE® © Scaled Agile, Inc.

5.11

Exercise: Who is involved in collaboration?

At your table, discuss what collaboration and research the PM would engage in. Who would be the stakeholders?

Collaboration

Research

PREPARE SHARE

7 min 3 min

SCALED AGILE® © Scaled Agile, Inc.

5.12

Vision inspires action

It provides a longer term context:

- ▶ How will our future solution solve the larger customer problems?
- ▶ How will it differentiate us?
- ▶ What is the future context that our solutions will operate?
- ▶ What is our current business context, and how must we evolve to meet this future state?



Example: John Deere Vision
<http://tinyurl.com/p5uloc5>
2:45

Vision: A postcard from the future



- Aspirational, yet realistic and achievable
- Motivational enough to engage others on the journey

Result: The teams start thinking about how to apply their strengths in order to get there.

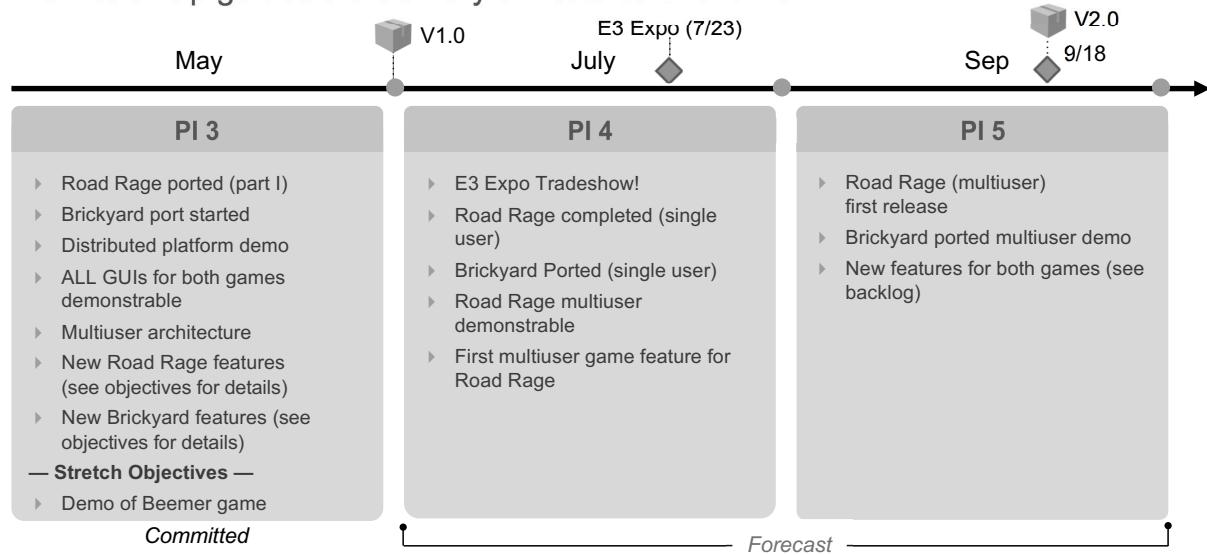
Switch: How to Change Things When Change Is Hard,
Heath and Heath, Broadway Books, 2010

5.13

SCALED AGILE® © Scaled Agile, Inc.

Roadmap creates the outline

The Roadmap guides the delivery of Features over time.



SCALED AGILE® © Scaled Agile, Inc.

5.14

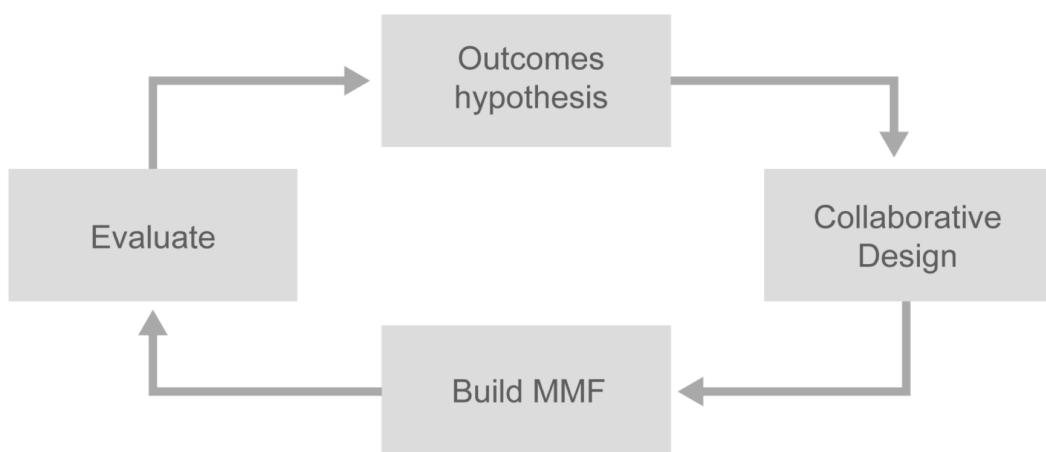
Exercise: Is a Roadmap a queue?

1. Find someone you have not paired with yet and answer the following questions.
2. If the Roadmap on the prior slide was full of features committed for three PIs, how long would it take to deliver a new, unanticipated feature?

3. When, if ever, is a Roadmap a queue?



Elaborate features with Lean UX



Features have benefit hypothesis and acceptance criteria

- ▶ “Feature” is an industry-standard term familiar to marketing and Product Management
- ▶ Benefit hypothesis justify Feature implementation cost, and provides business perspective when making scope decisions
- ▶ Acceptance criteria is typically defined during Program Backlog refinement
- ▶ Reflect functional and Nonfunctional Requirements
- ▶ Fits in one PI

SSO example:

Multi-factor authentication

Benefit hypothesis

Enhance user security via both password and a device.

Acceptance criteria

1. USB tokens as a first layer
2. Password authentication second layer
3. Multiple tokens on a single device
4. User activity log reflecting both authentication factors

Exercise: Features in your context

Identify three features from your business context or personal backlog items, and write them in prioritized order.

Feature	Benefit hypothesis

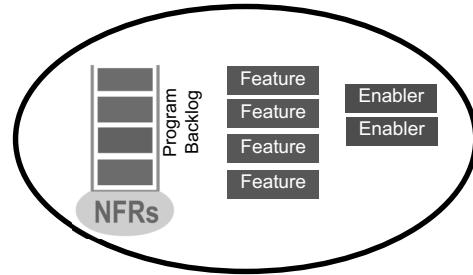


Prioritize Features for optimal ROI

In a flow system, job sequencing is the key to economic outcomes.

To prioritize based on lean economics,
we need to know two things:

1. What is the Cost of Delay (CoD) in delivering value?
2. What is the cost to implement the valuable thing?



If you only quantify one thing, quantify the Cost of Delay.

—Donald G. Reinertsen,
Principles of Product Development Flow E3



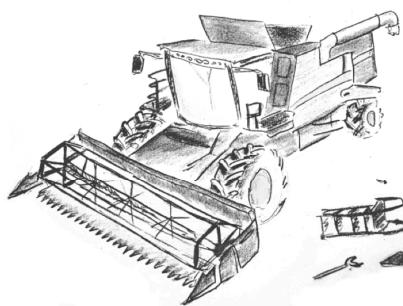
WSJF done right
<https://tinyurl.com/kpmjnuk>
20:34

SCALED AGILE® © Scaled Agile, Inc.

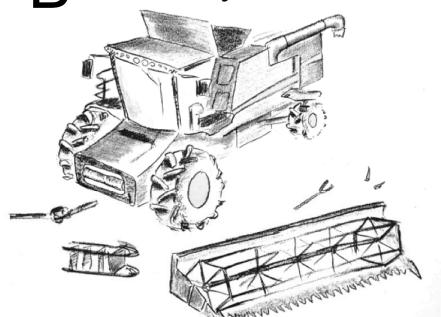
5.19

Example with equal CoD: which job first?

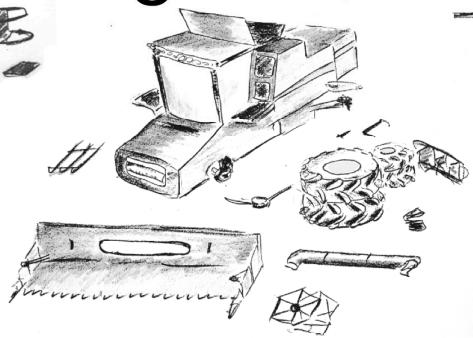
A \$\$, 1 day



B \$\$, 3 days



C \$\$, 10 days

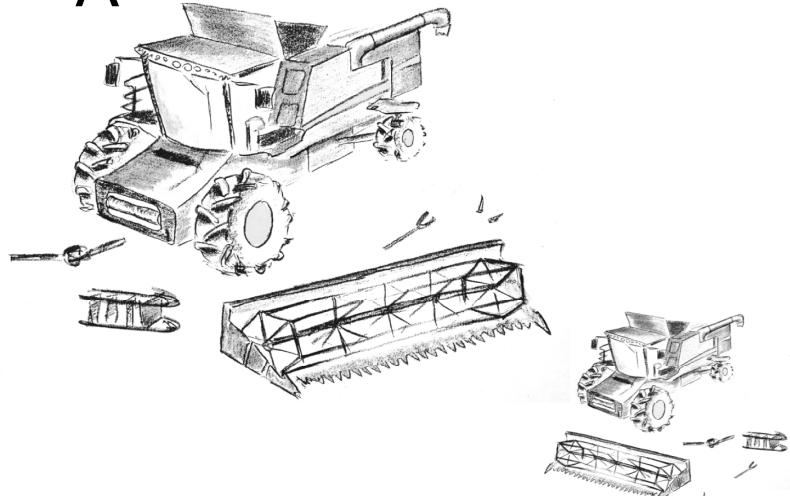


SCALED AGILE® © Scaled Agile, Inc.

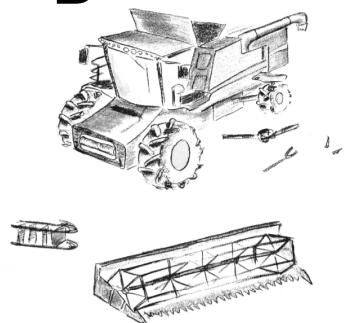
5.20

Example with equal Duration: which job first?

A \$\$\$, 3 days



B \$\$, 3 days



C \$, 3 days

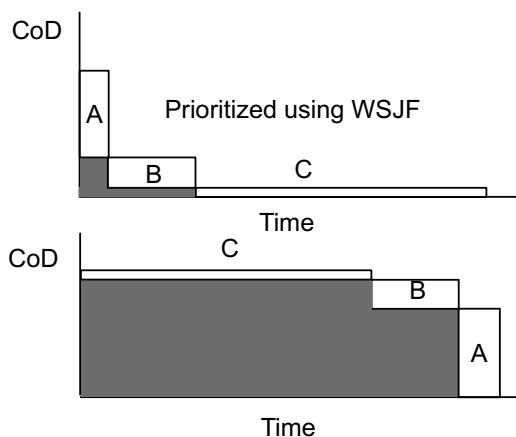


SCALED AGILE® © Scaled Agile, Inc.

5.21

General case: any CoD and Duration

In the general case, give preference to jobs with *shorter Duration* and *higher CoD*, using *Weighted Shortest Job First* (WSJF):



$$\text{WSJF} = \frac{\text{CoD}}{\text{Duration}}$$

Feature	Duration	CoD	WSJF
A	1	10	10
B	3	3	1
C	10	1	0.1

— Dark area: total Cost of Delay

Adapted from *The Principles of Product Development Flow*, Donald G. Reinertsen

5.22

SCALED AGILE® © Scaled Agile, Inc.

Components of Cost of Delay

User and business value	Relative value to the customer or business <ul style="list-style-type: none">▶ They prefer this over that▶ Revenue impact?▶ Potential penalty or other negative impact?
Time criticality	How User/Business Value decays over time <ul style="list-style-type: none">▶ Is there a fixed deadline?▶ Will they wait for us or move to another solution?▶ What is the current effect on customer satisfaction?
Risk Reduction & Opportunity Enablement (RR & OE)	What else does this do for our business <ul style="list-style-type: none">▶ Reduce the risk of this or future delivery?▶ Is there value in the information we will receive?▶ Enable new business opportunities?

Calculate WSJF with relative estimating

- ▶ In order to calculate WSJF, teams need to estimate Cost of Delay and duration
- ▶ For duration, use job size as a quick proxy for duration
- ▶ Relative estimating is a quick technique to estimate job size and relative value
- ▶ WSJF stakeholders: Business Owners, Product Managers, Product Owners, System Architects

$$\text{WSJF} = \frac{\text{CoD}}{\text{Job size}} = \frac{\text{User-business value} + \text{Time criticality} + \text{RR | OE value}}{\text{Job size}}$$

Exercise: WSJF prioritization

Prioritize your three features using WSJF.

Feature	User-business value	Time criticality	RR OE value	CoD	Job size	WSJF
	+	+	=	÷	=	
	+	+	=	÷	=	
	+	+	=	÷	=	

Scale for each parameter: 1, 2, 3, 5, 8, 13, 20

Note: Do one column at a time, start by picking the smallest item and giving it a "1."

There must be at least one "1" in each column!



Discussion: Duration

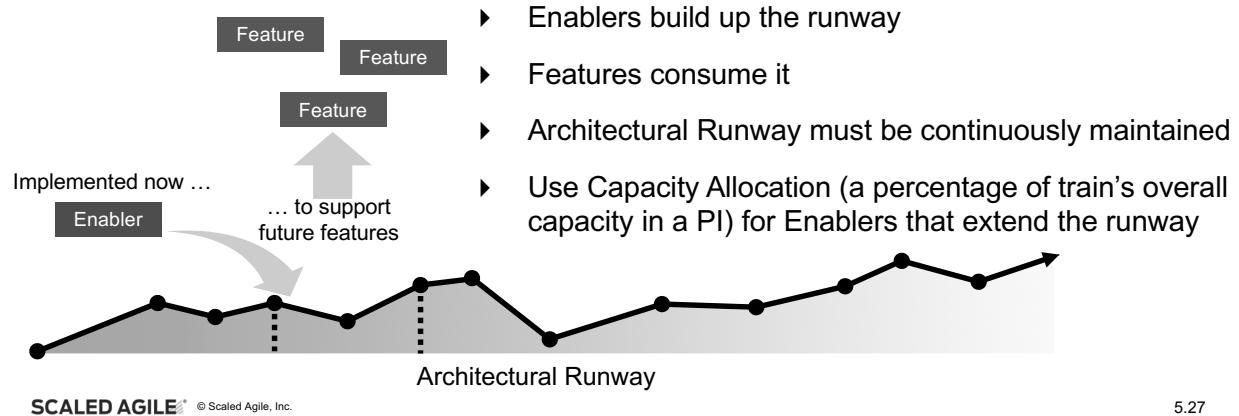
- ▶ Question: Is job size always a good proxy for duration?
- ▶ When might that NOT be the case?
- ▶ How would you adjust based on that case?



Architectural Runway

Architectural Runway is existing code, hardware components, etc. that technically enable near-term business features.

Example: A new, fuzzy search algorithm will enable a variety of future features that can accept potentially erroneous user input



Without Architectural Runway ...

- Architecture deteriorates quickly under the pressure of "now"
- Velocity peaks for a while, then falls off
- Releases are late and spasmodic; quality varies dramatically
- Solution robustness, maintainability, and quality decay rapidly
- Development pace is unsustainable

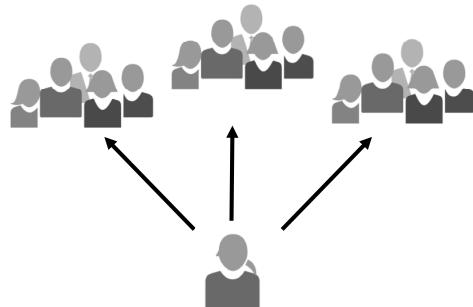


Emergent design and intentional architecture

Every team deserves to see the bigger picture. Every team is empowered to design their part.

Emergent design – teams grow the system design as user stories require

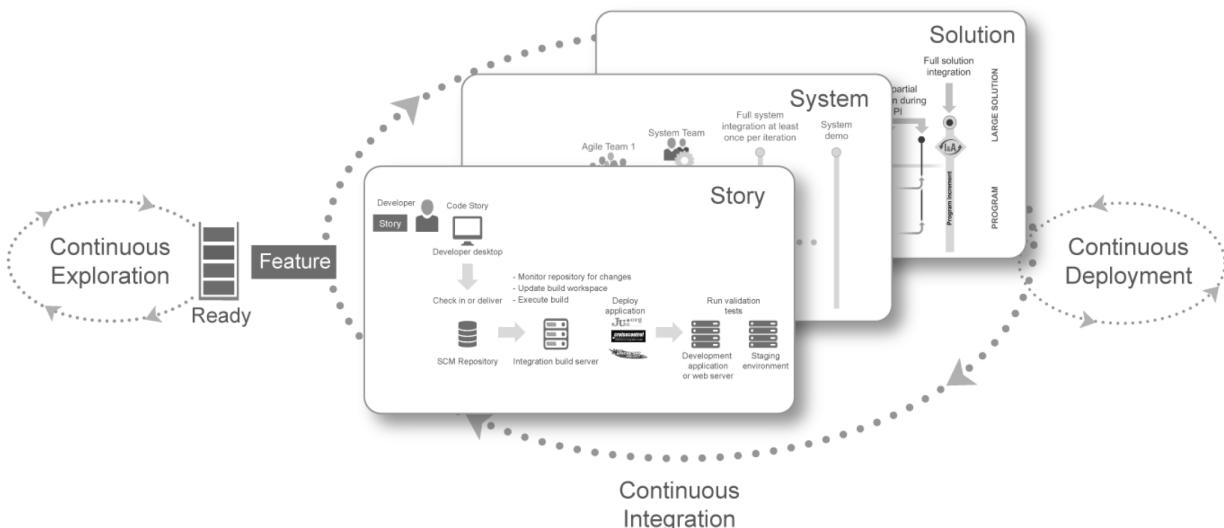
Intentional architecture – fosters team alignment and defines Architectural Runway



A balance between emergent design and intentional architecture is required for speed of development and maintainability.

5.3 Continuously integrate

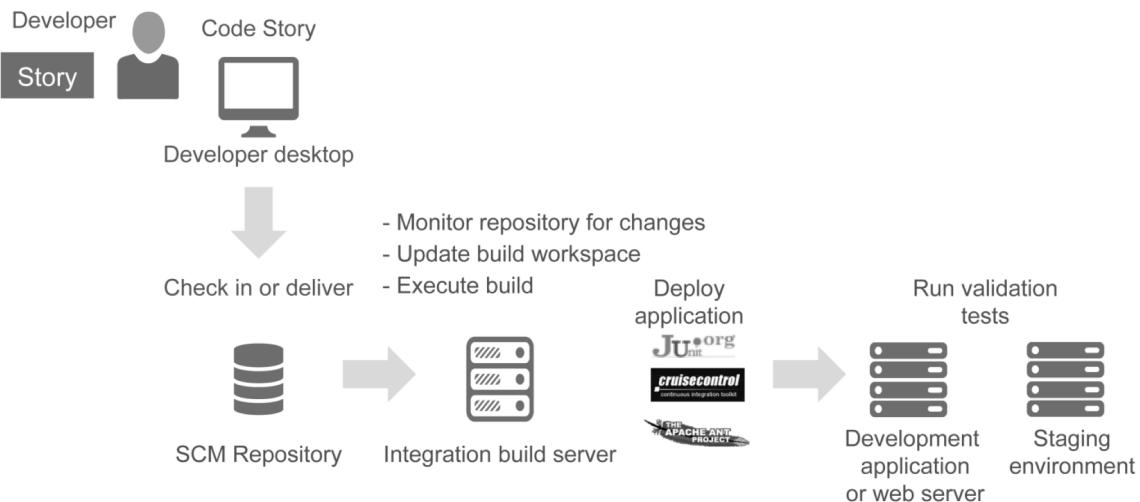
Continuously integrate stories and features



SCALED AGILE® © Scaled Agile, Inc.

5.31

Continuous story integration



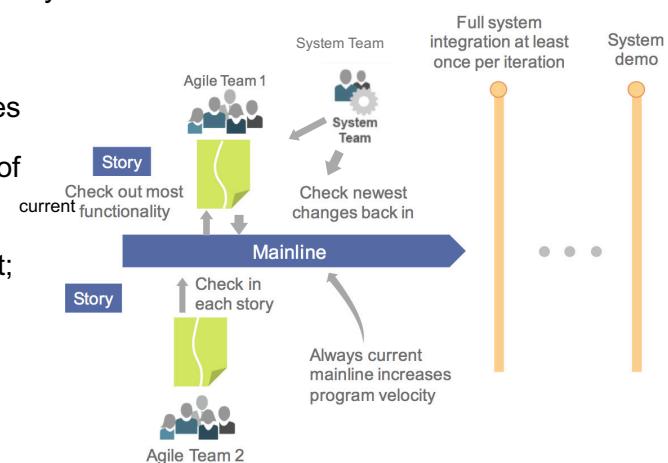
SCALED AGILE® © Scaled Agile, Inc.

5.32

Continuous system integration

Teams continuously integrate assets (leaving as little as possible to the System Team).

- ▶ Integrate every vertical slice of a user story
- ▶ Avoid physical branching for software
- ▶ Frequently integrate hardware branches
- ▶ Use development by intention in case of inter-team dependencies
 - Define interfaces and integrate first; then add functionality



Exercise: Continuous Integration

At your table, discuss challenges you've seen to doing continuous integration at the team and system levels. Identify strategies to solve these challenges.

Challenges	Solution strategies



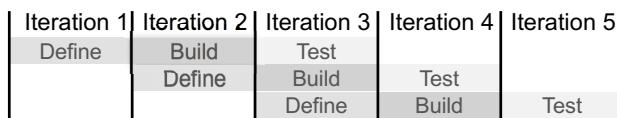
Exercise: Continuous Integration teach-back

- ▶ Find someone from another table who you haven't paired with before.
- ▶ Explain the challenges your table identified and your strategies to solve them. See if your partner has other ideas on how to solve them.
- ▶ After five minutes, change roles.



Execution: Don't waterfall Iterations

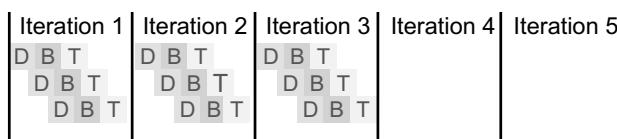
This is an inter-Iteration waterfall:



This is an intra-Iteration waterfall:



These are cross-functional Iterations:



Build quality in

“You can’t scale crappy code” (or hardware, or anything else)

Building quality in:

- ▶ Ensures that every increment of the solution reflects quality standards
- ▶ Is required for high, sustainable development velocity
- ▶ Software quality practices (most inspired by XP) include Continuous Integration, Test-First, Refactoring, Pair-Work, Collective Ownership and more
- ▶ Hardware quality is supported by exploratory, early iterations, frequent system level integration, design verification, MBSE and Set-Based Design

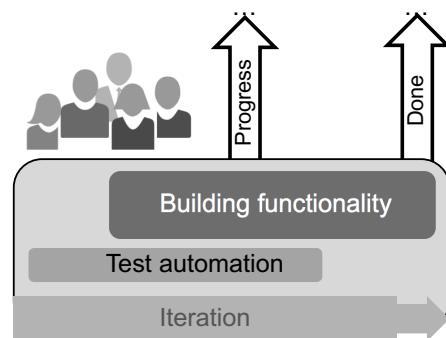


Test first: Automate now!

Else, velocity is bottlenecked, quality is speculative, and scaling is impossible.

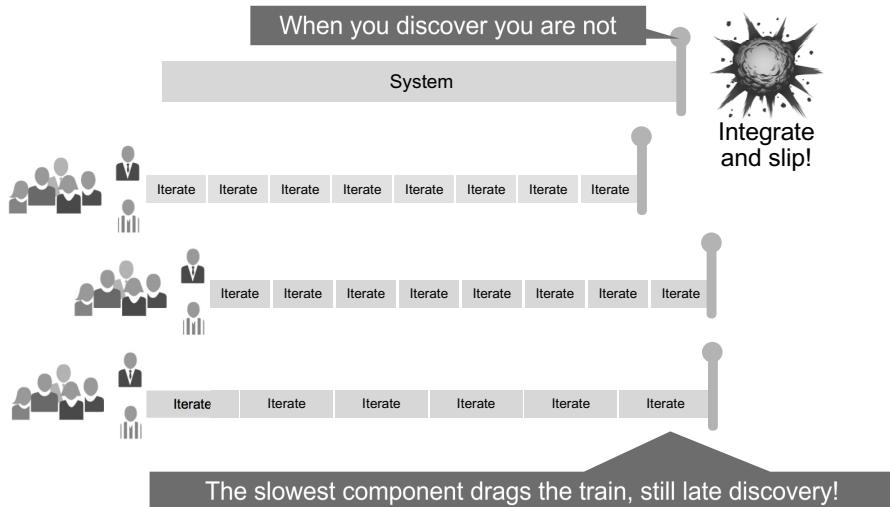
- ▶ Automated tests are implemented in the same iteration as the functionality
- ▶ The team that builds functionality also automates the tests
- ▶ Create an isolated automated test environment
- ▶ Actively maintain test data under version control
- ▶ Passing vs. not-yet-passing and broken automated tests are the *real* iteration progress indicator

✓ Test 1	✓ Test 1
● Test 2	✓ Test 2
✓ Test 3	✓ Test 3
● Test 4	✓ Test 4
✗ Test 5	✓ Test 5



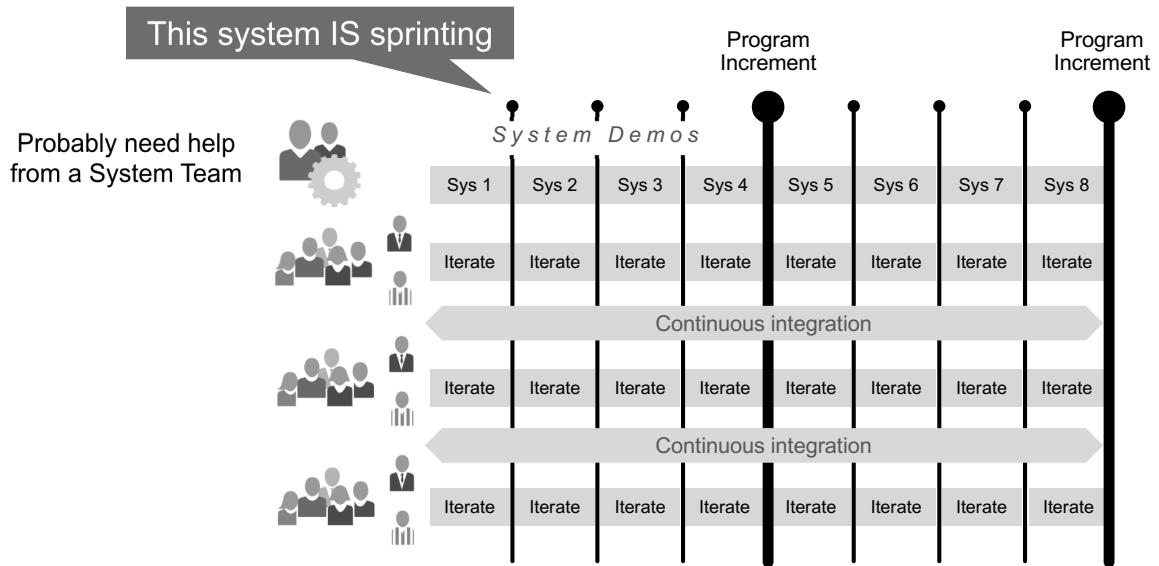
Cadence without synchronization is not enough

Time spent thinking you are on track

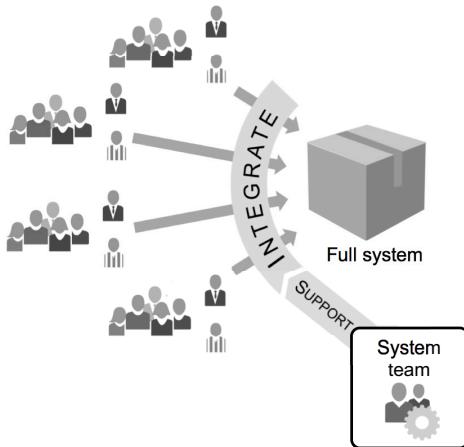


Synchronize to assure delivery

This system IS sprinting



Demo the full system increment every two weeks



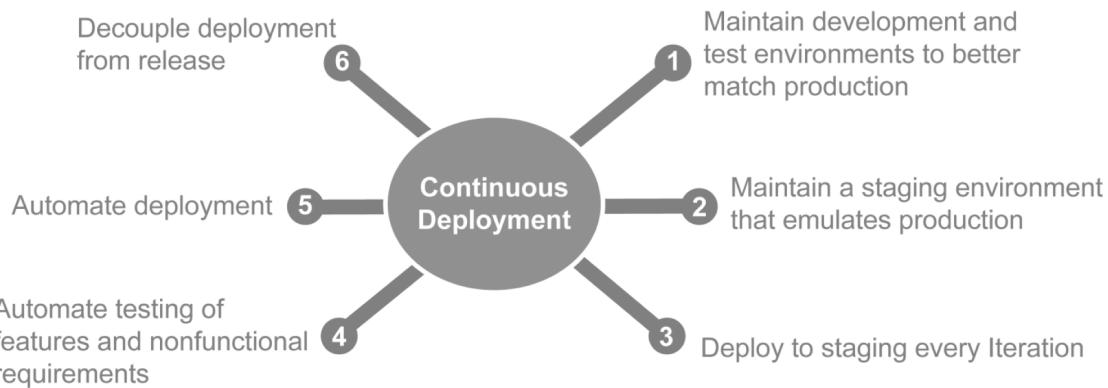
- ▶ Features are functionally complete or “toggled” so as not to disrupt demonstrable functionality
- ▶ New Features work together, and with existing functionality
- ▶ Happens after the teams’ demo (may lag by as much as one iteration, maximum)
- ▶ Demo from a staging environment, resemble production as much as possible



5.4 Continuously deploy with DevOps

Six Recommended Practices for Continuous Deployment (CD)

CD is an important practice for every team member, the team, and the ART.



Exercise: DevOps myth or fact game

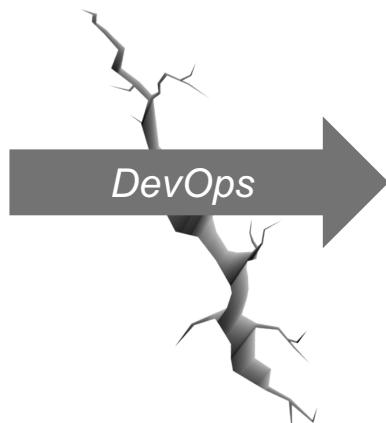
- ▶ Open the DevOps myth or fact envelope
- ▶ Sort the cards into two columns "Myth" and "Fact"
- ▶ Check yourself with the answer key



What is DevOps?

An agile approach to bridge the gap between development and operations to deliver value *faster and more reliable*.

Development:
▶ Create Change
▶ Add or Modify Features

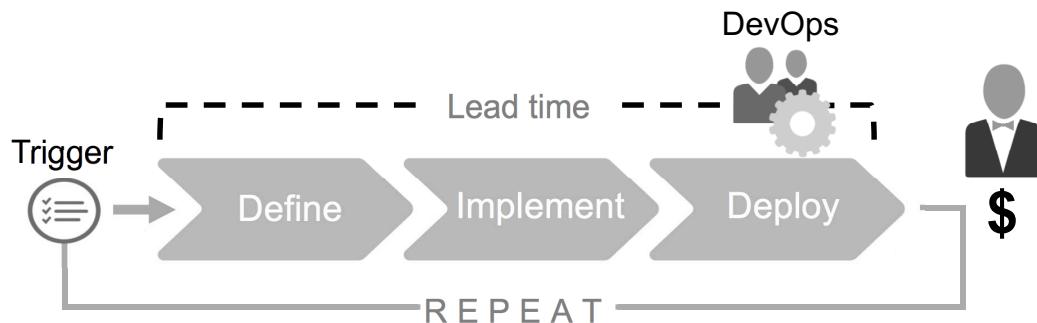


Operations:
▶ Create Stability
▶ Create or enhance services

DevOps is a capability of every Agile Release Train

DevOps is IN the Value Stream

Value occurs only when the end users are operating the Solution.



DevOps isn't optional. The only question is how efficient it is.

A CALMR approach to DevOps

- ▶ **C**ulture Establish a culture of shared responsibility for development, deployment and operations
- ▶ **A**utomation Automate the continuous delivery pipeline
- ▶ **L**ean flow Keep batch sizes small, limit WIP and provide extreme visibility
- ▶ **M**easurement Measure the flow through the pipeline. Implement application telemetry.
- ▶ **R**ecovery Architect and enable low risk releases. Establish fast recovery, fast reversion, and fast fix-forward.



Bing: Continuous Delivery
<https://youtu.be/3sFT7tgyEQk>
3:28

SCALED AGILE® © Scaled Agile, Inc.

5.47

Exercise: A CALMR poster

- ▶ Using the DevOps article in the appendix and the following five reference slides, build a poster that explains the CALMR approach to DevOps
- ▶ Identify the Lean-Agile leader's role in each concept
- ▶ Be prepared to share



SCALED AGILE® © Scaled Agile, Inc.

5.48

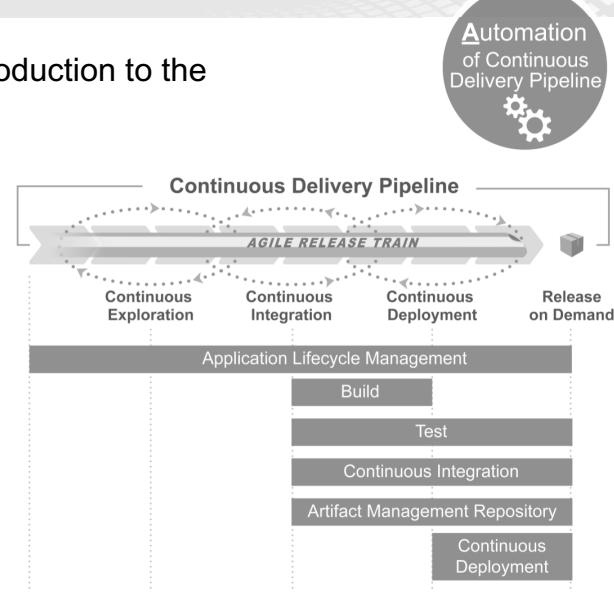
DevOps is a Cultural shift first of all

- ▶ A culture of shared responsibility for development and deployment
- ▶ DevOps requires a tolerance for failure and rapid recovery, and rewards risk taking
- ▶ Sharing discoveries, practices, tools, and learning across silos is encouraged

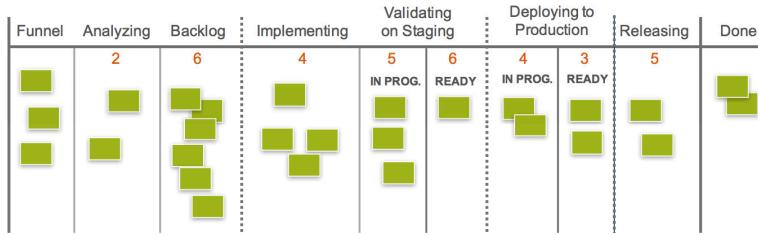


Automate the deployment process

1. Match development environments to production to the extent feasible
2. Maintain a staging environment that emulates production
3. Deploy a working system to staging every Iteration
4. Automate testing of Features and performance tests
5. Automate deployment
 - Everything under version control
 - Automatically build environments
 - Automate the actual deployment process

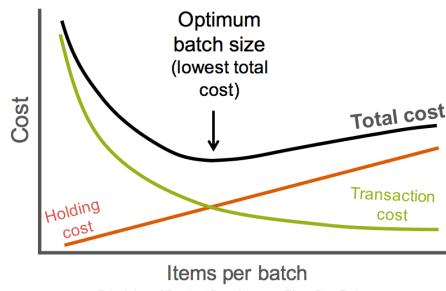


Focus on continuous Lean flow of value



Lean Flow
accelerates delivery

- ▶ Identify bottlenecks and balance the amount of WIP against the available development and operations
- ▶ Decrease the batch sizes of the work
- ▶ Manage and reduce, queue lengths



Measure everything

- ▶ Collect data on business, application, infrastructure and client layers
- ▶ Collect data about the deployment pipeline itself
- ▶ Store logs in ways that enable analysis
- ▶ Different telemetry for different stakeholders
- ▶ Broadcast measurements
- ▶ Overlay measurements with events (deploys, releases)
- ▶ Continuously improve telemetry during and after problem solving



Architect for release-ability and Recovery

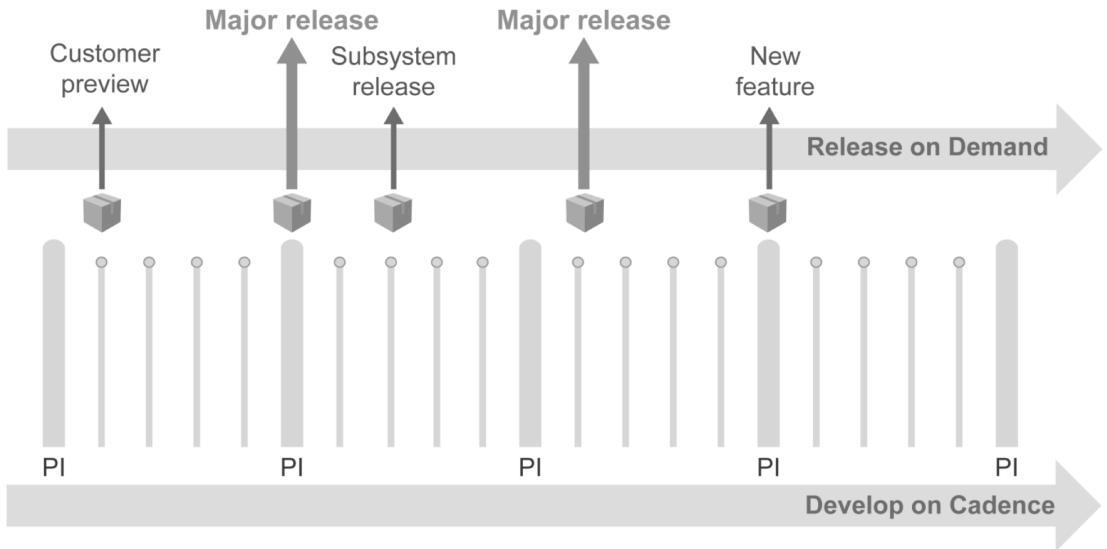
- ▶ Stop-the-line mentality
- ▶ Plan for and rehearse failures
- ▶ Build the environment for both roll back and fix forward
- ▶ Use tools such as:
 - Feature toggles
 - Dark launches
 - Chaos monkey
 - Canary Releases

Source: the DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations IT Revolution Press..Kim, Gene; Humble, Jez; Debois, Patrick; Willis, John



5.5 Release on demand

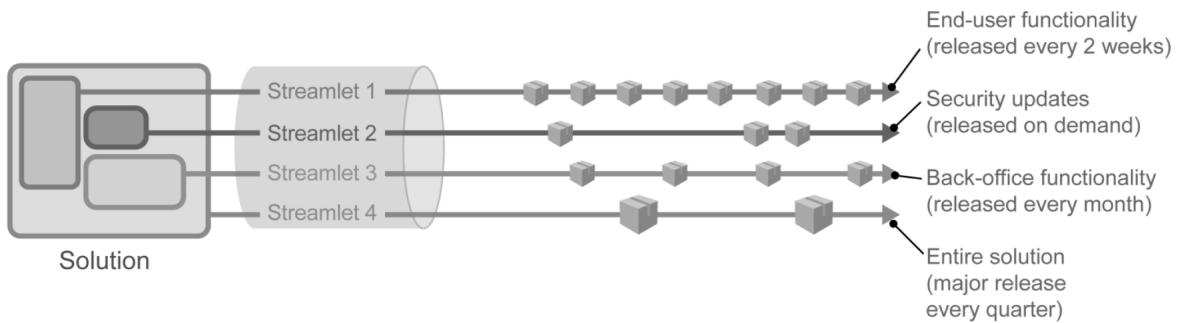
Decouple deployment from release



SCALED AGILE® © Scaled Agile, Inc.

5.55

Decouple release elements from the total solution



SCALED AGILE® © Scaled Agile, Inc.

5.56

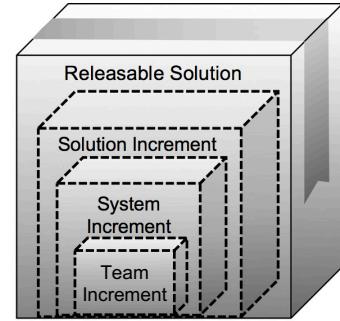
Releasing includes additional activities

System validation:

- ▶ User acceptance testing
- ▶ Final NFR testing
- ▶ Integration testing with other systems
- ▶ Regulatory standards and requirements

Documentation:

- ▶ Release communications
- ▶ End user documentation
- ▶ Bill of materials
- ▶ Training support personnel
- ▶ Installation/deployment instructions
- ▶ Legal, regulatory, other approvals
- ▶ etc. ...



Exercise: Continuous Delivery Pipeline

Implementing the Continuous Delivery Pipeline is not easy. Identify three minimum viable improvements you could implement at your organization.

Minimal viable improvements



5.6 Relentlessly improve results

Innovation and Planning Iteration

Facilitate reliability, Program Increment readiness, planning, and innovation

- ▶ Innovation: Opportunity for innovation, hackathons, and infrastructure improvements
- ▶ Planning: Provides for cadence-based planning
- ▶ Estimating guard band for cadence-based delivery



Provide sufficient capacity margin to enable cadence.

—Don Reinertsen, Principles of Product Development Flow

IP Iteration calendar

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
31	1	2	3	4	5	6
			Validation (if shipping)			
			Innovation / hackathon / spikes for next PI			
			PI Planning readiness			
7	8	9			12	13
	Continuing education			PI Planning		
	Inspect and Adapt workshop					

SCALED AGILE® © Scaled Agile, Inc.

5.61

Without the IP Iteration ...

- Lack of delivery capacity buffer impacts predictability
- Little innovation, tyranny of the urgent
- Technical debt grows uncontrollably
- People burn out
- No time for teams to plan together, demo together, and improve together



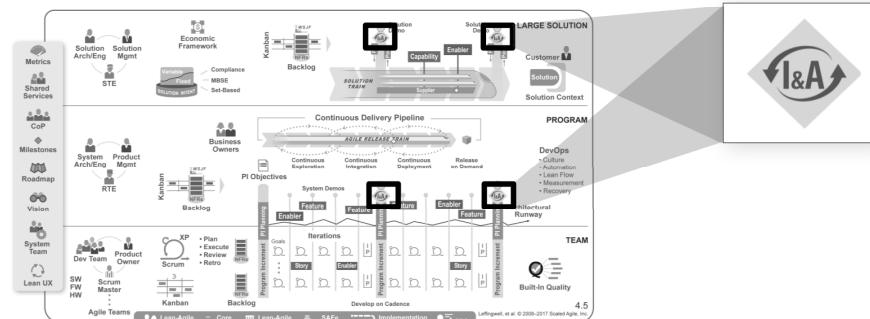
SCALED AGILE® © Scaled Agile, Inc.

5.62

Inspect and Adapt

Three parts:

1. The PI System Demo
 2. Quantitative measurement
 3. The problem-solving workshop
- ▶ Attendees: Teams and stakeholders
 - ▶ Timebox: 3 – 4 hours per PI



SCALED AGILE® © Scaled Agile, Inc.

5.63

PI System Demo

At the end of the PI, teams demonstrate the current state of the Solution to the appropriate stakeholders.

- ▶ Often led by Product Management, POs, and the System Team
- ▶ Attended by Business Owners, program stakeholders, Product Management, RTE, Scrum Masters, and teams



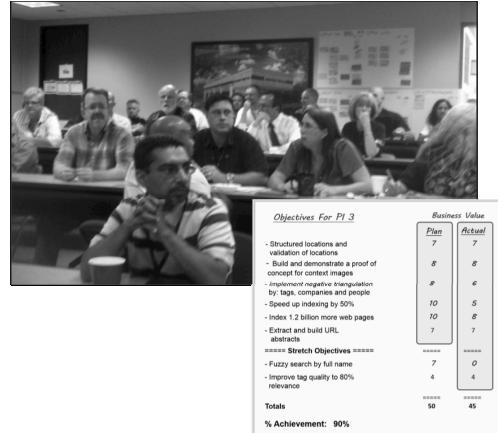
SCALED AGILE® © Scaled Agile, Inc.

5.64

Program performance reporting

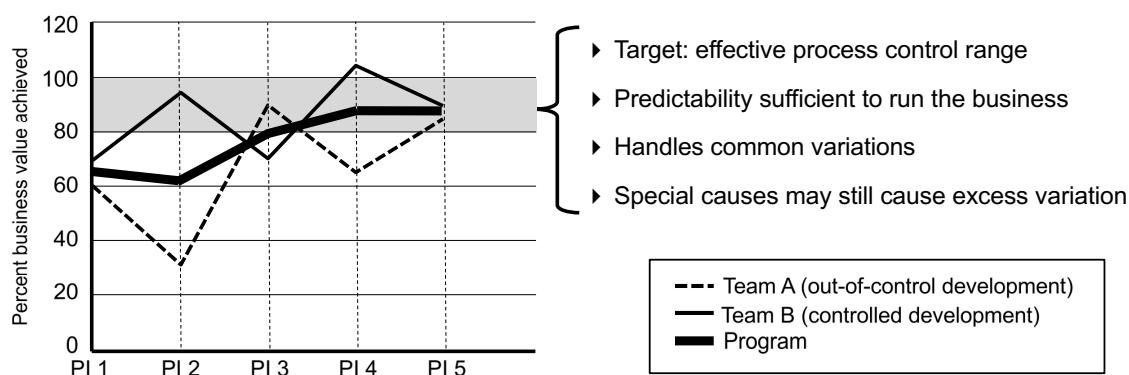
As part of the Solution Demo, teams compare planned vs. actual PI Objectives.

- ▶ Teams meet with their Business Owners to self-assess the business value they achieved for each objective
- ▶ Each team's planned vs. actual business value is then rolled up to the Program Level in the Program Predictability Measure



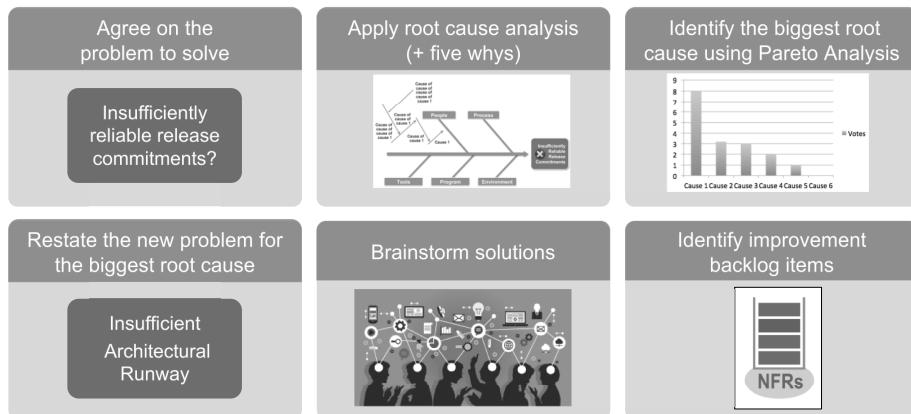
PI Predictability Measure

The PI Predictability Measure shows whether achievements fall into an acceptable process control band.



The problem-solving workshop

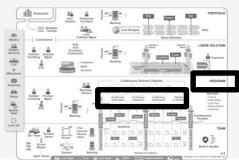
Teams conduct a short retrospective, then systematically address the larger impediments that are limiting velocity.



Lesson summary

In this lesson, you learned how to:

- ▶ Continuously deliver value
- ▶ Continuously explore customer need
- ▶ Develop the Vision, Roadmap and prioritize the Program Backlog
- ▶ Execute the Program Increment
- ▶ Continuously integrate
- ▶ Continuously deploy
- ▶ Release value on demand
- ▶ Improve program performance with Inspect and Adapt



Suggested Scaled Agile Framework reading: “Program Level”, “Continuous Exploration”, “Continuous Integration”, “Continuous Deployment”, and “Release on Demand” articles