

# SAFe® for Teams

Establishing Team Agility  
for Agile Release Trains

6.0.2

Workbook



# Welcome to the course!

# **Make the Most of**

# **Your Learning**



## **Access SAFe Studio**

Manage your member profile, access videos and training resources, join Communities of Practice, and more.



## **Prepare Yourself**

Access your learning plan featuring your digital workbook, study materials, and certification practice test



## **Become a Certified SAFe Professional**

Get certified to validate your knowledge, expand your professional capabilities, and open the door to new career opportunities.



## **Access SAFe Content and Tools**

Access professional development resources and toolkits.



## **Collaborate with Your Team**

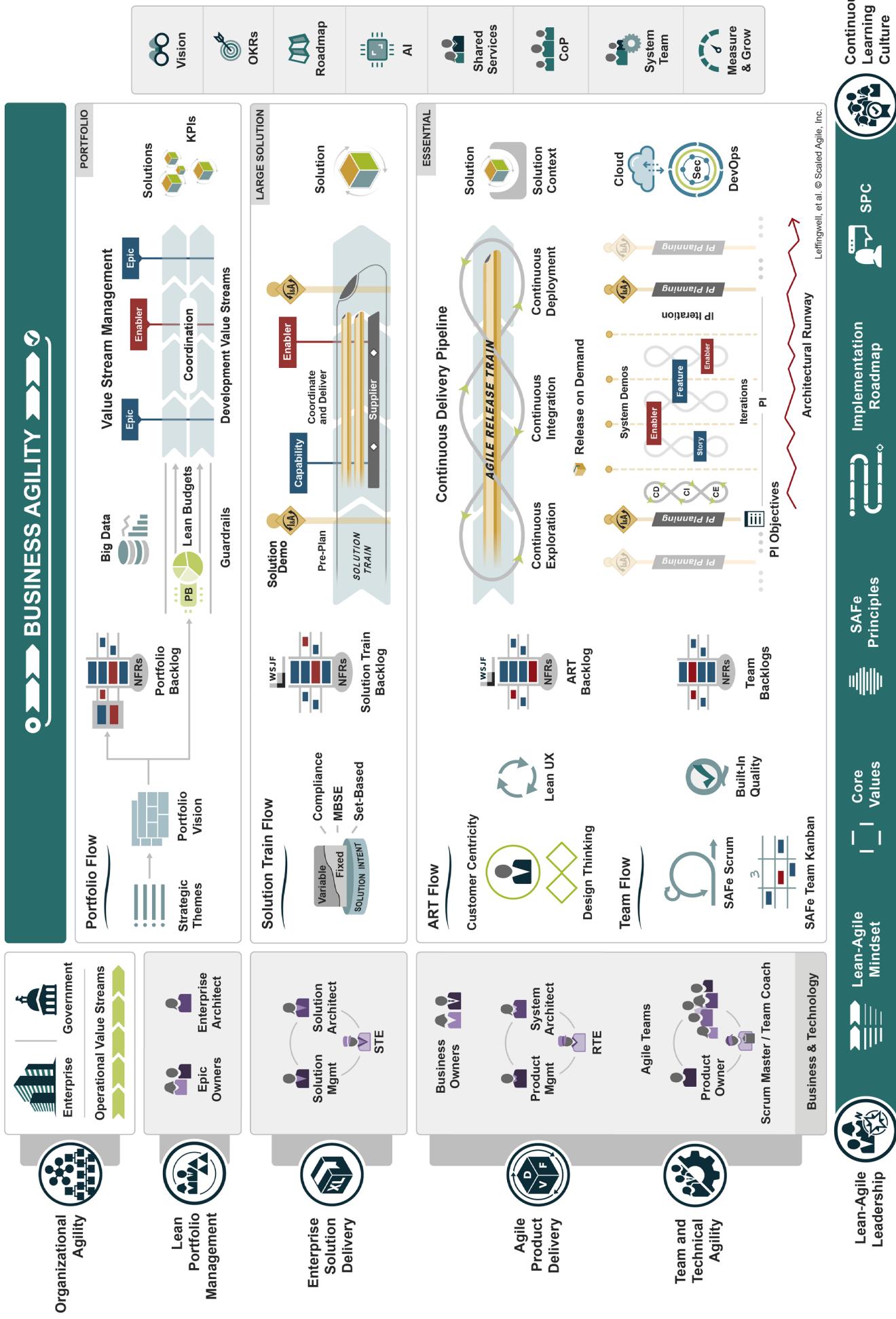
Choose from hundreds of collaboration templates to easily set up events like PI Planning and work in real time with your team and others—all with SAFe Collaborate.



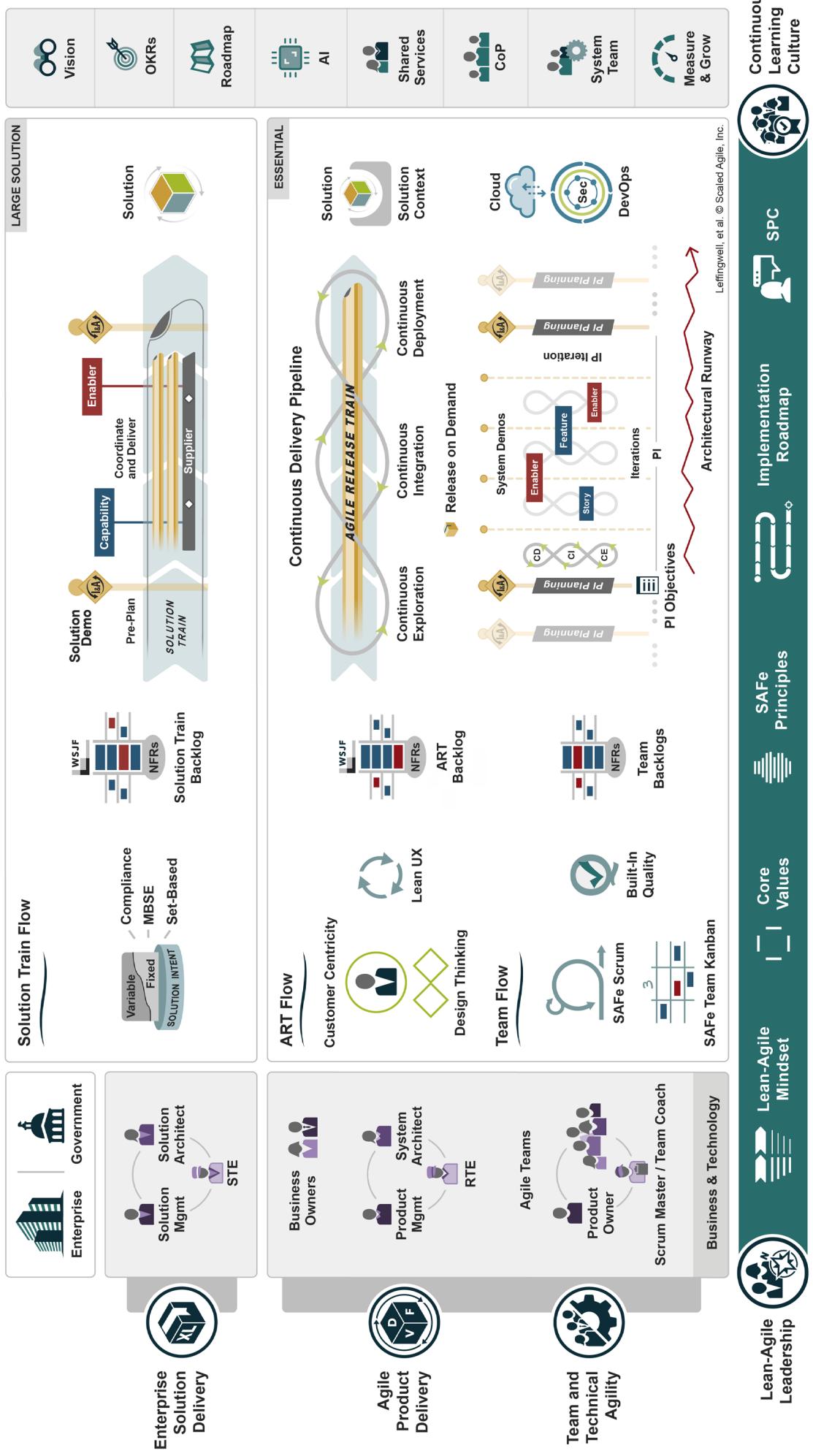
## **Showcase SAFe Credentials**

Display your digital badge to promote your SAFe capabilities and proficiencies throughout your career.

# SAFe® 6.0

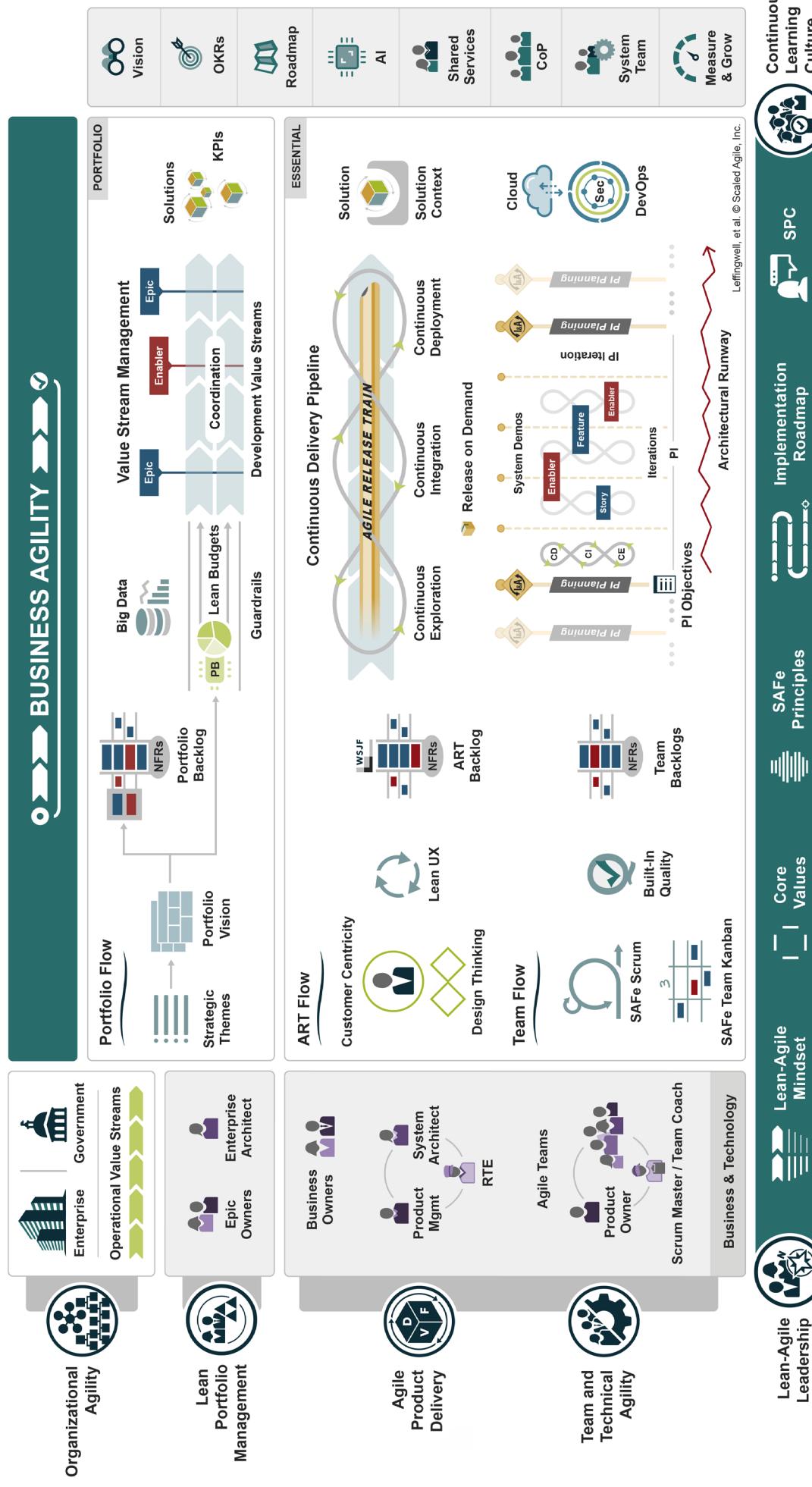


# SAFe® 6.0



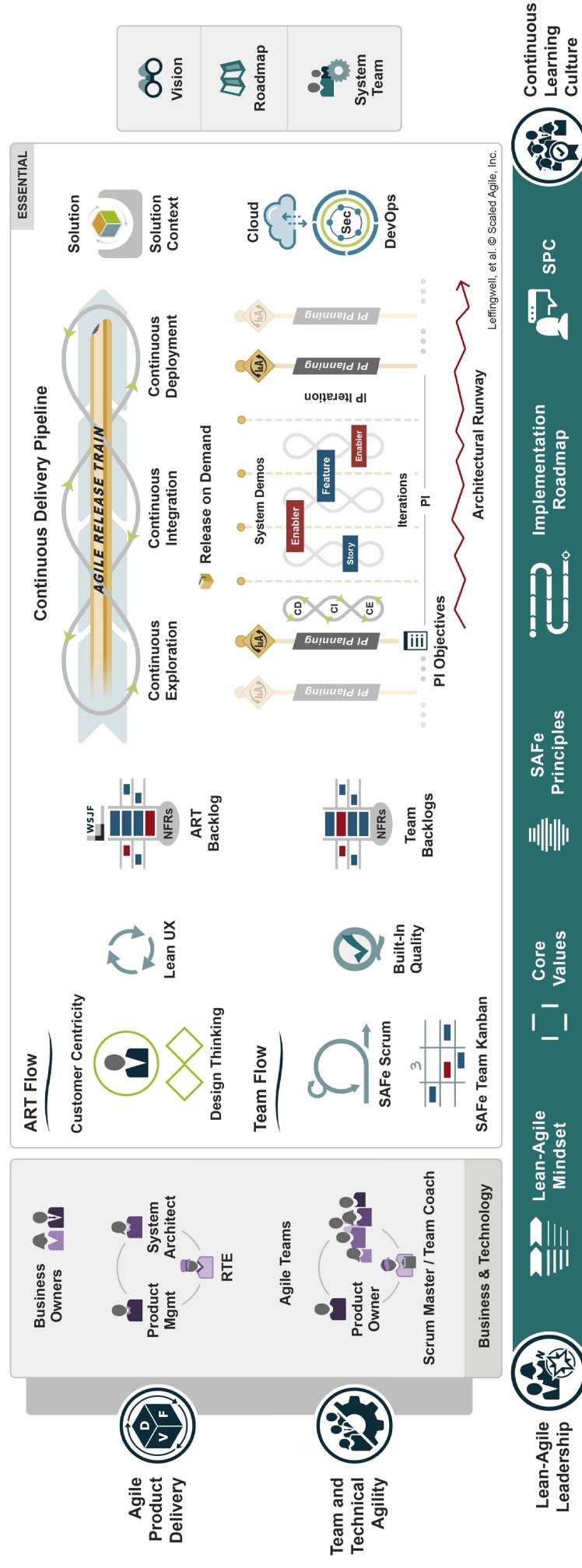
SCALED AGILE®

# SAFe® 6.0



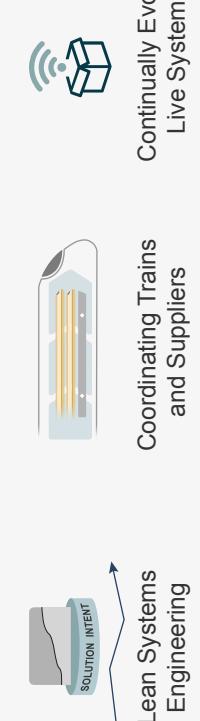
Leffingwell, et al. © Scaled Agile, Inc.

SCALED AGILE®



# BUSINESS AGILITY

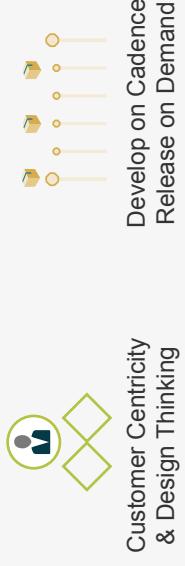
## Enterprise Solution Delivery



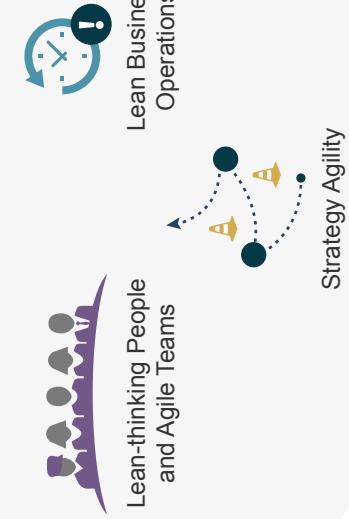
## Lean Portfolio Management



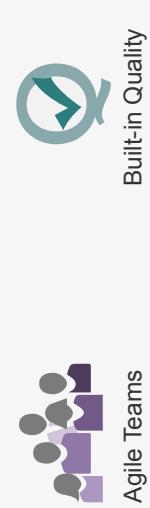
## Agile Product Delivery



## Organizational Agility



## Team and Technical Agility

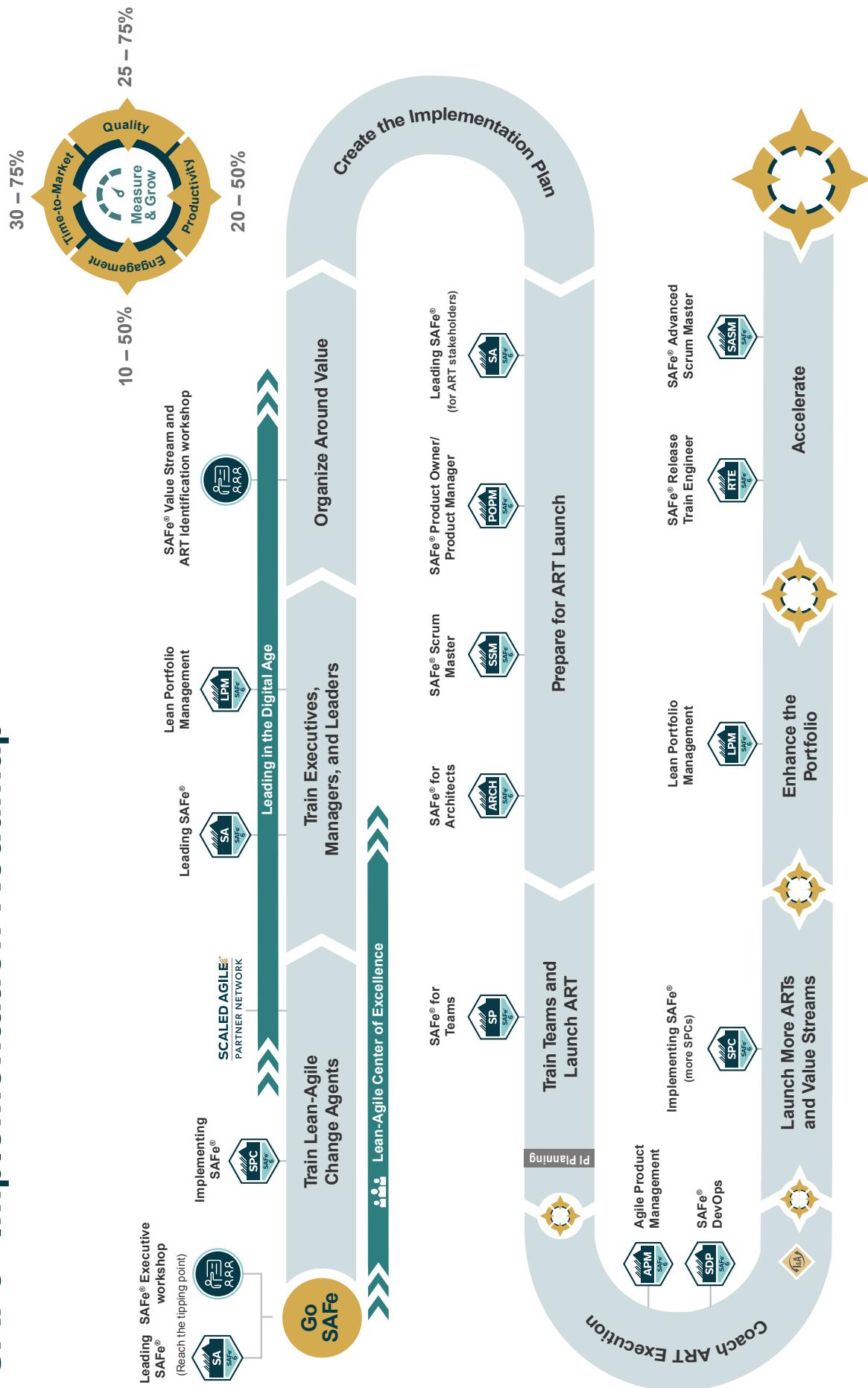


## Lean-Agile Leadership



# SAFe® Implementation Roadmap

## Business results



SCALED AGILE®  
© Scaled Agile, Inc.

# Table of Contents

Privacy Notice .....	10
Course Introduction .....	11
Lesson 1: Introducing SAFe® .....	13
Lesson 2: Form Agile Teams as an Agile Release Train .....	32
Lesson 3: Connect to the Customer .....	54
Lesson 4: Plan the Work.....	72
Lesson 5: Deliver Value .....	109
Lesson 6: Get Feedback.....	132
Lesson 7: Improve Relentlessly.....	149
Lesson 8: Practicing SAFe.....	175
SAFe Glossary .....	180

# Privacy Notice

Your name, company, and email address will be shared with Scaled Agile, Inc. for course fulfillment, including testing and certification. Your information will be used in accordance with the Scaled Agile privacy policy available at <https://www.scaledagile.com/privacy-policy/>.

# SAFe® for Teams

## Establishing Team Agility for Agile Release Trains

SAFe® Course - Attending this course gives learners access to the SAFe® Practitioner exam and related preparation materials.



 **SAFe®** | PROVIDED BY   
© Scaled Agile, Inc.

6.0.2

## Logistics

- ▶ Course meeting times
- ▶ Breaks
- ▶ Facilities
- ▶ Technology requirements
- ▶ Working agreements



## Activity: Access the Class Page

Duration  
5 min

- ▶ **Step 1:** Navigate to the Class Page on SAFe Studio
- ▶ **Step 2:** Select Learn, then My Classes, then SAFe for Teams
- ▶ **Step 3:** Click on the link to download the SAFe for Teams workbook

Start Date	Status	SAFe Collaborate Activities	Download Workbook
Mar 15, 2023	Active		<a href="#">Download Workbook</a>

**SAFe STUDIO™**

Visit the SAFe for Teams Class Page to download the workbook

<https://bit.ly/Studio-MyClasses>

SCALED AGILE® © Scaled Agile, Inc.

1-3

## Course outline

- ▶ Lesson 1: Introducing SAFe
- ▶ Lesson 2: Form Agile Teams as an Agile Release Train
- ▶ Lesson 3: Connect to the Customer
- ▶ Lesson 4: Plan the Work
- ▶ Lesson 5: Deliver Value
- ▶ Lesson 6: Get Feedback
- ▶ Lesson 7: Improve Relentlessly
- ▶ Lesson 8: Practicing SAFe

SCALED AGILE® © Scaled Agile, Inc.

1-4

## Introducing the SAFe® Practitioner Action Plan

In your workbook, you will find Action Plans after every lesson. Through the Action Plans you will have an opportunity to add ideas, insights, and improvement items as a takeaway from each of the lessons.



1-5

# Lesson 1

## Introducing SAFe®

SAFe® Course - Attending this course gives learners access to the SAFe® Practitioner exam and related preparation materials.



SCALED AGILE®

## Lesson Topics

**1.1** The Scaled Agile Framework

**1.2** The Seven Core Competencies of Business Agility

**1.3** The Lean-Agile Mindset and SAFe Core Values

**1.4** SAFe Principles



## Learning objectives

At the end of this lesson, you should be able to:

- ▶ Recognize SAFe as an operating system for Business Agility
- ▶ Summarize Team and Technical Agility and Agile Product Delivery competencies
- ▶ Describe the Lean-Agile Mindset and SAFe Core Values
- ▶ Identify the SAFe Lean-Agile Principles
- ▶ Identify one or more actions an individual or a team can take to represent the SAFe Core Values and the SAFe Lean-Agile Principles

## 1.1 The Scaled Agile Framework

### Rethinking the organization

“

The world is now changing at a rate at which the basic systems, structures, and cultures built over the past century cannot keep up with the demands being placed on them.”

—John P. Kotter, *Accelerate*

*Accelerate* by John P. Kotter  
Portrait of John Kotter. Photo courtesy of Kotter Inc.



## Business Agility is our opportunity

Business opportunity emerges

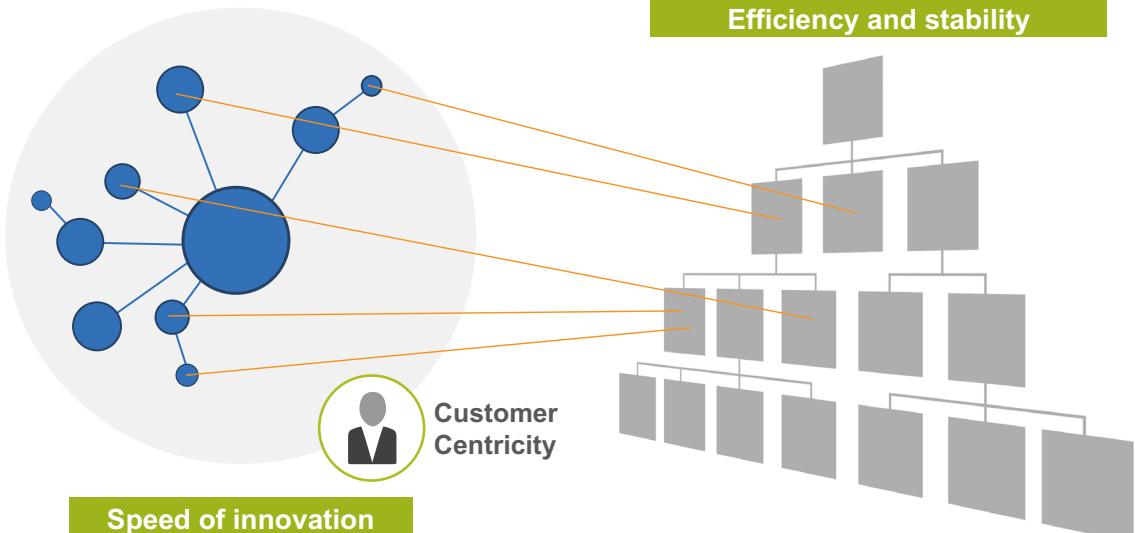
Business opportunity leveraged



SCALED AGILE® © Scaled Agile, Inc.

1-11

## We need a dual operating system for Business Agility

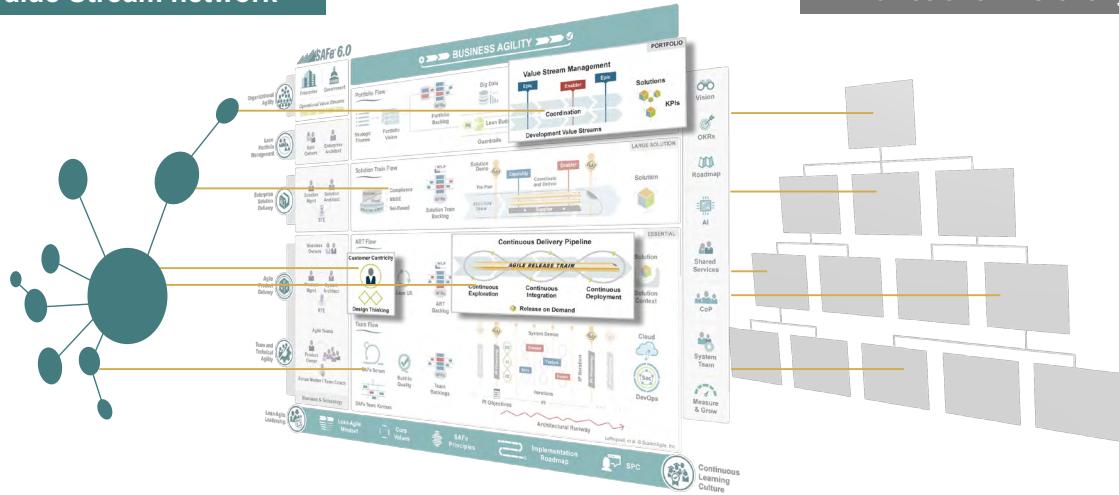


SCALED AGILE® © Scaled Agile, Inc.

1-12

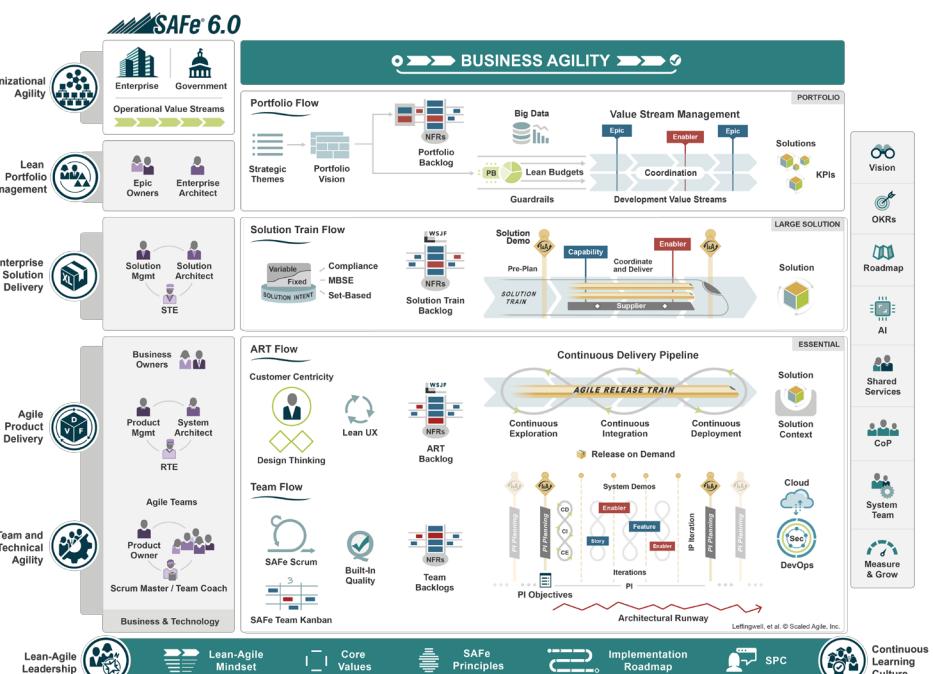
We have just such an operating system at our fingertips

### Value Stream network



SCALED AGILE® © Scaled Agile, Inc.

1-13



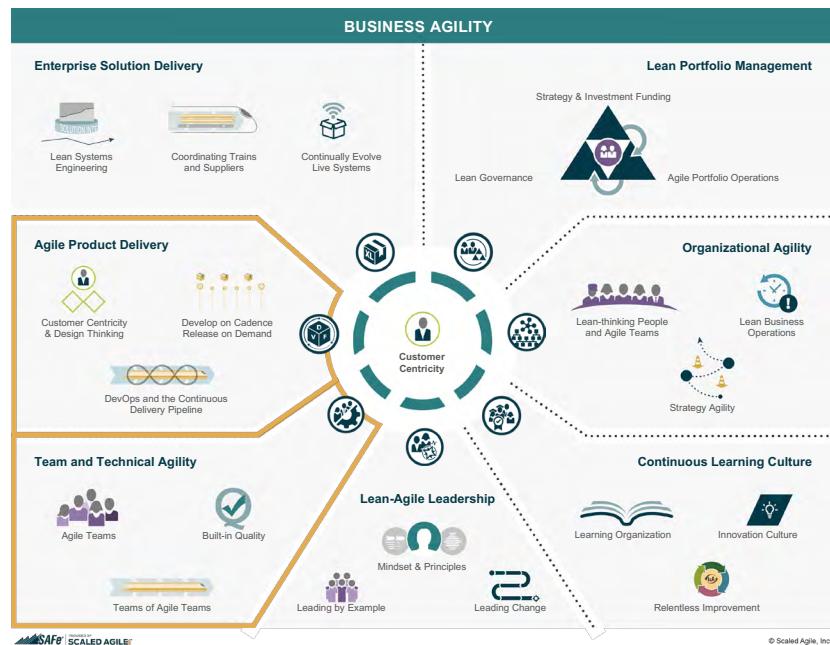
<https://www.scaledagileframework.com/>

1-14

## 1.2 The Seven Core Competencies of Business Agility

SCALED AGILE® © Scaled Agile, Inc.

1-15



<https://www.scaledagileframework.com/>

1-16



## Team and Technical Agility

- ▶ High-performing, cross-functional Agile Teams
- ▶ Teams of business and technical teams build Solutions
- ▶ Quality business Solutions delight Customers

### Agile Teams



### Teams of Agile Teams



### Built-In Quality



SCALED AGILE® © Scaled Agile, Inc.

1-17



## Agile Product Delivery

- ▶ The Customer is the center of your product strategy
- ▶ Decouple the release of value from the development cadence
- ▶ Continuously explore, integrate, deploy, and release

### Customer Centricity and Design Thinking



### Develop on cadence and Release on Demand



### DevOps and the Continuous Delivery Pipeline



SCALED AGILE® © Scaled Agile, Inc.

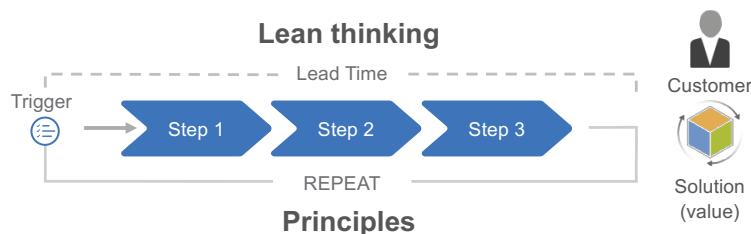
1-18

## 1.3 The Lean-Agile Mindset and SAFe Core Values

SCALED AGILE® © Scaled Agile, Inc.

1-19

### Lean thinking



#### Principles

- 1 Precisely specify value by product
- 2 Identify the Value Stream for each product
- 3 Make value flow without interruptions
- 4 Let the Customer pull value from the producer
- 5 Pursue perfection

SCALED AGILE® © Scaled Agile, Inc.

1-20

## The Agile Manifesto

### Agile Values

We are uncovering better ways of developing software by doing it and helping others do it

Through this work we have come to value:

**Individuals and interactions** over processes and tools

**Working software** over comprehensive documentation

**Customer collaboration** over contract negotiation

**Responding to change** over following a plan

That is, while there is value in the items on the right,  
we value the items on the left more.

Reference: Agile Manifesto

## The Agile Manifesto principles

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale.
4. Business people and developers must work together daily throughout the project.

## The Agile Manifesto principles

5. Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

## The Agile Manifesto principles

9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity—the art of maximizing the amount of work not done—is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

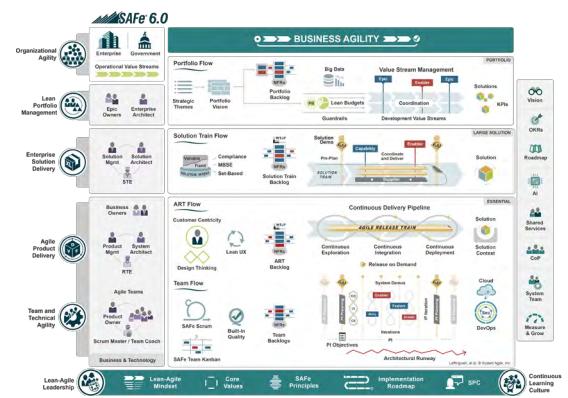
# SAFe Core Values

"Before we build cars,  
we build people."  
— from *The Toyota Way*

Alignment

Transparency

Respect for People



Relentless Improvement

SCALED AGILE® © Scaled Agile, Inc.

1-25

# SAFe Core Values

## Alignment

- ▶ Communicate the vision, mission, and strategy
- ▶ Connect strategy to execution
- ▶ Speak with a common language
- ▶ Constantly check for understanding
- ▶ Understand your Customer

## Transparency

- ▶ Create a trust-based environment
- ▶ Communicate directly, openly, and honestly
- ▶ Turn mistakes into learning moments
- ▶ Visualize work
- ▶ Provide ready access to needed information

SCALED AGILE® © Scaled Agile, Inc.

1-26

## SAFe Core Values

### Respect for People

- ▶ Hold precious what it is to be human
- ▶ Value diversity of people and opinions
- ▶ Grow people through coaching and mentoring
- ▶ Embrace 'Your Customer is whoever consumes your work'
- ▶ Build long-term partnerships based on mutual benefit

### Relentless Improvement

- ▶ Create a constant sense of urgency
- ▶ Build a problem-solving culture
- ▶ Reflect and adapt frequently
- ▶ Let facts guide improvements
- ▶ Provide time and space for innovation

SCALED AGILE® © Scaled Agile, Inc.

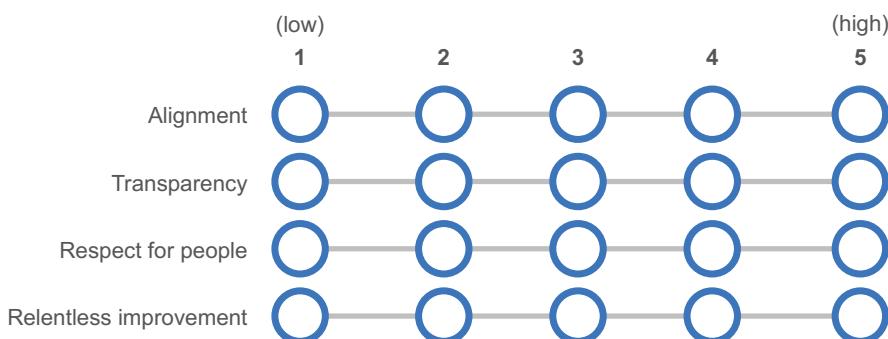
1-27



### Activity: SAFe Core Values



- ▶ **Step 1:** Individually assess where your organization stands in embracing the SAFe Core Values.
- ▶ **Step 2:** Discuss the results of the self-assessment within your group. What similarities and differences emerge?



SCALED AGILE® © Scaled Agile, Inc.

1-28

# SAFe Core Values

## Alignment

- Communicate the vision, mission, and strategy
- Connect strategy to execution
- Speak with a common language
- Constantly check for understanding
- Understand your Customer

## Respect for People

- Hold precious what it is to be human
- Value diversity of people and opinions
- Grow people through coaching and mentoring
- Embrace 'Your customer is whoever consumes your work'
- Build long-term partnerships based on mutual benefit

## Transparency

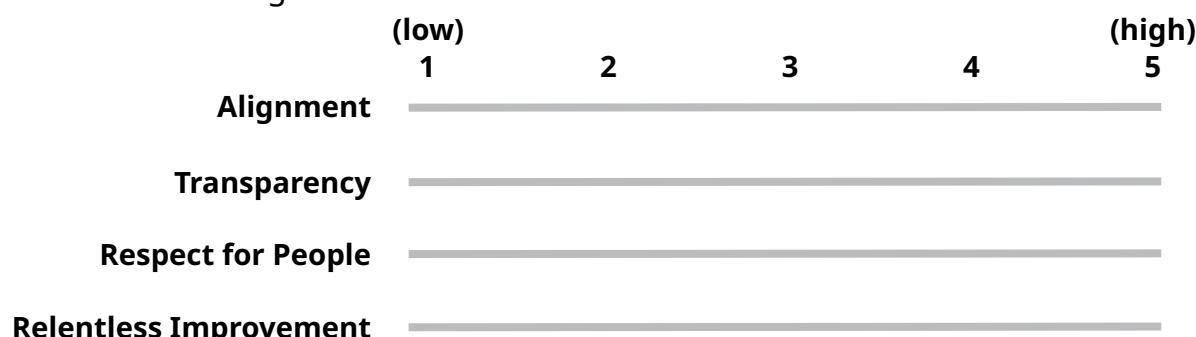
- Create a trust based environment
- Communicate directly, openly, and honestly
- Turn mistakes into learning moments
- Visualize work
- Provide ready access to needed information

## Relentless Improvement

- Create a constant sense of urgency
- Build a problem-solving culture
- Reflect and adapt frequently
- Let facts guide improvements
- Provide time and space for innovation

**Step 1:** Individually assess where your organization stands in embracing the SAFe Core Values.

**Step 2:** Discuss the results of the self-assessment within your group. What similarities and differences emerge?



## Notes

## 1.4 SAFe Principles

### SAFe Lean-Agile Principles

#1 Take an economic view

#2 Apply systems thinking

#3 Assume variability; preserve options

**#4 Build incrementally with fast, integrated learning cycles**

**#5 Base milestones on objective evaluation of working systems**

**#6 Make value flow without interruptions**

**#7 Apply cadence, synchronize with cross-domain planning**

#8 Unlock the intrinsic motivation of knowledge workers

#9 Decentralize decision-making

**#10 Organize around value**



## Action Plan: Introducing SAFe

Prepare      Share  
7 min      3 min

- ▶ **Step 1:** Individually in your workbook, brainstorm some daily actions you can take that will enable you to represent the SAFe Core Values. Add them to your Action Plan.
- ▶ **Step 2:** Share your ideas with your team.
- ▶ **Step 3:** With your team, determine which SAFe Principle to concentrate on first by looking at which one seems to have the highest value impact. Identify one action your team can take to represent the principle in daily practice. Add to your Action Plan.
- ▶ **Step 4:** Be prepared to share with the class.





# Action Plan

## Introducing SAFe

### Core Values

- Alignment
- Respect for People
- Transparency
- Relentless Improvement

### Principles

#1 Take an economic view

#2 Apply systems thinking

#3 Assume variability; preserve options

#4 Build incrementally with fast, integrated learning cycles

#5 Base milestones on objective evaluation of working systems

#6 Make Value Flow without interruptions

#7 Apply cadence, synchronize with cross-domain planning

#8 Unlock the intrinsic motivation of knowledge workers

#9 Decentralize decision-making

#10 Organize around value

## Lesson review

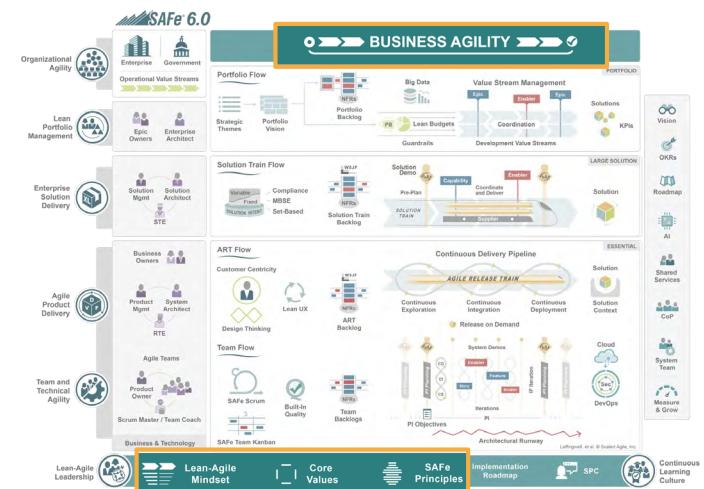
In this lesson you:

- ▶ Recognized SAFe as an operating system for Business Agility
- ▶ Summarized Team and Technical Agility and Agile Product Delivery competencies
- ▶ Described the Lean-Agile Mindset and SAFe Core Values
- ▶ Identified the SAFe Lean-Agile Principles
- ▶ Identified one or more actions an individual or a team can take to represent the SAFe Core Values and the SAFe Lean-Agile Principles

## Articles used in this lesson

Read these Framework articles to learn more about topics covered in this lesson:

- ▶ “Core Values”  
<https://www.scaledagileframework.com/safe-core-values/>
- ▶ “Lean-Agile Mindset”  
<https://www.scaledagileframework.com/lean-agile-mindset/>
- ▶ “SAFe Lean-Agile Principles”  
<https://www.scaledagileframework.com/safe-lean-agile-principles/>
- ▶ “Business Agility”  
<https://www.scaledagileframework.com/business-agility>



## Continue your SAFe journey with the following resources:

Watch this three-minute video, <i>Navigating the Big Picture</i> , to understand how to use the SAFe Big Picture. <a href="https://bit.ly/Video-NavigatingBP6">https://bit.ly/Video-NavigatingBP6</a>	Build your knowledge of the goals and methods of SAFe to achieve Business Agility with the <i>What is SAFe for Lean Enterprise</i> online learning. <a href="https://bit.ly/Community-GettingStarted">https://bit.ly/Community-GettingStarted</a>
Watch this four-minute video, <i>Lean-Agile Mindset</i> , to learn why a Lean-Agile mindset is an important Enabler for Business Agility. <a href="https://bit.ly/Video-LeanAgileMindset">https://bit.ly/Video-LeanAgileMindset</a>	Complete the online learning, <i>Agile Basics</i> , to learn more about what Agile is and how it supports value delivery. <a href="https://bit.ly/Community-GettingStarted">https://bit.ly/Community-GettingStarted</a>
Download and share the “Introducing SAFe” toolkit to familiarize people in your organization with SAFe. <a href="https://bit.ly/Community-ToolkitsandTemplates">https://bit.ly/Community-ToolkitsandTemplates</a>	Watch this video playlist <i>SAFe Lean-Agile Principles (6.0)</i> to explore each of the ten SAFe Lean-Agile Principles in depth. <a href="https://bit.ly/Playlist-Principles6">https://bit.ly/Playlist-Principles6</a>

## References

Agile Manifesto. “Manifesto for Agile Software Development.” Updated 2001.  
<https://agilemanifesto.org>.

Kotter, John P. *Accelerate: Building Strategic Agility for a Faster-Moving World*. Boston: Harvard Business Review Press, 2014. Kindle.

Liker, Jeffrey K. *The Toyota Way: 14 Management Principles from the World’s Greatest Manufacturer*. McGraw-Hill: New York, 2004. 199.

## Lesson notes

Enter your notes below. If using a digital workbook, save your PDF often so you don't lose any of your notes.

## Lesson 2

# Form Agile Teams as an Agile Release Train

SAFe® Course - Attending this course gives learners access to the SAFe® Practitioner exam and related preparation materials.



SCALED AGILE®

### Lesson Topics

- 2.1** Forming cross-functional Agile Teams
- 2.2** The Scrum Master / Team Coach and Product Owner roles
- 2.3** SAFe Scrum and SAFe Team Kanban
- 2.4** Becoming an Agile Release Train



## Learning objectives

At the end of this lesson, you should be able to:

- ▶ Form your Agile Team
- ▶ Explain the characteristics and responsibilities of an Agile Team
- ▶ Describe the SAFe Scrum Master / Team Coach and SAFe Product Owner roles
- ▶ Explain SAFe Scrum and SAFe Team Kanban
- ▶ Identify the roles within an Agile Release Train (ART)
- ▶ Identify one or more actions individuals or teams can take to develop cross-functional skillsets within their Agile Team

## 2.1 Forming cross-functional Agile Teams

## What are Agile Teams?

In SAFe, an Agile Team is a cross-functional group of generally ten or fewer individuals who define, build, test, and deliver value in short increments while continuously learning and adjusting.

- ▶ Optimized for communication and delivery of value
- ▶ Deliver value every two weeks
- ▶ Contain two specialty roles:
  - Scrum Master / Team Coach (SM/TC)
  - Product Owner (PO)



SCALED AGILE® © Scaled Agile, Inc.

2-5

## The power of a high-performing team:

### The people, the work, and the knowledge are all one

- ▶ A self-organizing team dynamically interacts with itself and the organization
- ▶ Team members create new points of view and resolve contradictions through dialogue
- ▶ The team is energized with intentions, Vision, interest, and mission
- ▶ Leaders provide autonomy, variety, trust, and commitment



2-6

## Responsibilities of the Agile Team



SCALED AGILE® © Scaled Agile, Inc.

2-7

## Characteristics of high-performing Agile Teams

- ▶ Foster the psychological safety needed to take risks without fear of embarrassment or punishment
- ▶ Have diverse knowledge and skills to make quick, effective decisions independently
- ▶ Trust each other, allowing for both healthy conflict and reliance on others
- ▶ Align on a shared Vision with clear goals and purpose
- ▶ Accountable to each other and the organization for reliably completing quality work
- ▶ Meet commitments
- ▶ Understand their work's broader impact on the organization
- ▶ Enjoy their work and working together

SCALED AGILE® © Scaled Agile, Inc.

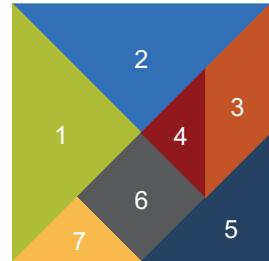
2-8



## Activity: Experience Teams



- ▶ **Step 1:** In your groups, select one team member to act as the timekeeper.
- ▶ **Step 2:** Working with your team, review the puzzle images to be built and the available pieces. Then solve the puzzle as quickly as possible within a three-minute time limit.
- ▶ **Step 3:** After the first Iteration, run a one-minute retrospective to discuss your process and how it can be improved.
- ▶ **Step 4:** Complete two more Iterations with different puzzle shapes. Discuss your approach after each Iteration in your team's one-minute retrospective.
- ▶ **Step 5:** Be prepared to share with the class.



**SCALED AGILE®** © Scaled Agile, Inc.

2-9

## Teams on the ART are composed of different team topologies



**Stream-aligned team** – Organized around the flow of work and has the ability to deliver value directly to the Customer or end user



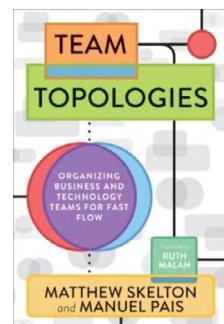
**Complicated subsystem team** – Organized around specific subsystems that require deep specialty skills and expertise



**Platform team** – Organized around the development and support of platforms that provide services to other teams



**Enabling team** – Organized to assist other teams with specialized capabilities and help them become proficient in new technologies

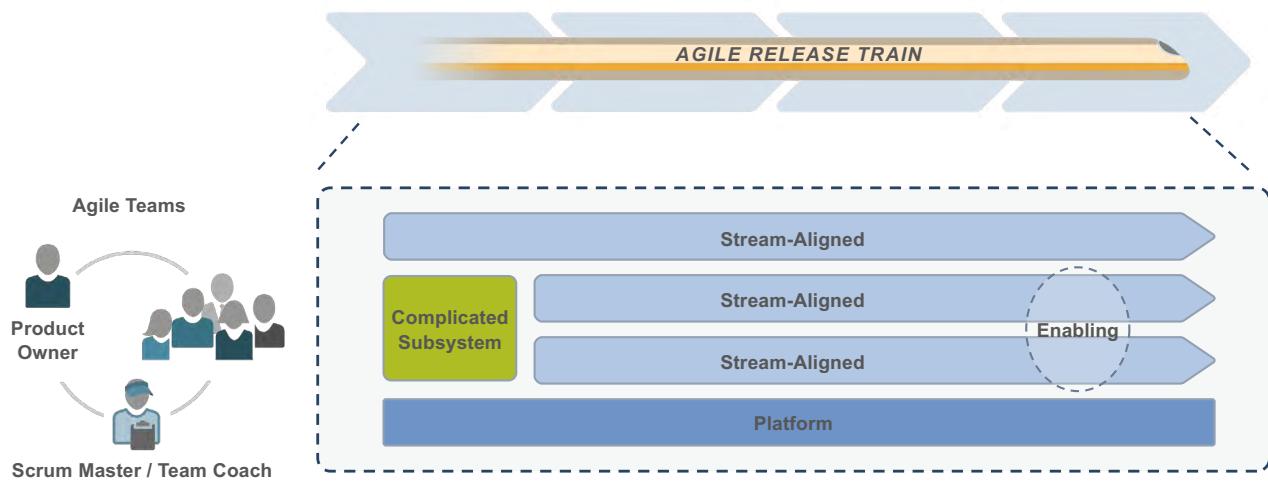


Team Topologies by Matthew Skelton and Manuel Pais

**SCALED AGILE®** © Scaled Agile, Inc.

2-10

## Team topologies help align interactions between teams



SCALED AGILE® © Scaled Agile, Inc.

2-11



### Activity: Identifying your team topology



- ▶ **Step 1:** As a group, discuss your team's responsibilities and skill sets
- ▶ **Step 2:** Use the team topologies on the following two slides to consider what behaviors need to be present on the team in relation to its responsibilities within the larger Solution
- ▶ **Step 3:** Select which of the four team topologies best applies to your team
- ▶ **Step 4:** Be ready to present your findings as part of an activity later in this lesson

SCALED AGILE® © Scaled Agile, Inc.

2-12

## Applying the four topologies – team behavior

FOR REFERENCE ONLY

### Stream-aligned teams

- ▶ Develop a steady flow of new features
- ▶ Support the solution in production
- ▶ Respond to customer need
- ▶ Collaborate with other teams
- ▶ Are cross-functional and long-lasting
- ▶ Develop efficiencies over time

### Platform teams

- ▶ Collaborate with stream-aligned teams in service of end customer requirements
- ▶ Focus on usability and self-service capabilities
- ▶ Lead by example, keeping platforms thin and fit for purpose
- ▶ Build and deploy incrementally with frequent feedback

SCALED AGILE® © Scaled Agile, Inc.

2-13

## Applying the four topologies – continued

FOR REFERENCE ONLY

### Enabling teams

- ▶ Identify improvement opportunities including new technology and practices
- ▶ Develop knowledge in other team types over time
- ▶ Collaborate with other teams proactively
- ▶ Communicate with organization new technologies and emerging methods
- ▶ Exemplify a continuous learning culture
- ▶ Short-lived by intention

### Complicated Subsystem teams

- ▶ Maintain deep expertise and ongoing technical excellence
- ▶ Plan and prioritize effectively aligned to needs of stream-aligned teams
- ▶ Develop appropriate interfaces which hide complexity
- ▶ Ensure quality, performance and architectural robustness of the subsystem

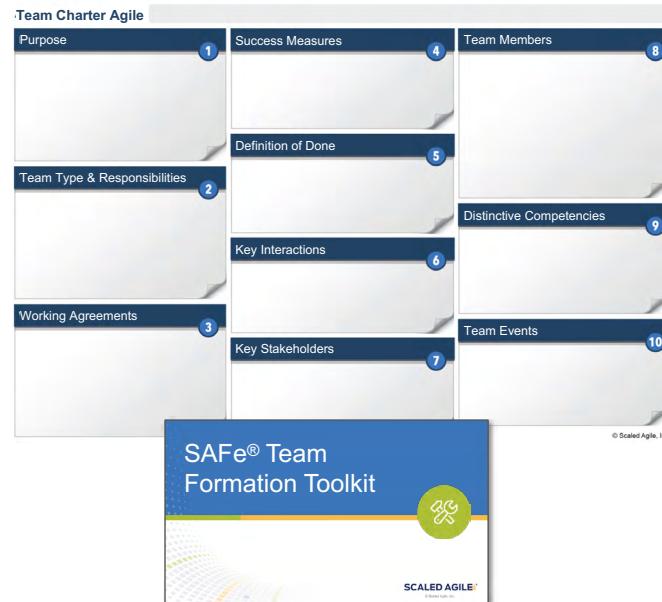
SCALED AGILE® © Scaled Agile, Inc.

2-14

## Facilitate an Agile Team Charter workshop

Part of the SAFe Team Formation Toolkit:

- ▶ The Agile Team Charter helps teams to clearly define their purpose, responsibilities, and success criteria, amongst other critical elements
- ▶ The process of completing the Agile Team Charter provides the opportunity for teams to discuss and reflect on how they want to work with each other and with other teams on the ART
- ▶ For each of the ten boxes, there's an interactive exercise to generate discussion and the required output



SALE AGILE® © Scaled Agile, Inc.

2-15

## 2.2 The Scrum Master / Team Coach and Product Owner roles

SALE AGILE® © Scaled Agile, Inc.

2-16

## Agile Teams have two speciality roles



### SM/TC

- Facilitates PI Planning
- Supports Iteration Execution
- Improves Flow
- Builds a high-performing team
- Improves ART Performance

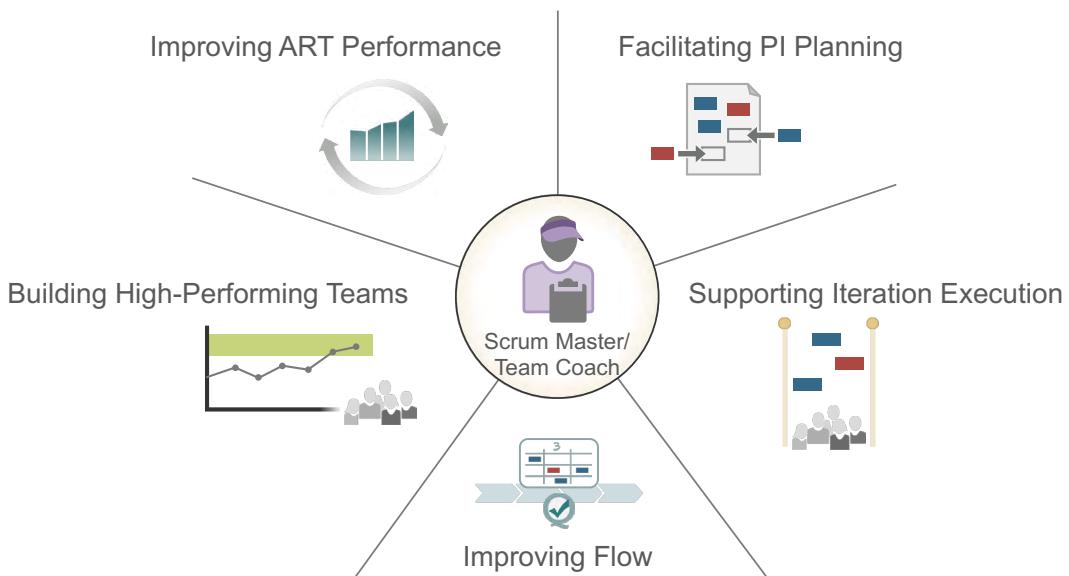
### PO

- Connects with the Customer
- Contributes to the Vision and Roadmap
- Manages and prioritizes the Team Backlog
- Supports the Team in Delivering Value
- Gets and Applies Fast Feedback

SCALED AGILE® © Scaled Agile, Inc.

2-17

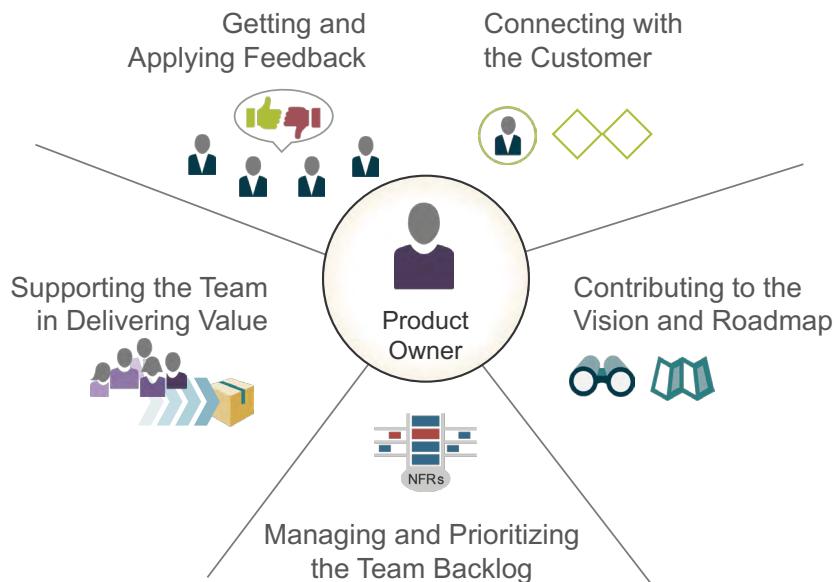
## The SM/TC responsibilities



SCALED AGILE® © Scaled Agile, Inc.

2-18

## The PO responsibilities



SCALED AGILE® © Scaled Agile, Inc.

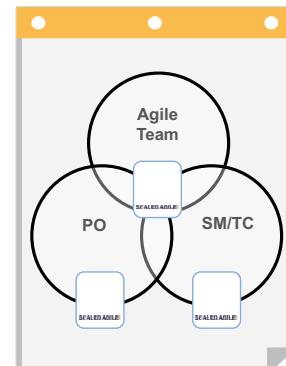
2-19



## Activity: Team roles and responsibilities



- ▶ **Step 1:** With your group, draw the following Venn diagram
- ▶ **Step 2:** Review the provided responsibility cards
- ▶ **Step 3:** Place them either in the circle for a role or in an applicable intersection on the Venn diagram
- ▶ **Step 4:** Present your Venn diagram to the class



SCALED AGILE® © Scaled Agile, Inc.

2-20

## 2.3 SAFe Scrum and SAFe Team Kanban

### SAFe Lean-Agile Principles

#1 Take an economic view

#2 Apply systems thinking

#3 Assume variability; preserve options

#### **#4 Build incrementally with fast, integrated learning cycles**

#5 Base milestones on objective evaluation of working systems

#### **#6 Make value flow without interruptions**

#7 Apply cadence, synchronize with cross-domain planning

#8 Unlock the intrinsic motivation of knowledge workers

#9 Decentralize decision-making

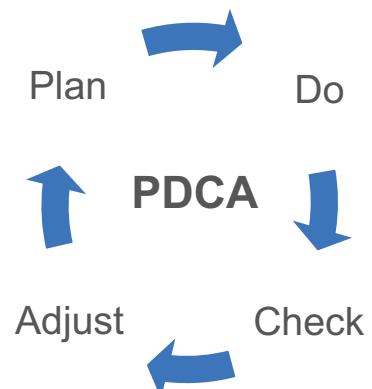
#10 Organize around value

## #4 Build incrementally with fast, integrated learning cycles

Fast feedback accelerates knowledge.

- ▶ Improves learning efficiency by decreasing the time between action and effect
- ▶ Reduces the cost of risk-taking by truncating unsuccessful paths quickly
- ▶ Is facilitated by small batch sizes
- ▶ Requires increased investment in development environment

### The iterative learning cycle



**The shorter the cycles, the faster the learning.**

SCALED AGILE® © Scaled Agile, Inc.

2-23

## #6 Make value flow without interruptions

Instead of a large group...



working on all the requirements...



and integrating and delivering value toward the end of development...



have small teams aligned together...



working on small batches of requirements...

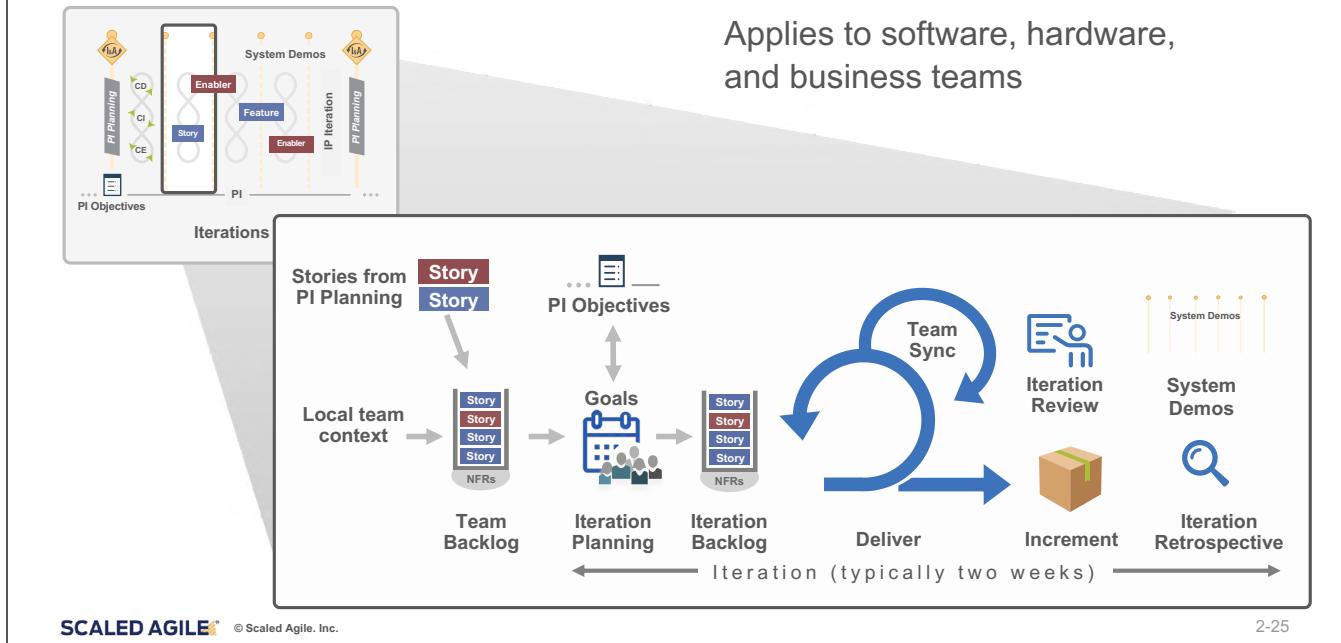


and delivering value in short timeboxes with frequent integration and improvement cycles.

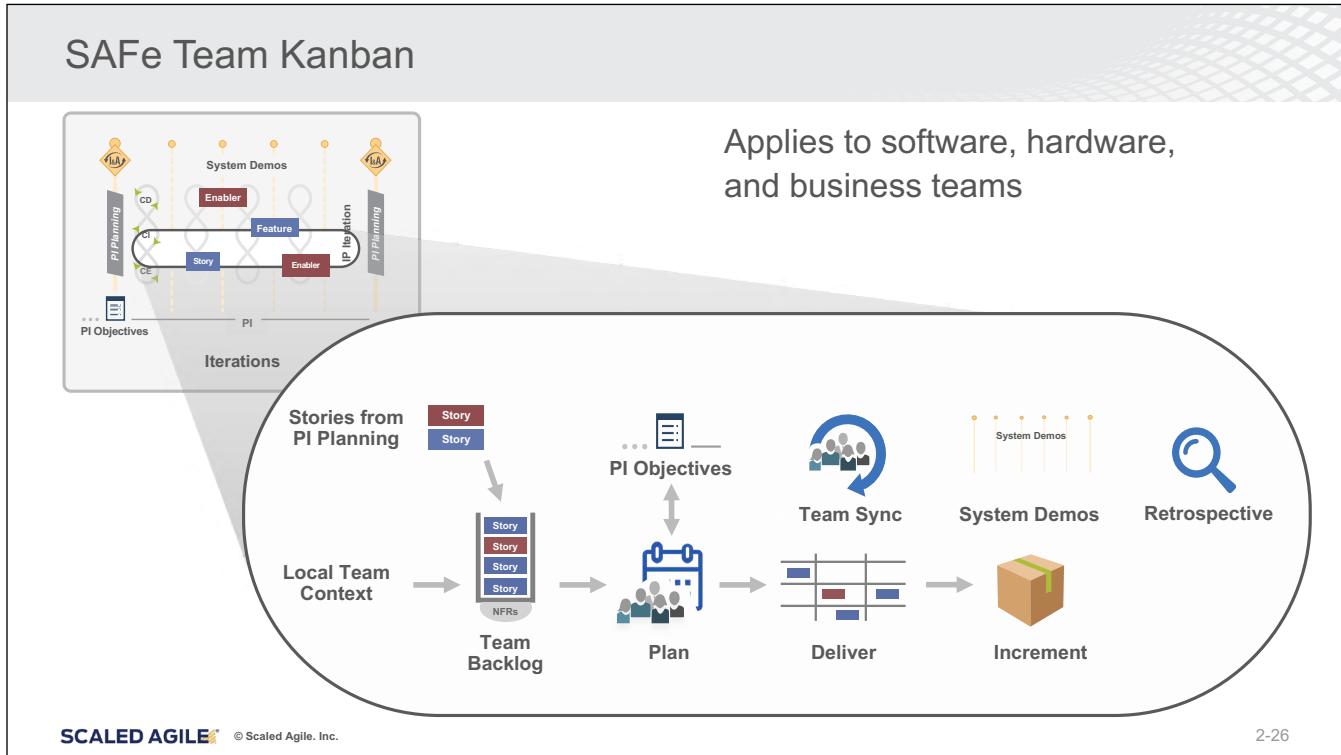
SCALED AGILE® © Scaled Agile, Inc.

2-24

## Starting with SAFe Scrum



## SAFe Team Kanban





## Video: Designing Your Team's Kanban System

Duration  
5 min

# Designing Your Team's Kanban System

<https://bit.ly/Video-DesignKanban>

SCALED AGILE® © Scaled Agile, Inc.

2-27

## Benefits of SAFe Scrum and SAFe Team Kanban

### SAFe Scrum

- ▶ Great for new technology Solution teams
- ▶ Provides known methods for team alignment via Scrum events
- ▶ Creates shared team commitment to timeboxed goals

### SAFe Team Kanban

- ▶ Great for new business Solution teams
- ▶ Provides daily ability to react to changing demands
- ▶ Creates priority alignment and next to pull backlog

SCALED AGILE® © Scaled Agile, Inc.

2-28



## Activity: Building your team

Duration  
10 min

- ▶ **Step 1:** As a group, discuss and identify each person's responsibilities and skill sets
- ▶ **Step 2:** Create a group name
  - Names should not be the names of components, subsystems, or Feature areas. Instead, create a fun name, a mascot, or even a cheer.
- ▶ **Step 3:** Use your findings from "Identifying your team topology" and select either SAFe Team Kanban or SAFe Scrum as a team starting point
- ▶ **Step 4:** Prepare a one-minute presentation about your group
  - Include the name, role on the train, and special skills other groups should know about.

SCALED AGILE® © Scaled Agile, Inc.

2-29

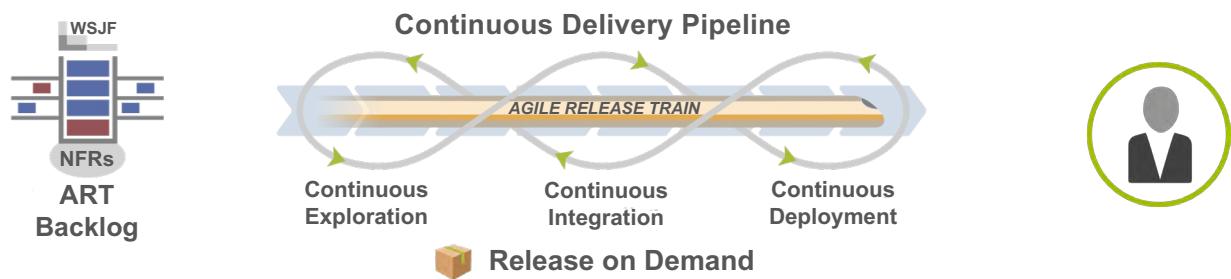
## 2.4 Becoming an Agile Release Train

SCALED AGILE® © Scaled Agile, Inc.

2-30

## ARTs

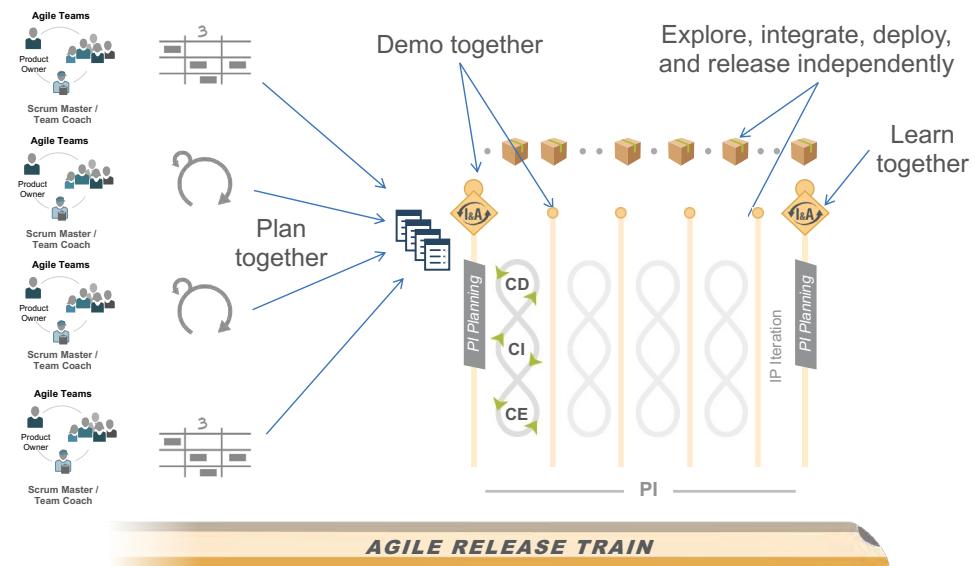
- ▶ A virtual organization of 5 – 12 teams (50 – 125+ individuals)
- ▶ Synchronized on a common cadence, a Planning Interval (PI)
- ▶ Aligned to a common mission via a single ART Backlog



SCALED AGILE® © Scaled Agile, Inc.

2-31

## Teams in SAFe are part of an ART



SCALED AGILE® © Scaled Agile, Inc.

2-32

## Roles on the ART



**Release Train Engineer (RTE)** acts as the chief coach for the train



**System Architect** provides architectural guidance and technical enablement to the teams on the train



**Product Management** owns, defines, and prioritizes the ART Backlog



**System team** provides processes and tools to integrate and evaluate assets early and often



**Business Owners** are key stakeholders on the ART

**AGILE RELEASE TRAIN**

## Responsibilities of the ART

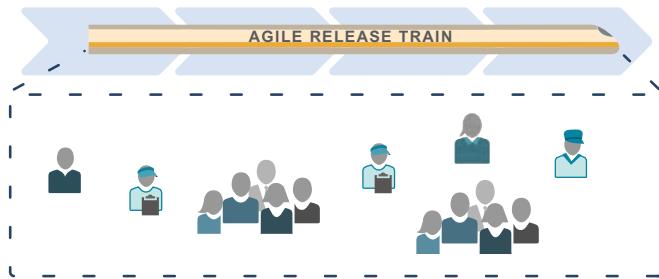




## Activity: Know the people on the train

Duration  
20 min

- ▶ **Step 1:** The RTE introduces themself
- ▶ **Step 2:** The RTE presents the main players on the train:
  - Product Management
  - System Architect
  - Lean User Experience (UX)
  - Shared Services
- ▶ **Step 3:** Each team presents itself (name, area of responsibility, special skills)



SCALED AGILE® © Scaled Agile, Inc.

2-35



## Action Plan: Form Agile Teams as an ART

Prepare  
5 min  
Share  
3 min

- ▶ **Step 1:** Individually, identify one skill you could grow to help expand your team's cross-functional skillset
- ▶ **Step 2:** Share your ideas with your group
- ▶ **Step 3:** In your group, identify pairs or groups of team members that can work together to help each other build the skills identified in the previous step
- ▶ **Step 4:** Add your chosen actions to the Action Plan in your workbook and be prepared to share with the class



SCALED AGILE® © Scaled Agile, Inc.

2-36



## Action Plan

Form Agile Teams  
as an ART

## Lesson review

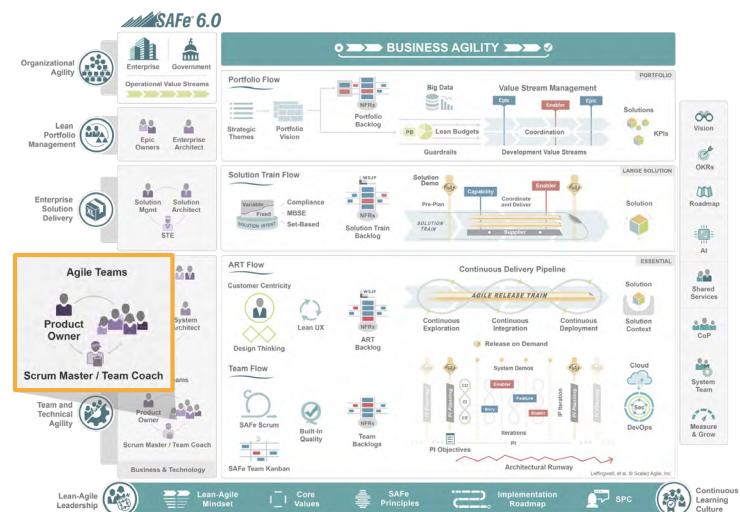
In this lesson you:

- ▶ Formed your Agile Team
- ▶ Explained the characteristics and responsibilities of an Agile Team
- ▶ Described the SAFe SM/TC and SAFe PO roles
- ▶ Explained SAFe Scrum and SAFe Team Kanban
- ▶ Identified the roles within an ART
- ▶ Identified one or more actions individuals or teams can take to develop cross-functional skillsets within their Agile Team

## Articles used in this lesson

Read these Framework articles to learn more about topics covered in this lesson

- ▶ “Agile Teams”  
<https://www.scaledagileframework.com/agile-teams/>
- ▶ “Product Owner”  
<https://www.scaledagileframework.com/product-owner/>
- ▶ “Scrum Master / Team Coach”  
<https://www.scaledagileframework.com/scrum-master-team-coach/>
- ▶ “Agile Release Train”  
<https://www.scaledagileframework.com/agile-release-train/>



## Continue your SAFe journey with the following resources:

Download and use the “Team Formation Toolkit” to build your Agile Team Charter and clearly define your purpose, responsibilities and success criteria, and other critical elements necessary for your new team to flourish. <a href="https://bit.ly/Community-ToolkitsandTemplates">https://bit.ly/Community-ToolkitsandTemplates</a>	Download and use the “Measure and Grow Workshop Toolkit” to run a workshop to identify growth opportunities for your team as you work toward mastering SAFe and Business Agility. <a href="https://bit.ly/Community-ToolkitsandTemplates">https://bit.ly/Community-ToolkitsandTemplates</a>
Download and share the “Essential SAFe Toolkit” with your team to help ensure a shared understanding of the basic building blocks needed for a successful SAFe implementation. <a href="https://bit.ly/Community-ToolkitsandTemplates">https://bit.ly/Community-ToolkitsandTemplates</a>	Facilitate a team-building exercise using the “Experience Teams!” SAFe Collaborate template to practice working as a team and using retrospectives to improve team performance. <a href="https://bit.ly/Template-ExperienceTeams">https://bit.ly/Template-ExperienceTeams</a>
Watch this 61-minute video, <i>Community Webinar: Team Formation Toolkit - Facilitating an Agile Team Charter Workshop</i> to get a deep dive of how to prepare for and run an Agile Team Charter workshop. <a href="https://bit.ly/Video-AgileTeamCharterWebinar">https://bit.ly/Video-AgileTeamCharterWebinar</a>	Use the “SAFe Agile Team Charter” SAFe Collaborate template to help a remote or distributed team capture the ideas and decisions from the “Team Formation Toolkit.” <a href="https://bit.ly/Template-SAFeAgileTeamCharter">https://bit.ly/Template-SAFeAgileTeamCharter</a>

## References

Skelton, Matthew and Manuel Pais. *Team Topologies: Organizing Business and Technology Teams for Fast Flow*. IT Revolution: Portland, 2019. Kindle Edition.

## Lesson notes

Enter your notes below. If using a digital workbook, save your PDF often so you don't lose any of your notes.

# Lesson 3

## Connect to the Customer

SAFe® Course - Attending this course gives learners access to the SAFe® Practitioner exam and related preparation materials.



SCALED AGILE®

### Lesson Topics

- 3.1** Apply Customer Centricity and Design Thinking
- 3.2** Recognize product Vision and Roadmaps
- 3.3** Define the work in support of the Customer



## Learning objectives

At the end of this lesson, you should be able to:

- ▶ Explain the Customer-centric mindset
- ▶ Summarize how to use Design Thinking tools to create Customer-centric products
- ▶ Connect Vision concepts to strategy execution
- ▶ Identify Roadmap techniques
- ▶ Summarize strategies for creating well-written Stories
- ▶ Identify steps your team can take to uncover key details about your Customers

## 3.1 Apply Customer Centricity and Design Thinking



## Discussion: Customer Centricity



- ▶ **Step 1:** Discuss the following with your group:
  - Why is it important to maintain focus on the Customer?
  - What are some of the characteristics of a Customer-centric Enterprise?
  - List your team's primary Customer(s).
- ▶ **Step 2:** Be prepared to share with the class



SCALED AGILE® © Scaled Agile, Inc.

3-5

## Customer Centricity is a mindset

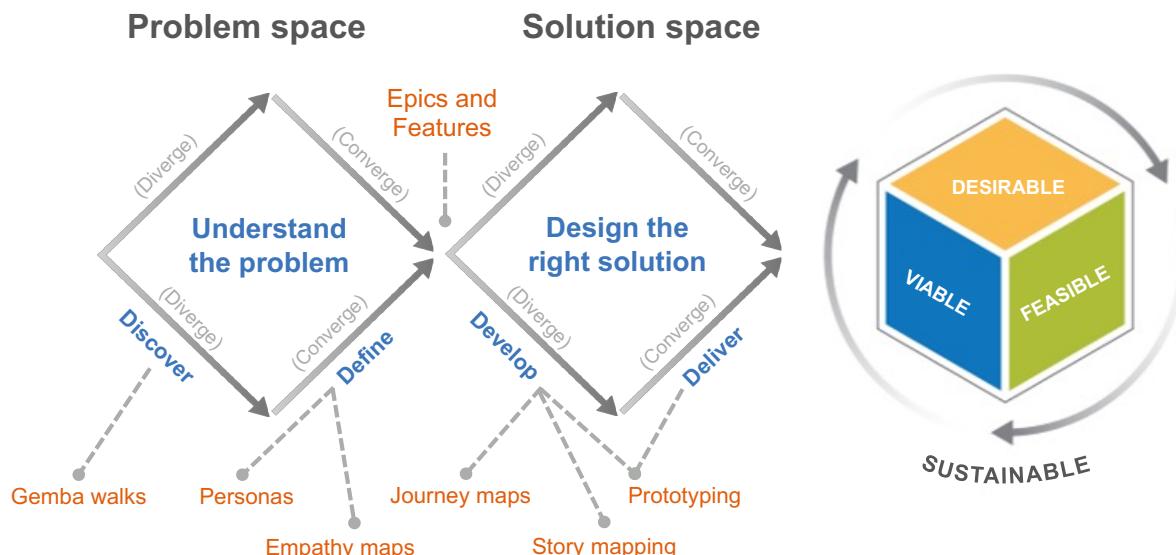
Customer-centric organizations deliver Solutions that are designed with a deep understanding of Customer needs.



SCALED AGILE® © Scaled Agile, Inc.

3-6

## Design Thinking is a Customer-centric development process



SCALED AGILE® © Scaled Agile, Inc.

3-7

## Use personas to understand Customers

Personas are fictional characters that represent the different people who might use your product.

Personas:

- ▶ Convey the problems they're facing in context and key triggers for using the product
- ▶ Capture rich, concise information that inspires great products without unnecessary details



### Cary the Consumer

Age: 36

Location: Reno, Nevada, US

Time in App: 10 minutes

"I'm a working dad with three children ages three, six, and ten. I'm also in a band, which means I want to spend as much time as possible with my kids and my band. I need my package delivered on time so that I can maximize time with my family."

I like technology! I have an iPhone, iPad, and nice home Wi-Fi setup.	I'm not home on some weekends.	I'd rather order online than dial the phone and talk to somebody.
My wife also works during the week, so she doesn't have much spare time to help.	Text is my favorite form of communication with suppliers.	I don't own a computer, only tablets and phones.

SCALED AGILE® © Scaled Agile, Inc.

3-8

## Use empathy maps to identify with Customers

- ▶ The empathy map is a tool that helps teams develop deep, shared understanding and empathy for the Customer
- ▶ Use it to design better User Experiences and Value Streams



## 3.2 Recognize product Vision and Roadmaps

## Vision aligns everyone on the product's direction

The product Vision is a description of the future state of the product.

- ▶ How will the product solve our Customer's problems?
- ▶ What Features does it have?
- ▶ How will it differentiate us?
- ▶ What nonfunctional requirements (NFRs) does it deliver?



3-11

SCALED AGILE® © Scaled Agile, Inc.

## Communicate the vision differently to different audiences

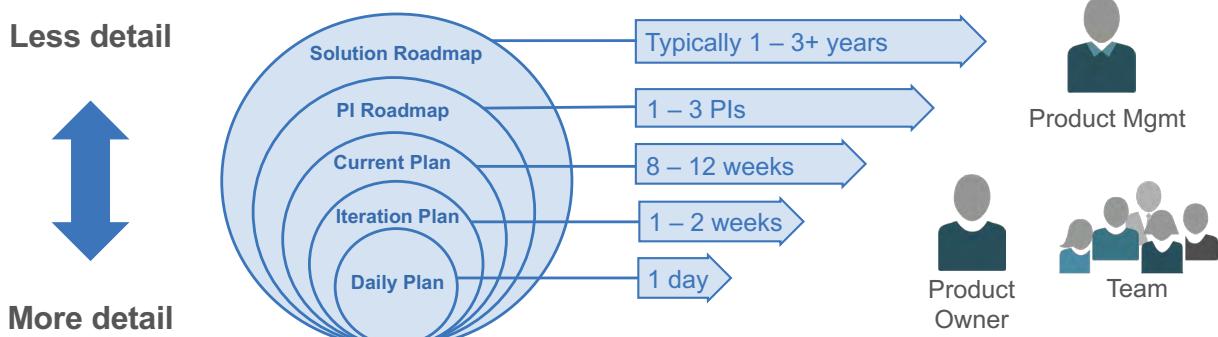
Vision technique	Audience	Goal
Elevator pitch	Marketing/ Sales	Effectively communicate desired competitive positioning
We're #1 statement	Brand team	Align Solution with brand attributes and mission
Vision video, postcard from the future, Vision box	Solution Train and ARTs	Inspire the future and establish context
Vision video	Customers	For business-to-business and business-to-people, maintain and strengthen relationships
	Investors	Secure investors for early stage offers
Cover story / Press release	Solution Train and ARTs	Provide context and inspiration for the PI or next major release Supported by preview or Solution brief

SCALED AGILE® © Scaled Agile, Inc.

3-12

## Roadmaps forecast upcoming work to realize the product Vision

- Hierarchical Roadmaps provide multiple planning horizons
- Roadmaps communicate intent without over-constraining the implementation



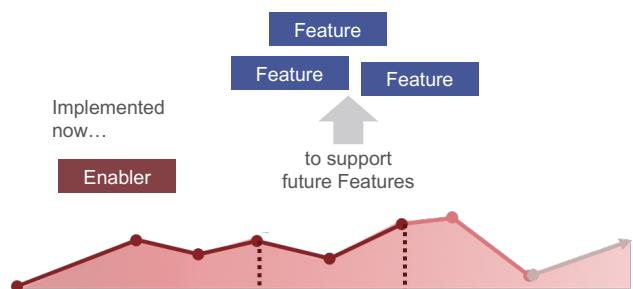
SCALED AGILE® © Scaled Agile, Inc.

3-13

## Architectural Runway supports Vision execution

- Provides the technical Vision to realize the product Vision
- Consists of existing code, hardware components, marketing branding guidelines, and so on, that enable near-term business Features
- Enablers build up the runway
- Features consume it

**Example:**  
A single sign-on mechanism will enable sign-on in multiple applications



SCALED AGILE® © Scaled Agile, Inc.

3-14

## Intentional architecture and emergent design

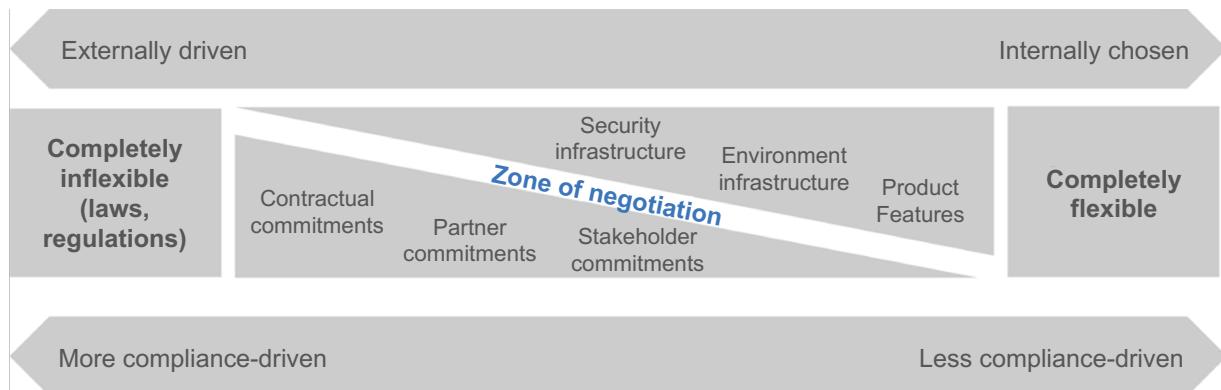
- ▶ **Intentional architecture:** fosters team alignment and defines the Architectural Runway
- ▶ **Emergent design:** teams grow the system design as User Stories require

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

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



## Roadmaps are not commitments

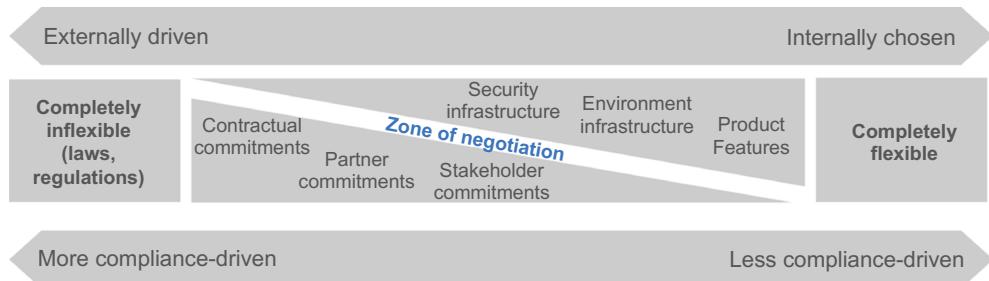




## Activity: What's negotiable in when and how we deliver value?



- ▶ **Step 1:** As a team, consider your upcoming Customer and Enabling work. Use sticky notes to capture your ideas.
- ▶ **Step 2:** Place the sticky notes on the corresponding space on the image and discuss which items have flexibility.
- ▶ **Step 3:** Be prepared to share your rationale with the ART.



SCALED AGILE® © Scaled Agile, Inc.

3-17

## 3.3 Define the work in support of the Customer

SCALED AGILE® © Scaled Agile, Inc.

3-18

## Features represent the work for the ART

- ▶ The Feature benefit hypothesis justifies development cost, Customer value, and provides business perspective for decision-making
- ▶ Features contain acceptance criteria typically defined as part of refining the ART backlog
- ▶ They reflect functional and nonfunctional requirements
- ▶ Features can be completed within a PI timebox

### In-service software update

#### Benefit hypothesis

Significantly reduced planned downtime

#### Acceptance criteria

1. Nonstop routing availability
2. Automatic and manual update support
3. Rollback capability
4. Support through existing admin tools
5. All enabled services are running after the update

#### Example Feature

## Example Features

### Technology example

#### Multi-factor authentication

#### Benefit hypothesis

Enhanced user security will reduce risk of a system data breach

#### 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
5. Data breach tests pass

### Business example

#### Incident response plan for protected customer data

#### Benefit hypothesis

Organizational readiness to quickly respond to incidents

#### Acceptance criteria

1. Incident response plan is fully documented
2. Incident response plan is reviewed and approved by legal and security
3. Incident response is placed within company repository for policies and procedures

## Teams break down Features into User Stories and Enabler Stories

- ▶ User Stories are short descriptions of a small piece of desired functionality, written in the user's language
- ▶ The recommended form of expression is the user-voice form, as follows:  
**As a** (user role), **I want to** (activity), **so that** (business value).

As a driver, I want to limit the amount of money I spend before I fuel so that I can control my expenditure.

As a driver, I want to get a receipt after fueling so that I can expense the purchase.

As the vehicle sensor system, I want to get information from the gas tank so that I can send notifications about how soon a refill is needed to the driver interface.

## Using personas to better understand users

Personas are detailed fictional characters acting as a representative user.



**Jane: Mileage-sensitive**  
– Law-abiding driver  
– obeys all traffic signs  
– Wants to save on gas



**Bob: Time-sensitive**  
– Impatient driver  
– Ignores traffic signs if they slow him down

As Jane, I want to travel at the legal limit and operate in an energy saving manner so that I don't get a ticket and save money.

As Bob, I want to travel at the maximum speed the roadway and my vehicle safely allows so that I arrive quickly.

## INVEST in a good Story

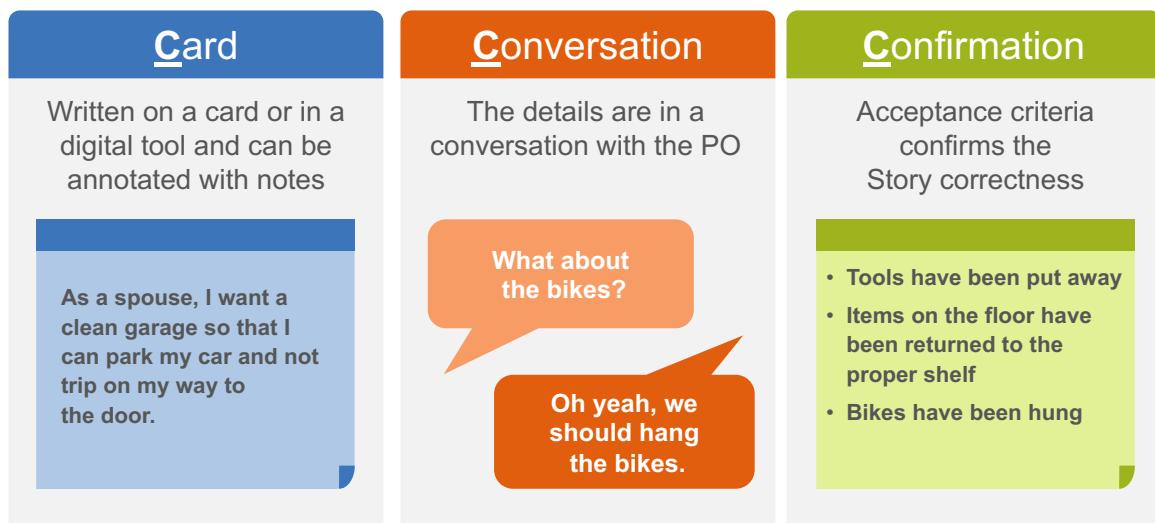
I	<b>Independent</b>	Write Stories that can be developed on their own
N	<b>Negotiable</b>	Write Stories that have a flexible scope
V	<b>Valuable</b>	Write Stories that are useful to the Customer
E	<b>Estimable</b>	Write Stories that can be estimated
S	<b>Small</b>	Write Stories that can fit in an Iteration
T	<b>Testable</b>	Write Stories that are testable

Reference: Wake, "INVEST in Good Stories, and SMART Tasks"

SCALED AGILE® © Scaled Agile, Inc.

3-23

## Writing good Stories: The Three Cs



Reference: Jeffries, "Essential XP: Card, Conversation, Confirmation"

SCALED AGILE® © Scaled Agile, Inc.

3-24

## Enabler Stories

Enabler Stories build the groundwork for future User Stories. There are four types of Enabler Stories:

- ▶ **Infrastructure:** Build development and testing frameworks that enable a faster and more efficient development process
- ▶ **Architecture:** Build the Architectural Runway, which enables smoother and faster development
- ▶ **Exploration:** Build understanding of what is needed by the Customer to understand prospective Solutions and evaluate alternatives
- ▶ **Compliance:** Facilitate specific activities such as verification and validation, documentation, signoffs, regulatory submissions, and approvals



## Action Plan: Connect to the Customer



- ▶ **Step 1:** As a group, identify three or more questions you have about the Customers you identified in the discussion “Customer Centricity” earlier in this lesson
- ▶ **Step 2:** With your group, plan how and when you can work together to answer these questions
- ▶ **Step 3:** Add the outcomes of your discussion to the Action Plan in your workbook





# Action Plan

Connect to the  
Customer

## Lesson Review

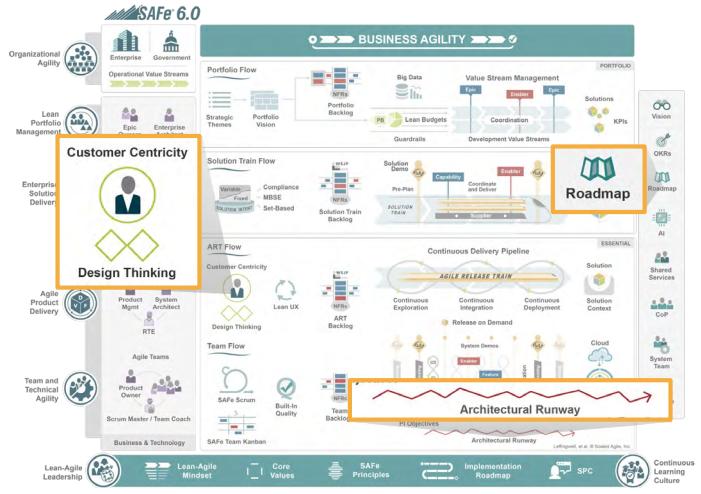
In this lesson you:

- ▶ Explored the Customer-centric mindset
- ▶ Summarized how to use Design Thinking tools to create Customer-centric products
- ▶ Connected Vision concepts to strategy execution
- ▶ Identified Roadmap techniques
- ▶ Explored strategies for creating well-written Stories
- ▶ Identified steps your team can take to uncover key details about your Customers

## Articles used in this lesson

Read these Framework articles to learn more about topics covered in this lesson

- ▶ "Customer Centricity"  
<https://www.scaledagileframework.com/customer-centricity/>
- ▶ "Design Thinking"  
<https://www.scaledagileframework.com/design-thinking/>
- ▶ "Roadmap"  
<https://www.scaledagileframework.com/roadmap/>
- ▶ "Architectural Runway"  
<https://www.scaledagileframework.com/architectural-runway/>



SALE AGILE® © Scaled Agile, Inc.

3-28

## Continue your SAFe journey with the following resources:

Listen to this 30-minute podcast episode, "Customer Centricity in SAFe," to explore how Customer centricity ties to organizational agility.  
<https://bit.ly/Podcast-CustomerCentricityinSAFe>

Watch the two-video series, *Stories*, to learn more about the different types of Stories and how to write them so that they are most effective.  
<https://bit.ly/Video-StoriesPlaylist>

Use the "Empathy Map" SAFe Collaborate template to conduct an empathy mapping exercise with your team to learn more about your Customer and their needs.

<https://bit.ly/Template-EmpathyMap>

Use the "Persona Canvas" SAFe Collaborate Template with your team to practice stepping into the shoes of your Customer.

<https://bit.ly/Template-PersonaCanvas>

SALE AGILE® © Scaled Agile, Inc.

3-29

## References

Jeffries, Ron. "Essential XP: Card, Conversation, Confirmation." Ron Jeffries. August 30, 2001.  
<https://ronjeffries.com/xprog/articles/expcardconversationconfirmation>.

Wake, Bill. "INVEST in Good Stories, and SMART Tasks." XP123. August 17, 2003.  
<https://xp123.com/articles/invest-in-good-stories-and-smart-tasks>.

## Lesson notes

Enter your notes below. If using a digital workbook, save your PDF often so you don't lose any of your notes.

# Lesson 4

## Plan the Work

SAFe® Course - Attending this course gives learners access to the SAFe® Practitioner exam and related preparation materials.



SCALED AGILE®

### Lesson Topics

- 4.1 Creating the backlog
- 4.2 PI Planning
- 4.3 Team Planning



## Learning objectives

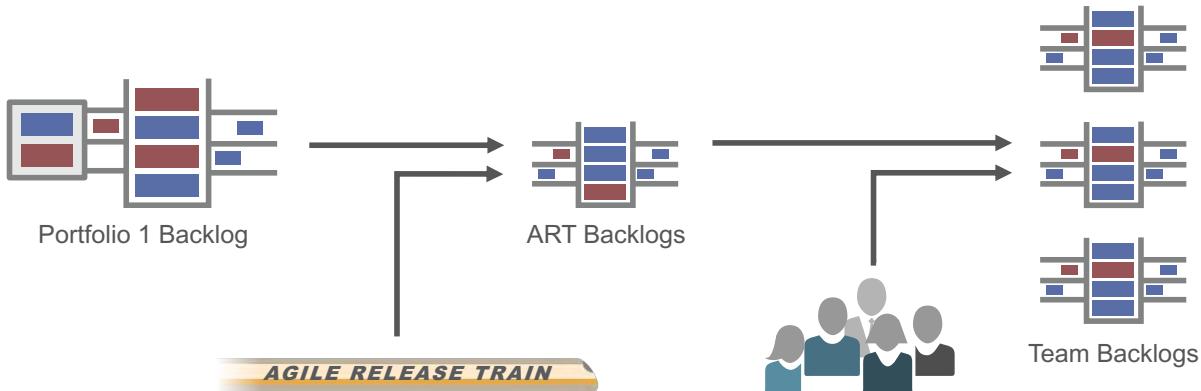
At the end of this lesson, you should be able to:

- ▶ Explain techniques for creating a Team Backlog
- ▶ Practice preparing your backlog by breaking down Features into Stories
- ▶ Apply strategies for creating well-written Stories
- ▶ Explain relative sizing of User Stories
- ▶ Summarize the steps of PI Planning
- ▶ Describe the purpose of Iteration Goals and PI Objectives
- ▶ Summarize the Iteration Planning event and its outcomes
- ▶ Create a set of questions your team can use to understand and size upcoming work

## 4.1 Creating the backlog

## Connected Kanban systems drive backlogs

Work originates locally and from broader contexts

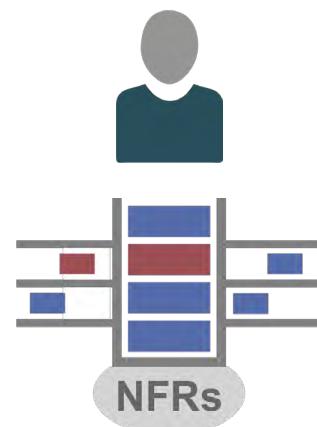


SCALED AGILE® © Scaled Agile, Inc.

4-5

## The Agile Team Backlog is comprised of Stories

- ▶ Contains all the work for the team
- ▶ Created by the PO and the team
- ▶ Prioritized by the PO
- ▶ Contains User and Enabler Stories
  - User Stories provide Customers with value
  - Enabler Stories build the infrastructure and architecture that makes User Stories possible
- ▶ Stories for near-term Iterations are more detailed than Stories for later Iterations
- ▶ NFRs are backlog constraints



SCALED AGILE® © Scaled Agile, Inc.

4-6

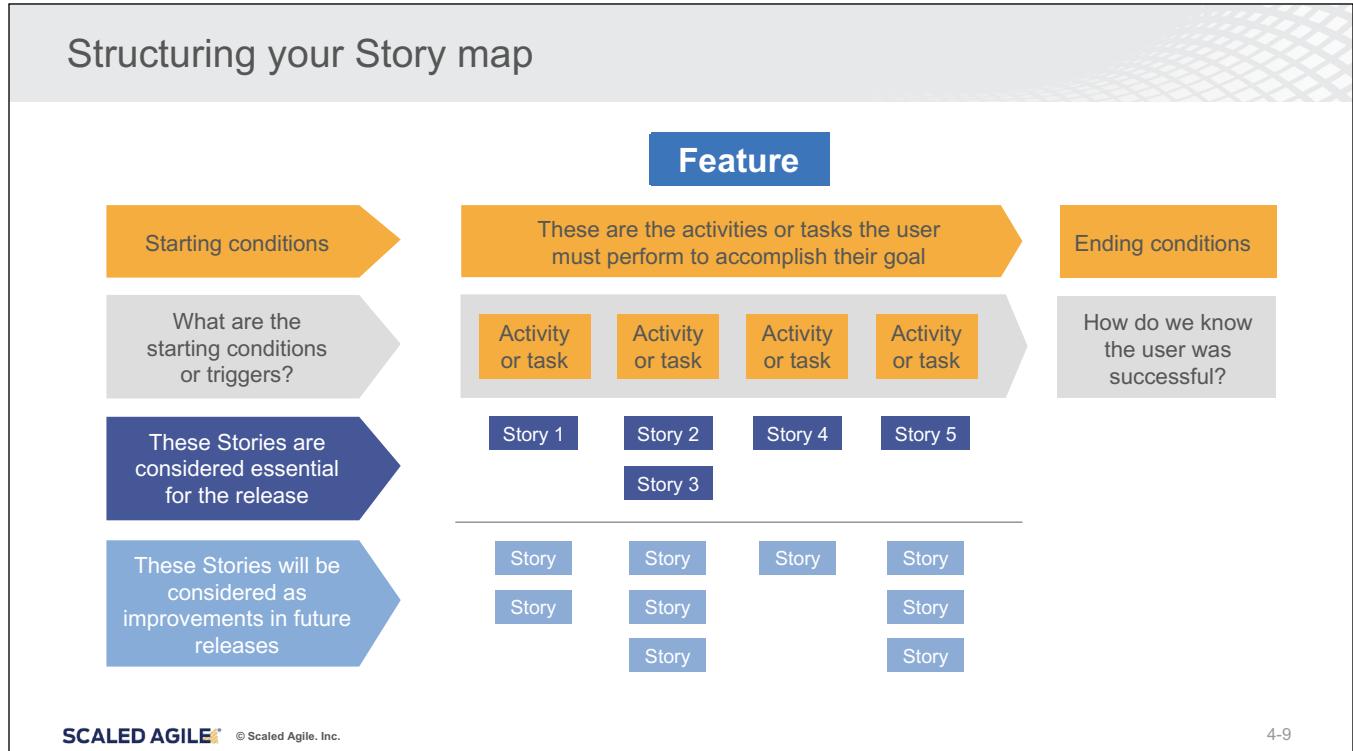
## Backlog prioritization without connection to the user journey

- ▶ Backlogs are optimized to help teams focus on priorities
- ▶ Backlogs have some challenges:
  - It can be difficult to understand workflows
  - The relationship between ‘Stories that make something good’ and ‘Stories that make something better or great’ is lost
  - It is hard to validate that the Stories in the backlog support all steps needed by the user to accomplish their objective

## Story maps overcome these challenges

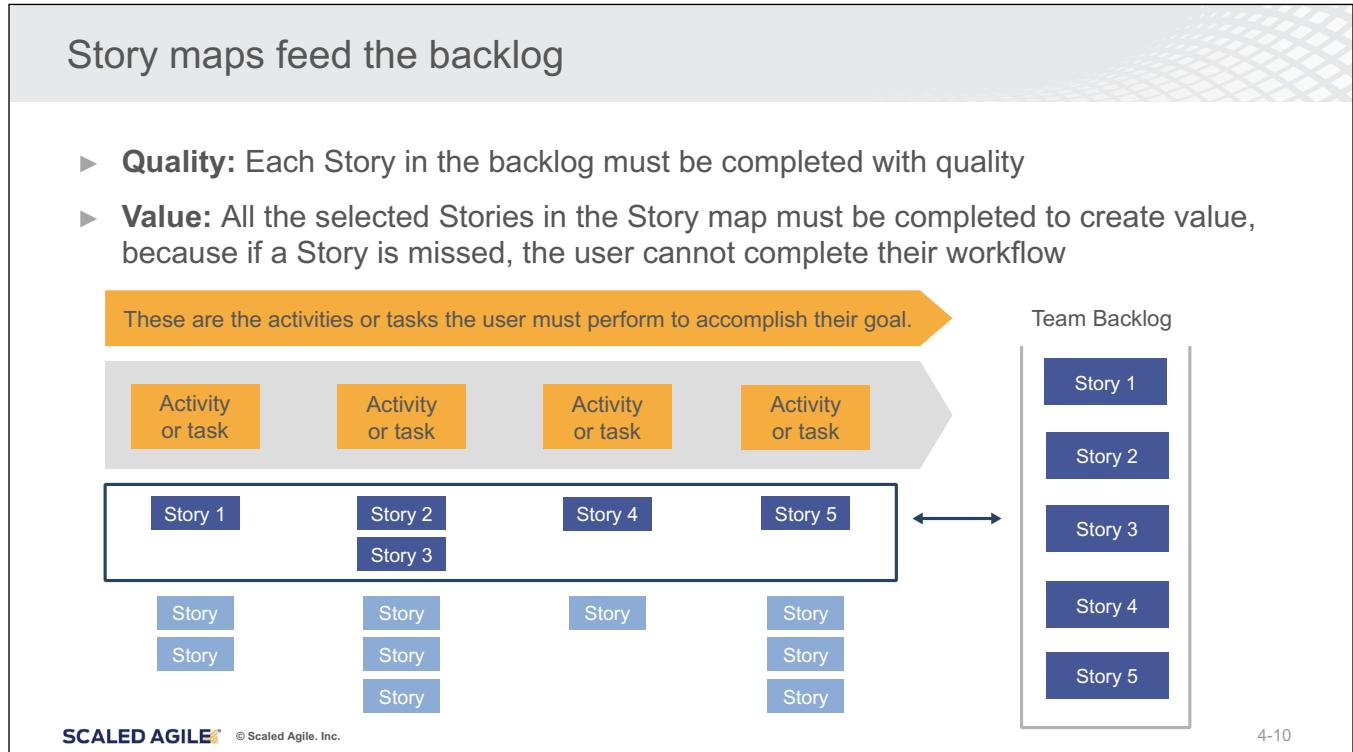
- ▶ Story maps overcome backlog challenges and promote Design Thinking
- ▶ Story maps are for ideating on how to break down big User Stories as they are told
- ▶ Story maps enable broader thinking toward delighting a user rather than starting with identifying sequence or priority
- ▶ Story maps may produce User Stories or even additional Features

## Structuring your Story map



## Story maps feed the backlog

- **Quality:** Each Story in the backlog must be completed with quality
- **Value:** All the selected Stories in the Story map must be completed to create value, because if a Story is missed, the user cannot complete their workflow

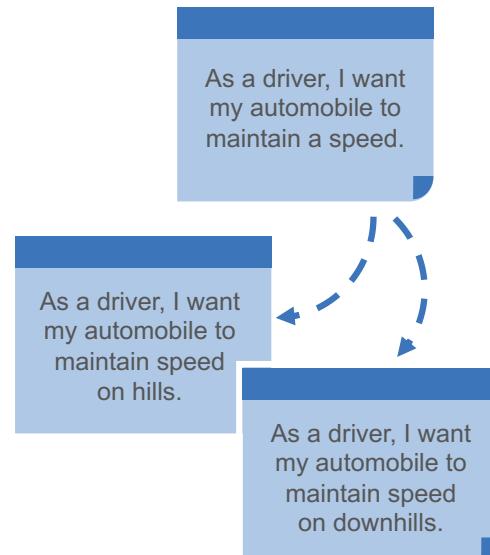


## Splitting Stories

- ▶ **In support of small batches for flow, decrease size to the minimum:**
  - Split Stories into essential and nonessential parts and eliminate the nonessential
  - Ensure you have something releasable
- ▶ **In support of feedback:**

Deploy small Stories to get technical/user feedback quickly; maximize feedback
- ▶ **In support of Iteration Planning:**

Split Stories so they fit into an Iteration



SCALED AGILE® © Scaled Agile, Inc.

4-11

## Acceptance criteria

- ▶ Provide the details of the Story from a testing point of view
- ▶ Are created by the Agile Team
- ▶ Can be written in the Given-When-Then format

**As a driver, I want** to limit the amount of money I can spend before I fuel **so that** I can control my expenses.  
**Acceptance criteria:**  
**Given** that the driver indicated a maximum amount of money  
**When** the fuel cost reaches that amount  
**Then** the fueling process stops automatically

**As a driver, I want** to get a receipt after fueling **so that** I can expense the purchase.  
**Acceptance criteria:**  
**Given** that the fueling is over  
**When** the driver asks for the receipt  
**Then** it is printed and includes:  
amount fueled, the amount paid, tax, date, time

SCALED AGILE® © Scaled Agile, Inc.

4-12



## Activity: Write User Stories



► **Step 1:** With your group, select one Feature from the ART Backlog. It can be either your own or the example provided in your workbook.

► **Step 2:** Break the Feature into at least three Stories, written in User Story format: **As a** (user role), **I want** (activity) **so that** (business value).

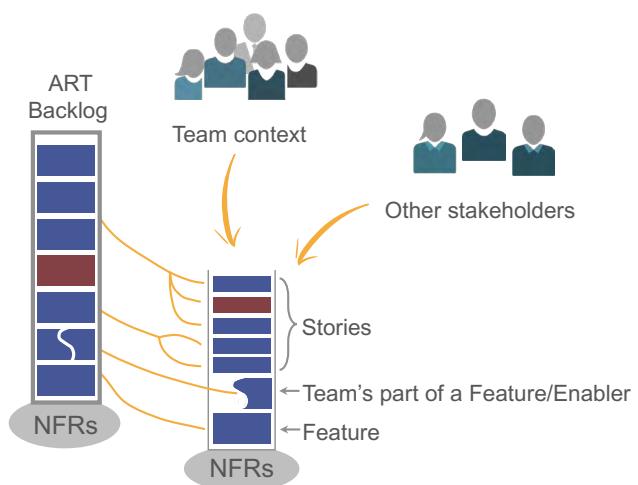
Break the Feature into Stories in a way that retains business value.

► **Step 3:** Write acceptance criteria in the Given-When-Then format for each User Story. Be sure that the acceptance criteria are testable.

► **Step 4:** Be prepared to share with the class.

## Sequencing Stories

- The PO and the team sequence Stories based on:
  - Priorities inherited from the ART Backlog
  - Capacity allocations for defects, maintenance, and refactors
  - Dependencies with other Stories, teams, events, Milestones, and releases
- Initial sequencing happens during PI Planning
- Adjustments happen at Iteration boundaries



# Write User Stories

**Instructions:** With your group, select three Features you created and decompose these Features into Stories. Write these Stories in the User Story format: As a (user role) I want (activity) so that (business value).

**Story:** As a Fleet Manager, I can search my fleet so that I can find vans that need maintenance. Vans that are overdue or need a safety recall are highlighted.

**Story:** As a Fleet Manager, I can review safety recalls so that I can prioritize the maintenance schedules of my fleet.

**As a** (user role)

**I want to** (activity)

**So that** (business value)

**As a** (user role)

**I want to** (activity)

**So that** (business value)

# ART Backlog Example Features

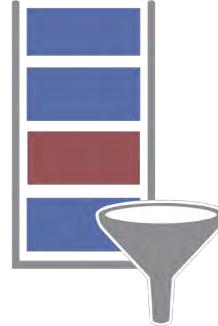
<b>Feature:</b>	<b>Flexible search</b>	
<b>Benefit:</b>	Users will have a flexible, easy-to-use search capability to locate books.	
<b>Description:</b>	Search by author, title, or genre from a single search field. Misspelling substitutions (i.e., "Did you mean ..."). Present results as per-match algorithm.	
<b>Feature:</b>	<b>Shopping Cart</b>	
<b>Benefit:</b>	Users can manage items in a shopping cart for immediate or future purchase.	
<b>Description:</b>	Users can easily access their cart from any page, view the same information displayed in the book list, change the quantity, remove it from their cart, or save it for later. A subtotal for all items in their shopping cart should be displayed at the bottom. Items saved for later should appear below that.	
<b>Feature:</b>	<b>Purchase by credit card</b>	
<b>Benefit:</b>	Users can purchase products from us (as soon as implemented—only beta up until then).	
<b>Description:</b>	Users can select from their preferred credit card and shipping address as defined in their profile or add new ones. Visa, MasterCard, Discover, and Diners Club are required. American Express is optional. Must be PCI compliant.	
<b>Feature:</b>	<b>Shipping method selection</b>	
<b>Benefit:</b>	Users can select a shipping method based on cost, delivery speed, and carrier.	
<b>Description:</b>	Users can select a shipping method based on the price, delivery speed, and estimated delivery date for all major carriers (USPS, UPS, and FedEx).	
<b>Feature:</b>	<b>Profile management</b>	
<b>Benefit:</b>	Users can create and maintain their profiles rather than enter in their information each time they order.	
<b>Description:</b>	Users can manage their login credentials (ID, password), personal information (name, email address, home address), nickname for book rating and commenting, credit card information (multiple), and shipping address (multiple). Physical addresses, email addresses, and credit card info should be verified as valid. Passwords must meet current security standards.	
<b>Feature:</b>	<b>Book detail</b>	
<b>Benefit:</b>	Users can see informative and enticing details about a book.	
<b>Description:</b>	Display book name, book cover (which can be enlarged when clicked), author and bio, book description, genre, publishing info (publisher, release date, etc.), book rating, and comments. Hyperlink author's name to a list of other books by the same author.	
<b>Feature:</b>	<b>Book list sorting</b>	
<b>Benefit:</b>	Users can sort a list of books in a number of ways to more easily find what they are looking for.	
<b>Description:</b>	Sort by book title, author, price, book rating, and release date. Allow for users to select the number of search results to appear on each page.	

## Backlog refinement

Refinement is a collaborative process that involves the entire team.

Backlog refinement:

- ▶ Helps the team turn new hypotheses into User Stories
- ▶ Allows the team to consider recent learnings before Iteration Planning or in preparation for PI Planning
- ▶ May occur daily throughout the Iteration or at a longer event on a cadence
- ▶ Improves Stories, adds acceptance criteria, and identifies missing information
- ▶ Leverages the team's collective knowledge and creativity
- ▶ Creates joint buy-in and ownership

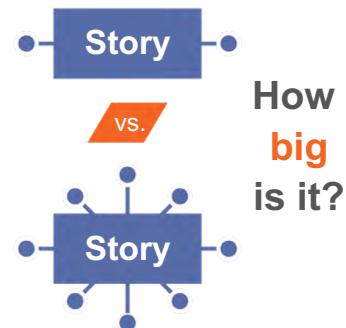


SCALDED AGILE® © Scaled Agile, Inc.

4-15

## Fibonacci is used for estimation

- ▶ A Story point is a singular number that represents:
  - Volume: How much is there?
  - Complexity: How hard is it?
  - Knowledge: What do we know?
  - Uncertainty: What's not known?
- ▶ Story points are relative. They are not connected to any specific unit of measure.
  - An 8-point Story should take 4 times longer than a 2-point Story to complete
  - Typically, a 1-point Story would take 1 day to develop and test



SCALDED AGILE® © Scaled Agile, Inc.

4-16

## Apply estimating poker for fast, relative estimating

- ▶ Estimating poker combines expert opinion, analogy, and disaggregation for quick but reliable estimates
- ▶ All members participate
- ▶ Increases accuracy by including all perspectives
- ▶ Builds understanding and creates shared understanding

Steps	
1	Each estimator gets a deck of cards
2	Read a job
3	Estimators privately select cards
4	Cards are turned over
5	Discuss differences
6	Re-estimate

*Agile Estimating and Planning* by Mike Cohn

**Warning:** Estimation performed by a manager, architect, or select group negates these benefits.

SCALED AGILE® © Scaled Agile, Inc.

4-17



## Activity: Estimate Stories



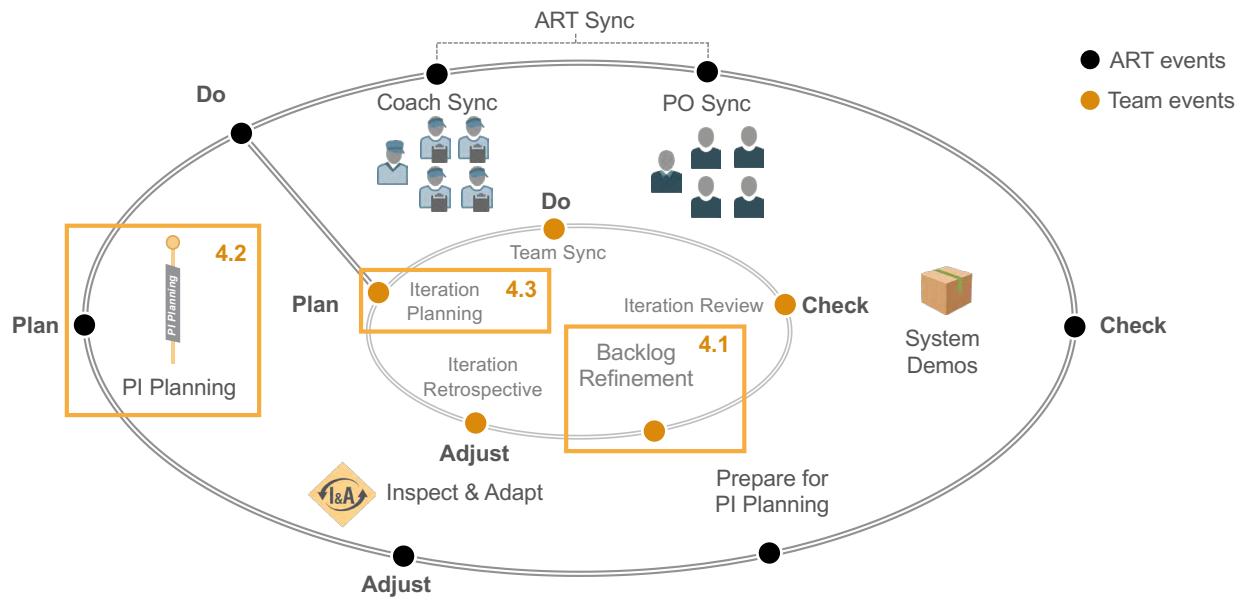
- ▶ **Step 1:** As a group, use estimating poker to relatively estimate the Stories created in “Write User Stories”
- ▶ **Step 2:** Share the following with the class:
  - Where do you find challenges when engaged in Story estimation?
  - Are you, as a team, aligned around the combination of qualities that represent a Story point: volume, complexity, knowledge, uncertainty?



SCALED AGILE® © Scaled Agile, Inc.

4-18

## ART and team planning events drive the train



## 4.2 PI Planning

## Innovation and Planning (IP) Iteration

PI Planning occurs on cadence within the IP Iteration which facilitates reliability, PI readiness, planning, and innovation.

- ▶ **Innovation:** Opportunity for innovation, hackathons, and infrastructure improvements
  - ▶ **Planning:** Provides for cadence-based planning
  - ▶ The IP Iteration also provides an estimating **guard band** for cadence-based delivery.

“Provide sufficient capacity margin to enable cadence.”

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

© Scaled Agile, Inc.

4-21

## Example IP Iteration calendar

© Scaled Agile, Inc.

4-22



## Video: The Power of PI Planning

Duration  
3 min



<https://bit.ly/Video-PowerofPIPlanning>

SCALED AGILE® © Scaled Agile, Inc.

4-23

## What is PI Planning?

Planning Interval (PI) Planning is a cadence-based event that serves as the heartbeat of the ART, aligning all teams on the ART to a shared mission and Vision.

- ▶ Two days every 8 – 12 weeks (10 weeks is typical)
- ▶ Everyone plans together
- ▶ Product Management owns Feature priorities
- ▶ Development teams own Story planning and high-level estimates
- ▶ Architect and UX work as intermediaries for governance, interfaces, and dependencies

SCALED AGILE® © Scaled Agile, Inc.

4-24

## The benefits of PI Planning

- ▶ Establishes personal communication across all team members and stakeholders
- ▶ Aligns development of business goals with the business context, Vision, and Team/ART PI Objectives
- ▶ Identifies dependencies and fosters cross-team and cross-ART collaboration
- ▶ Provides the opportunity for the right amount of architecture and Lean UX guidance
- ▶ Matches demand to capacity, eliminating excess work in process (WIP)
- ▶ Allows for faster decision-making

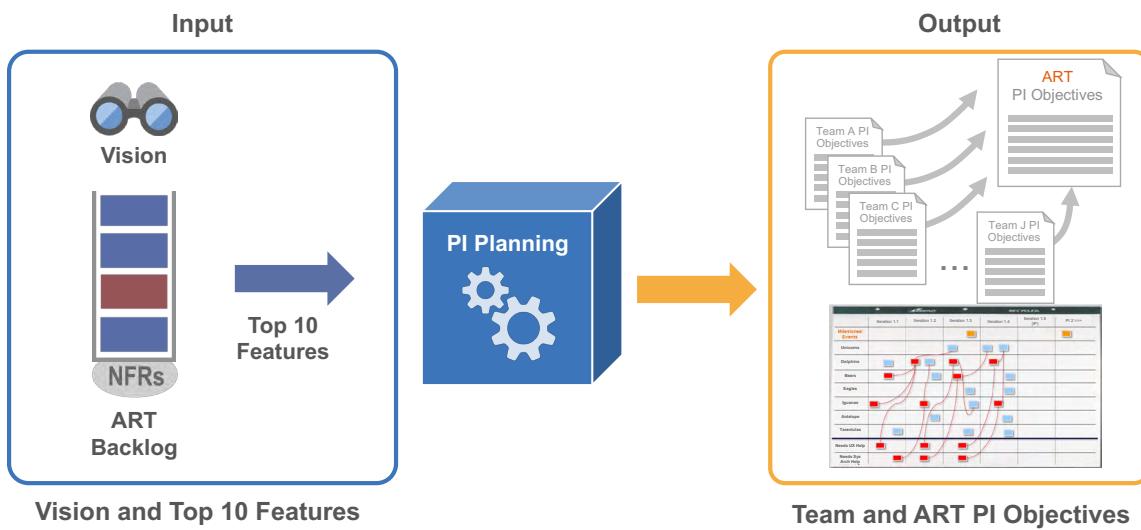


Cross-team collaboration

SCALED AGILE® © Scaled Agile, Inc.

4-25

## The PI Planning process



SCALED AGILE® © Scaled Agile, Inc.

4-26

## Create alignment with PI Objectives

- ▶ Objectives are business summaries of what each team intends to deliver in the upcoming PI
- ▶ They often directly relate to intended Features in the backlog
- ▶ Other examples:
  - Aggregation of a set of Features
  - A Milestone like a trade show
  - An Enabler Feature supporting the implementation
  - A major refactoring

Objectives for PI 1	BV	AV
1. Show routing calculations between the 5 most frequent destinations	—	—
2. Navigate autonomously from distribution center to the most frequent destination	—	—
3. Parallel park for a delivery	—	—
4. Return to the distribution center after delivery	—	—
5. Include traffic data in route planning	—	—
6. Recall a delivery that is already in progress	—	—
<b>Uncommitted Objectives</b>		
7. Spike: Reduce GPS signal loss by 25%	—	—
8. Demonstrate real-time rerouting to avoid delays (e.g., accident, construction)	—	—

SCALED AGILE® © Scaled Agile, Inc.

4-27

## Maintain predictability with uncommitted objectives

Uncommitted objectives help improve the predictability of delivering business value.

- ▶ They are planned, not extra things teams do "just in case you have time"
- ▶ They are not included in the commitment, thereby making the commitment more reliable
- ▶ If a team has low confidence in meeting a PI Objective, it should be moved to uncommitted
- ▶ If an objective has many unknowns, consider moving it to uncommitted and putting in early spikes\*
- ▶ Uncommitted objectives count when calculating load

Objectives for PI 1	
—	
—	
—	
—	
—	
<b>Uncommitted Objectives</b>	
7. Spike: Reduce GPS signal loss by 25%	
8. Demonstrate real-time rerouting to avoid delays (e.g., accident, construction)	

\*Spikes are research Stories, considered exploration-style Enablers.

SCALED AGILE® © Scaled Agile, Inc.

4-28

## Outcomes of the PI Planning simulation

Actively participating in a simulated PI Planning event will enable you to:



### Communication

Experience the business benefits of establishing communication across all team members and stakeholders



### Estimate Capacity

Experience estimating capacity for the Iteration



### Objectives

Experience drafting PI Objectives for achieving the PI and committing to the plan



### Manage risks

Experience managing PI Risks



## Activity: Identify ART roles

Duration  
3 min

- ▶ **Step 1:** Identify ART roles for the simulation
- ▶ **Step 2:** Ensure that you have all key roles required for the PI Planning simulation

Simulation role	Assigned to
Executive	Volunteer
Product Manager	Volunteer
System Architect, UX, Development Manager	Volunteer
RTE	Trainer



## Simulation: Why are we here?



### Alignment to a common mission

We are here to gain alignment and commitment around a clear set of prioritized objectives. I will now review the agenda for the next two days of the PI Planning event.



## Simulation: Day 1 agenda

<b>Business context</b>	8:00 – 9:00	<ul style="list-style-type: none"><li>State of the business</li></ul>
<b>Product/Solution Vision</b>	9:00 – 10:30	<ul style="list-style-type: none"><li>Vision and prioritized Features</li></ul>
<b>Architecture Vision and development practices</b>	10:30 – 11:30	<ul style="list-style-type: none"><li>Architecture, common frameworks, and so on</li><li>Agile tooling, engineering practices, and so on</li></ul>
<b>Planning context and lunch</b>	11:30 – 1:00	<ul style="list-style-type: none"><li>Facilitator explains the planning process</li></ul>
<b>Team breakouts</b>	1:00 – 4:00	<ul style="list-style-type: none"><li>Teams develop draft plans and identify risks and impediments</li><li>Architects and Product Managers circulate</li></ul>
<b>Draft plan review</b>	4:00 – 5:00	<ul style="list-style-type: none"><li>Teams present draft plans, risks, and impediments</li></ul>
<b>Management review and problem-solving</b>	5:00 – 6:00	<ul style="list-style-type: none"><li>Adjustments made based on challenges, risks, and impediments</li></ul>



## Simulation: Day 2 agenda

<b>Planning adjustments</b>	8:00 – 9:00	<ul style="list-style-type: none"><li>Planning adjustments made based on previous day's management meeting</li></ul>
<b>Team breakouts</b>	9:00 – 11:00	<ul style="list-style-type: none"><li>Teams develop final plans and refine risks and impediments</li><li>Business Owners circulate and assign business value to team objectives</li></ul>
<b>Final plan review and lunch</b>	11:00 – 1:00	<ul style="list-style-type: none"><li>Teams present final plans, risks, and impediments</li></ul>
<b>PI risks</b>	1:00 – 2:00	<ul style="list-style-type: none"><li>Remaining PI-level risks are discussed and ROAMed</li></ul>
<b>PI confidence vote</b>	2:00 – 2:15	<ul style="list-style-type: none"><li>Team and ART confidence vote</li></ul>
<b>Plan rework if necessary</b>	2:15 – ???	<ul style="list-style-type: none"><li>If necessary, planning continues until commitment is achieved</li></ul>
<b>Planning Retrospective and moving forward</b>	After commitment	<ul style="list-style-type: none"><li>Retrospective</li><li>Moving forward</li><li>Final instructions</li></ul>



## Simulation: Briefings



**Executive**



**Product Management**



**System Architect**



## Simulation: Planning guidance

Expect this first PI Planning to feel a bit chaotic. Future PI Planning meetings will become more routine.

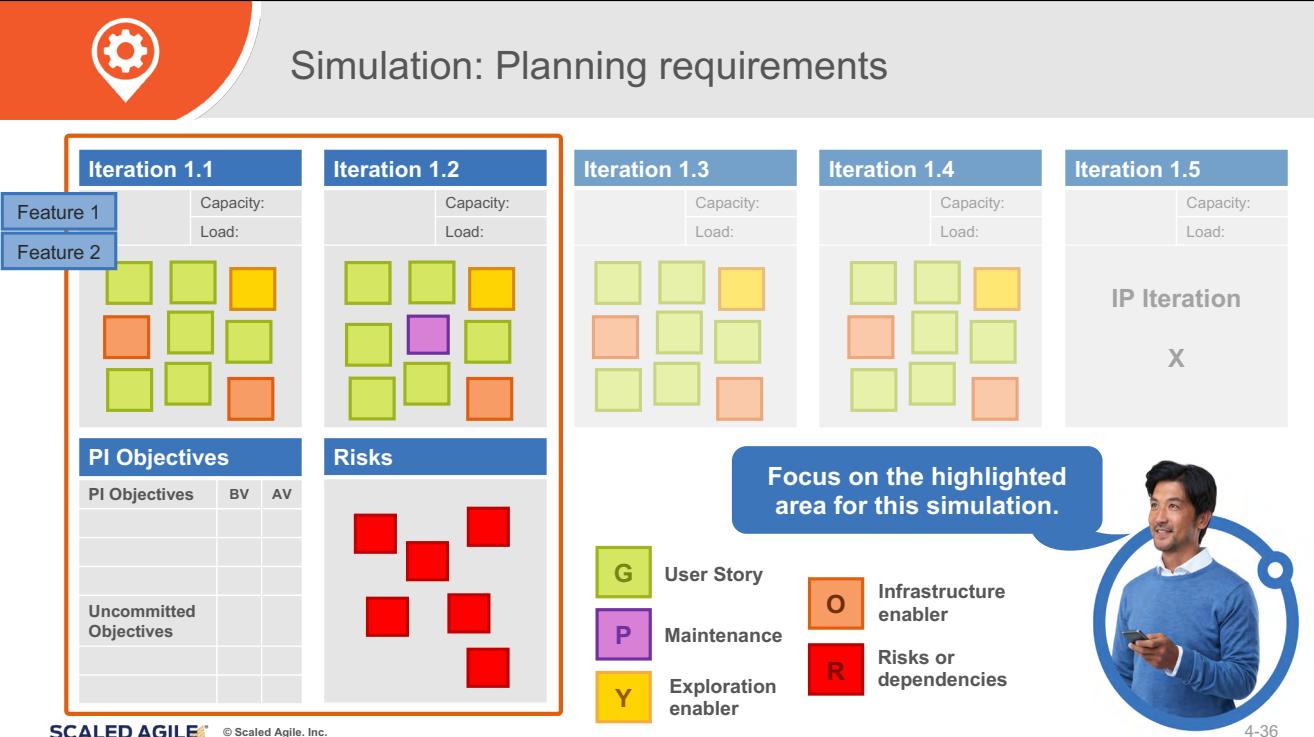
**PO:** You are responsible for making decisions at the User Story level by leveraging your content authority

**SM/TC:** You are responsible for managing the timebox, the dependencies, and the ambiguities

**Agile Team:** You are responsible for defining User Stories, planning them into the Iteration, and working out interdependencies with other teams

**SCALED AGILE®** © Scaled Agile, Inc.

4-35



## Simulation: Planning requirements

**Iteration 1.1**

Feature 1	Capacity: Load:
Feature 2	

**PI Objectives**

PI Objectives	BV	AV

**Uncommitted Objectives**

**Iteration 1.2**

Capacity: Load:

**Risks**

**Iteration 1.3**

Capacity: Load:

**Iteration 1.4**

Capacity: Load:

**Iteration 1.5**

Capacity: Load:
IP Iteration X

**Focus on the highlighted area for this simulation.**

**Legend:**

- G User Story
- P Maintenance
- Y Exploration enabler
- O Infrastructure enabler
- R Risks or dependencies

**SCALED AGILE®** © Scaled Agile, Inc.

4-36



## Activity: Select the Feature from Product Management

Duration  
5 min

- ▶ **Step 1:** Work with your team to select a Feature from Product Management
- ▶ **Step 2:** Ensure this information is visible so it can be referred to during the PI Planning simulation

SCALED AGILE® © Scaled Agile, Inc.

4-37



## Simulation: Using historical data to calculate velocity

Establish velocity by looking at the average output of the last Iterations

### Velocity

6 Iterations

→ 180 Story points

→ 30 SP / Iteration

SCALED AGILE® © Scaled Agile, Inc.

4-38



## Simulation: Calculate your capacity

Calculating Iteration capacity:

- ▶ For every full-time Agile Team member contributing to Solution development, give the team 8 points; adjust for those who are part-time.
- ▶ Subtract 1 point for every team member's vacation day and holiday.
- ▶ Find a small Story that would take about a half-day to develop and a half-day to test and validate. Call it a 1-point Story.
- ▶ Estimate every other Story relative to that 1.

### Example:

A seven-person team composed of three developers, two testers, one PO, and one SM/TC

Exclude the PO, SM/TC, and time off from the calculation

Calculated capacity:  
 $5 \times 8 \text{ points} = 40 \text{ points per Iteration}$

SCALED AGILE® © Scaled Agile, Inc.

4-39



## Activity: Team breakout #1

Duration  
50 min

You will be planning a short PI with two Iterations with your team.

- ▶ **Step 1:** Calculate and enter the capacity for each Iteration (two-week Iteration)
  - The first Iteration starts Monday
  - Use your real availability
- ▶ **Step 2:** Estimate the Stories using Story points
- ▶ **Step 3:** Load the Stories into the Iterations
- ▶ **Step 4:** Write the PI Objectives using clear statements
- ▶ **Step 5:** Identify the uncommitted objectives
- ▶ **Step 6:** Identify any ART PI Risks and dependencies



SCALED AGILE® © Scaled Agile, Inc.

4-40



## Activity: Coach Sync



- ▶ **Step 1:** The teams observe the Coach Sync, conducted by the RTE.
- ▶ **Step 2:** Choose a SM/TC to provides the team's current status and address the questions from the RTE.
- ▶ **Step 3:** The RTE holds a meet-after following the sync (limited to one or two topics for the simulation).

**Note:** Coach Sync questions are on the following slide.



## Activity: Coach Sync

Coach Sync Questions	Team 1
Have you identified the capacity for each Iteration of the PI?	
Have you identified most of the Stories for the first two Iterations and begun estimating?	
Have you begun resolving dependencies with other teams?	
Are you discussing tradeoffs and conflicting priorities with your Business Owners?	
Have you identified any ART PI Risks?	
Will you be ready to start writing PI Objectives in the next 15 minutes?	
Is there anything you need to discuss with other SM/TCs? If so, stay for the meet-after.	



## Activity: Draft plan review

Duration  
7 min

- ▶ **Step 1:** The teams will present summaries of their first two Iterations and one or more draft PI Objectives.
- ▶ **Step 2:** Make sure to include the following:
  - Capacity and load for each Iteration
  - Draft PI Objectives
  - ART PI Risks and impediments

## Management review and problem-solving

At the end of Day 1, management meets to make adjustments to scope and objectives based on the day's planning.

### Common questions:

- What did we just learn?
- Where do we need to adjust: in Vision, in scope, in team assignments?
- Where are the bottlenecks?
- What Features must be de-scoped?
- What decisions must we make between now and tomorrow to address these issues?



Photo of management review. Photo courtesy of Hybris Software

## Activities during Day 2

Day 1		Day 2	
<b>Business context</b>	8:00–9:00	<b>Planning adjustments</b>	8:00–9:00
<b>Product / Solution Vision</b>	9:00–10:30	<b>Team breakouts</b>	9:00–11:00
<b>Architecture Vision and development practices</b>	10:30–11:30	<b>Final plan review and lunch</b>	11:00 –1:00
<b>Planning context and lunch</b>	11:30–1:00	<b>ART PI Risks</b>	1:00–2:00
<b>Team breakouts</b>	1:00–4:00	<b>PI confidence vote</b>	2:00–2:15
<b>Draft plan review</b>	4:00–5:00	<b>Plan rework if necessary</b>	2:15–???
<b>Management review and problem-solving</b>	5:00–6:00	<b>Planning Retrospective and moving forward</b>	After commitment

SCALED AGILE® © Scaled Agile, Inc. 4-45

## Make planning adjustments

- ▶ Based on the previous day's management review and problem-solving meeting, adjustments are discussed
- ▶ Possible changes:
  - Business priorities
  - Adjustment to Vision
  - Changes to scope
  - Realignment of work and teams



## Team breakout #2

Based on new knowledge and a good night's sleep, teams work to create their final plans.

- ▶ In the second team breakout, Business Owners circulate and assign business value to PI Objectives from low (1) to high (10)
- ▶ Teams finalize the PI plan
- ▶ Teams also consolidate PI Risks, impediments, and dependencies
- ▶ Uncommitted objectives provide the capacity and guard band needed to increase the reliability of cadence-based delivery

Objectives for PI 1	BV	AV
1. Show routing calculations between the 5 most frequent destinations	10	
2. Navigate autonomously from distribution center to the most frequent destination	8	
3. Parallel park for a delivery	7	
4. Return to the distribution center after delivery	10	
5. Include traffic data in route planning	7	
6. Recall a delivery that is already in progress	7	
<b>Uncommitted Objectives</b>		
7. Spike: Reduce GPS signal loss by 25%	2	
8. Demonstrate real-time rerouting to avoid delays (e.g., accident, construction)	5	



## Activity: Setting business value

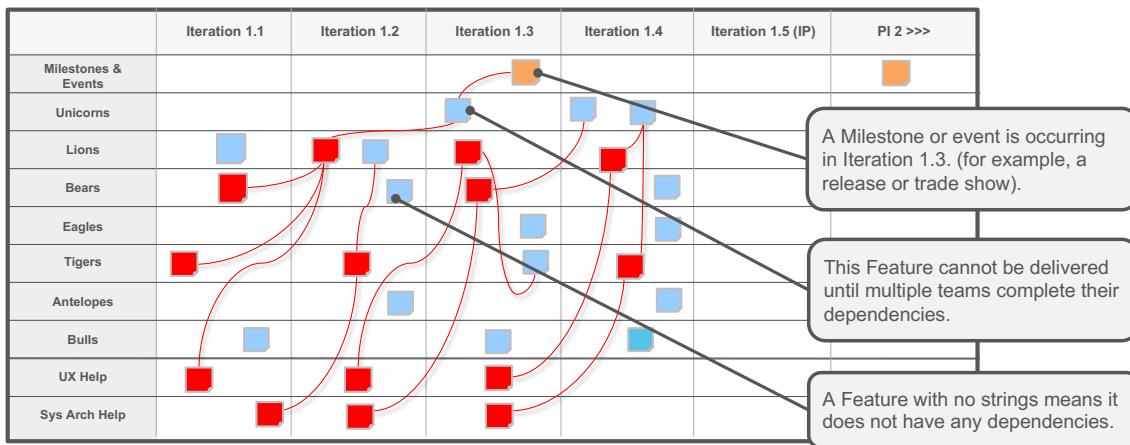
Duration  
7 min

The trainer will demonstrate assigning business value for one team's objectives.

- ▶ **Step 1:** Bring the Business Owners to one team's draft plans
- ▶ **Step 2:** The Business Owners will set value on a scale of 1 – 10 for each identified objective
- ▶ **Step 3:** Observe the discussion that takes place, illustrating the larger purposes and thought processes around assigning business value

Objectives for PI	BV	AV
1. Show routing calculations between the 5 most frequent destinations	10	
2. Navigate autonomously from distribution center to the most frequent destination	8	
3. Parallel park for a delivery	7	
4. Return to the distribution center after delivery	10	
5. Include traffic data in route planning	7	
6. Recall a delivery that is already in progress	7	
<b>Uncommitted Objectives</b>		
7. Spike: Reduce GPS signal loss by 25%	2	
8. Demonstrate real-time rerouting to avoid delays (e.g., accident, construction)	5	

## ART Planning Board: Feature delivery, dependencies, and Milestones



ART Planning Board Legend:



Red strings, or lines for digital boards, are used to connect a Feature or Milestone to one or more dependencies. Sometimes a dependency has its own dependency (see Lions in Iteration 1.2).

**SCALED AGILE®** © Scaled Agile, Inc.

4-49

## Final plan review

Teams and Business Owners review all final plans.

### Final plan review agenda

1. Changes to capacity and load
2. Final PI Objectives with business value
3. PI Risks and impediments
4. Q&A session



**SCALED AGILE®** © Scaled Agile, Inc.

4-50

## Building the final plan

- ▶ Final plans are reviewed by all teams
- ▶ Business Owners are asked whether they accept the plan
  - If so, the plan is accepted
  - If not, the plans stay in place, and the team continues planning after the review



Photo of final plan presentation. Photo courtesy of Discount Tire Corporation.

## Addressing ART PI Risks

After all plans have been presented, remaining PI Risks and impediments are discussed and categorized.

### ROAMing risks:

- ▶ **Resolved:** Has been addressed; No longer a concern
- ▶ **Owned:** Someone has taken responsibility
- ▶ **Accepted:** Nothing more can be done; If risk occurs, release may be compromised.
- ▶ **Mitigated:** Team has plans to adjust as necessary.





## Activity: Manage ART PI Risks



The trainer will assess one to two ROAMing risks for one team.

- ▶ **Step 1:** Pick two risk examples.
- ▶ **Step 2:** Read them in front of all teams and stakeholders.
- ▶ **Step 3:** Ask if anyone can own, mitigate, or resolve the risk. Otherwise, accept the risk as is.
- ▶ **Step 4:** Put each risk into its corresponding quadrant of the ROAM sheet for the ART.



SCALDED AGILE® © Scaled Agile, Inc.

4-53

## Confidence vote: Team and ART

Once ART PI Risks have been addressed, a confidence vote is taken by the team and ART.

### A commitment with two parts:

1. Teams agree to do everything in their power to meet the agreed-to objectives
2. If fact patterns dictate that it is simply not achievable, teams agree to escalate immediately so that corrective action can be taken



No confidence



Little confidence



Good confidence



High confidence



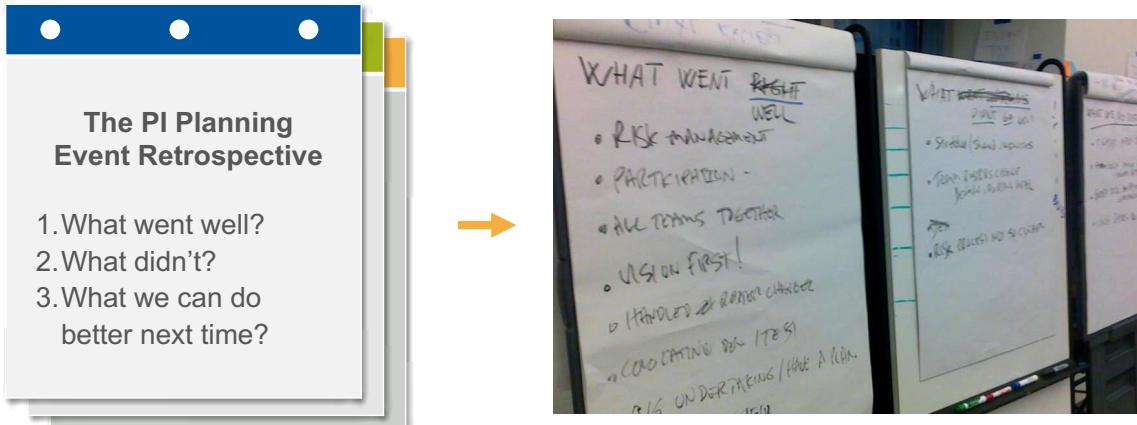
Very high confidence

SCALDED AGILE® © Scaled Agile, Inc.

4-54

## Run a planning meeting retrospective

The PI Planning event will evolve over time. Ending with a retrospective will help continuously improve it.



SCALED AGILE® © Scaled Agile, Inc.

4-55

## 4.3 Team Planning

SCALED AGILE® © Scaled Agile, Inc.

4-56

## SAFe Scrum – Plan and commit with Iteration Planning

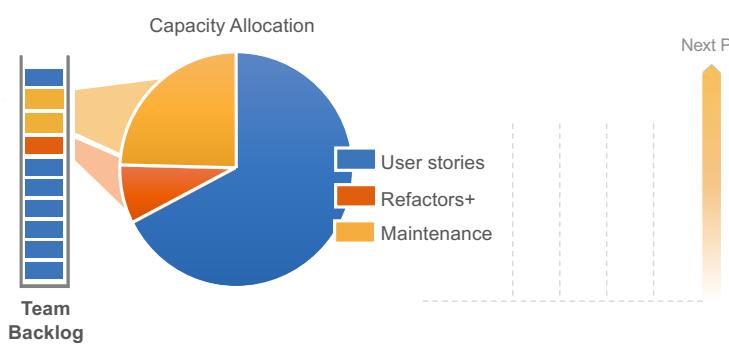
Purpose	<ul style="list-style-type: none"><li>Define and commit to what will be built in the Iteration</li></ul>
Responsibilities	<ul style="list-style-type: none"><li>The PO defines the potential value of the Iteration</li><li>The team defines how the value will be delivered via Stories that meet the definition of done</li></ul>
Result	<ul style="list-style-type: none"><li>Iteration Goals and a backlog of the team's commitment</li></ul>
Reciprocal commitment	<ul style="list-style-type: none"><li>Team commits to delivering specific value</li><li>Stakeholders commit to leaving priorities unchanged during the Iteration</li></ul>

SCALED AGILE® © Scaled Agile, Inc.

4-57

## Capacity allocation for a healthy balance

- By having capacity allocation defined, the PO doesn't need to prioritize unlike things against each other
- Once the capacity allocation is set, the PO and team can prioritize like things against each other



### Capacity allocation

- Helps alleviate velocity degradation due to technical debt
- Keeps existing Customers happy with bug fixes and enhancements
- Can change at Iteration or PI boundaries

SCALED AGILE® © Scaled Agile, Inc.

4-58

## Commit to the Iteration goals

Team commitments are not just to the work. Teams are committed to other teams, the ART, and the stakeholders.

### A team meets its commitment:

By doing everything they said they would do,

- OR -

By immediately raising the concern if it isn't feasible to do so.

#### Commitment

Too rigid of a commitment can lead to burnout, inflexibility, and quality problems.



#### Adaptability

Too little commitment can lead to unpredictability and lack of focus on results.

## Iteration Goals: Examples

#### Software example

##### Iteration Goals

1. Finalize and push last-name search and first-name morphology
2. Index 80% of remaining data
3. Other Stories:
  - Establish search replication validation protocol
  - Refactor artifact dictionary schema

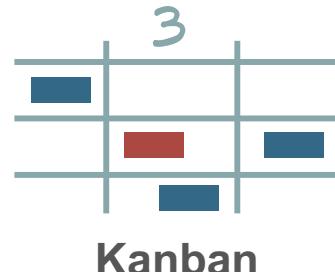
#### Business example

##### Iteration Goals

1. Roll out the incident report procedures
2. Have documentation in one folder for external audit
3. Obtain commitment to audit days from auditors and internal leaders

## Team Planning for SAFe Team Kanban

- ▶ Some teams have a more responsive nature to their work, such as maintenance teams and system teams
- ▶ These teams will tend toward emphasizing continual backlog refinement over detailed Iteration plans
- ▶ Kanban teams could still publish Iteration goals and integrate with other teams continuously or on cadence
- ▶ They commit to the goals and communicate average response time for incoming work based on their known historical lead time, enabling other teams to accurately depend on them
- ▶ They participate in PI Planning, System Demos, and Inspect & Adapt like all other teams



Kanban



## Action Plan: Plan the work



- ▶ **Step 1:** As a group, brainstorm four or more questions you can ask to help uncover the volume, complexity, knowledge, or uncertainty of upcoming work
- ▶ **Step 2:** Make a plan for how you will bring these questions into PI Planning and Iteration Planning
- ▶ **Step 3:** Add your plan to the Action Plan in your workbook





# Action Plan

## Plan the Work

## Learning review

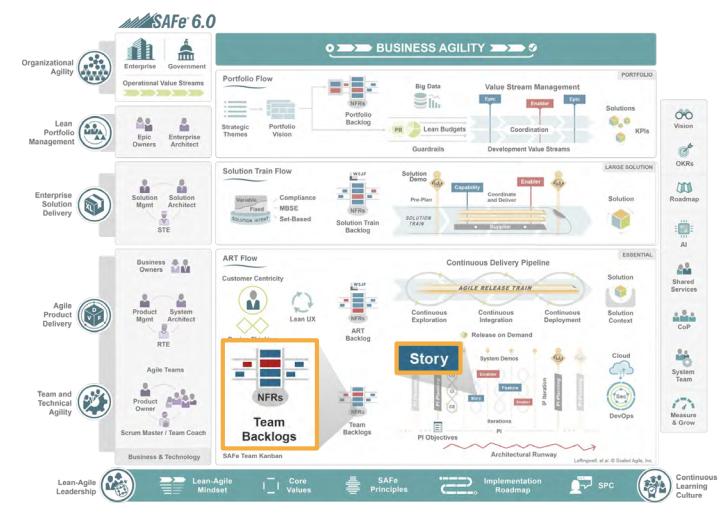
In this lesson you:

- ▶ Explored techniques for creating a Team Backlog
- ▶ Practiced preparing your backlog by breaking down Features into Stories
- ▶ Applied strategies for creating well-written Stories
- ▶ Discovered relative sizing of User Stories
- ▶ Experienced the steps of PI Planning
- ▶ Discovered the purpose of Iteration Goals and PI Objectives
- ▶ Explored the Iteration Planning event and its outcomes
- ▶ Created a set of questions your team can use to understand and size upcoming work

## Articles used in this lesson

Read these Framework articles to learn more about topics covered in this lesson

- ▶ "Story"  
<https://www.scaledagileframework.com/story/>
- ▶ "Team Backlogs"  
<https://www.scaledagileframework.com/team-backlog/>
- ▶ "PI Planning"  
<https://www.scaledagileframework.com/pi-planning/>
- ▶ "Iteration Planning"  
<https://www.scaledagileframework.com/iteration-planning/>



## Continue your SAFe journey with the following resources:

Access the “SAFe ART and Team Events” page on SAFe Studio for additional tools and guidance for preparing to facilitate SAFe events. <a href="https://bit.ly/Community-SAFeARTandTeamEvents">https://bit.ly/Community-SAFeARTandTeamEvents</a>	Download and read the “Story Writing and Splitting Guide” to learn what makes a good Story and how to split Stories. <a href="https://bit.ly/Community-StoryWritingGuide">https://bit.ly/Community-StoryWritingGuide</a>
Practice Story splitting with your team using the SAFe Collaborate template, “Story Splitting on an Agile Team.” <a href="https://bit.ly/Template-StorySplitting">https://bit.ly/Template-StorySplitting</a>	Download the “SAFe Iteration Execution Toolkit” for a set of tools and guides for facilitating significant Iteration events. <a href="https://bit.ly/Community-ToolkitsandTemplates">https://bit.ly/Community-ToolkitsandTemplates</a>
Download “The Facilitator’s Guide to SAFe: Backlog Refinement” for support with agenda setting, preparation checklists, and tips and tricks for facilitating great backlog refinement events. <a href="https://bit.ly/Community-FGBacklogRefinement">https://bit.ly/Community-FGBacklogRefinement</a>	Watch this quick one-minute video, <i>SAFe Developer Stories: My First PI Planning</i> , to hear about Elizabeth Flournoy’s first PI Planning experience and how it’s affected how she approaches her work. <a href="https://bit.ly/Video-FirstPIPlanning">https://bit.ly/Video-FirstPIPlanning</a>

## References

Cohn, Mike. *Agile Estimating and Planning*. Pearson Education, Inc.: Upper Saddle River, 2006. 56-59.

Reinertsen, Donald G. *The Principles of Product Development Flow: Second Generation of Lean Product Development*. Redondo Beach: Celeritas 2009. 178.

## Lesson notes

Enter your notes below. If using a digital workbook, save your PDF often so you don't lose any of your notes.

# Lesson 5

## Deliver Value

SAFe® Course - Attending this course gives learners access to the SAFe® Practitioner exam and related preparation materials.



SCALED AGILE®

### Lesson Topics

- 5.1** Continuously integrate, deploy, and release
- 5.2** Syncs align delivery
- 5.3** Building quality in



## Learning objectives

At the end of this lesson, you should be able to:

- ▶ Explain the importance of a Continuous Delivery Pipeline (CDP)
- ▶ Describe the communication and synchronization benefits of SAFe sync events
- ▶ Summarize different practices to build quality into Solutions
- ▶ Create action items as a team to take into the upcoming PI around improving your Continuous Delivery Pipeline and Built-in Quality

## 5.1 Continuously integrate, deploy, and release

## SAFe Lean-Agile Principles

#1 Take an economic view

#2 Apply systems thinking

#3 Assume variability; preserve options

### #4 Build incrementally with fast, integrated learning cycles

#5 Base milestones on objective evaluation of working systems

#6 Make value flow without interruptions

#7 Apply cadence, synchronize with cross-domain planning

#8 Unlock the intrinsic motivation of knowledge workers

#9 Decentralize decision-making

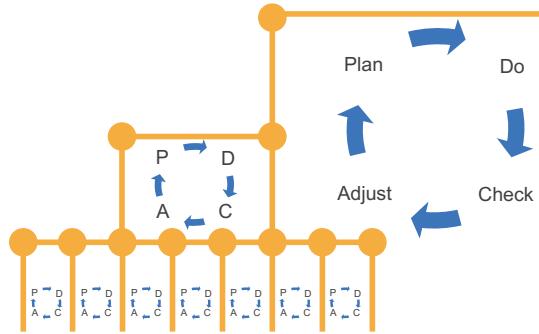
#10 Organize around value

## Integration points control product development

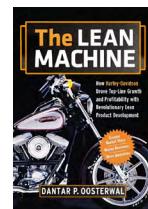
“Product development is the creation of reusable knowledge through set-based design and the establishment of development cadence and flow.”

—Dantar P. Oosterwal, *The Lean Machine*

- ▶ Integration points accelerate learning
- ▶ Development can proceed no faster than the slowest learning loop
- ▶ Improvement comes through synchronization of design loops and faster learning cycles



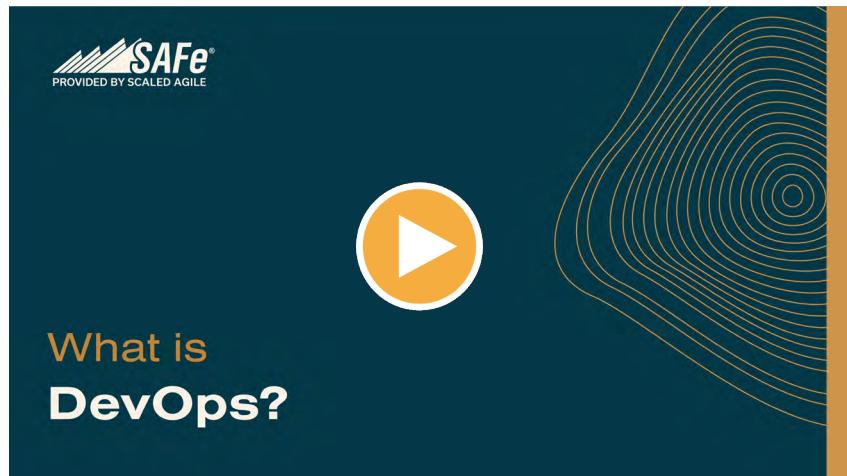
*The Lean Machine* by  
Dantar P. Oosterwal





## Video: What is DevOps?

Duration  
5 min

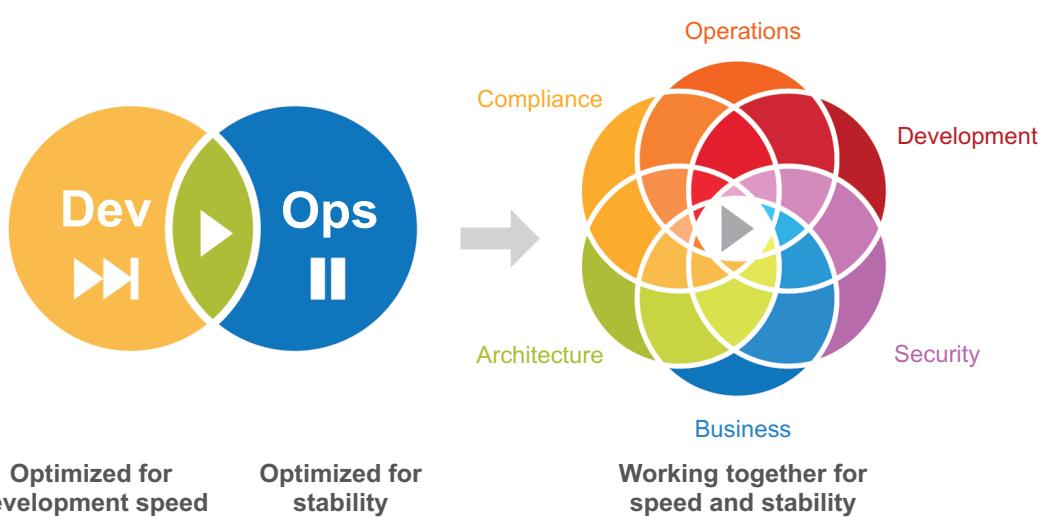


<https://bit.ly/Video-WhatIsDevOps>

SCALED AGILE® © Scaled Agile, Inc.

5-7

Maximize speed *and* stability



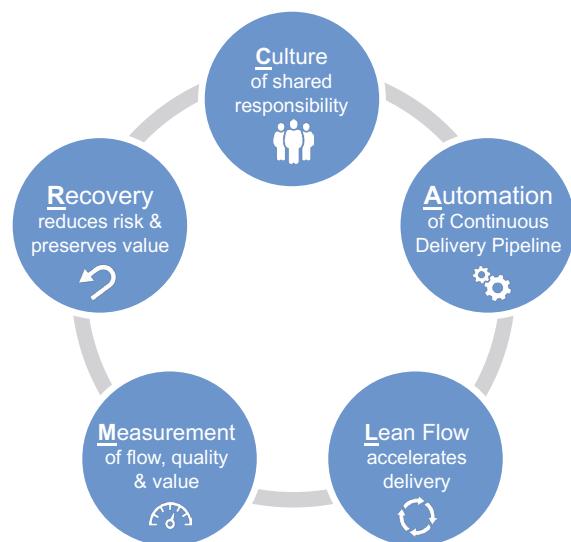
SCALED AGILE® © Scaled Agile, Inc.

5-8

## Maximize speed and stability

C	<b>Culture</b>	Establish a culture of shared responsibility for development, deployment, and operations.
A	<b>Automation</b>	Automate the CDP.
L	<b>Lean flow</b>	Keep batch sizes small, limit WIP, and provide extreme visibility.
M	<b>Measurement</b>	Measure the flow through the pipeline. Implement full-stack telemetry.
R	<b>Recovery</b>	Design for low-risk releases. Establish fast recovery, fast reversion, and fast fix-forward.

## A CALMR approach to DevOps

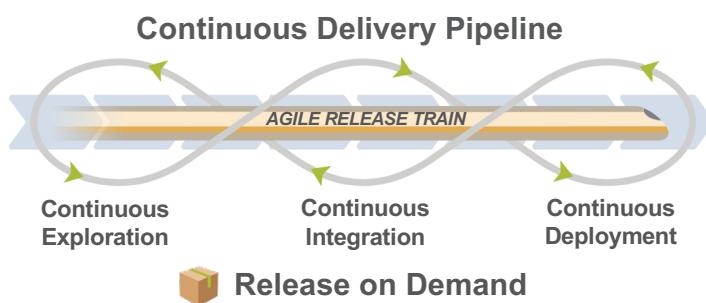


SCALED AGILE® © Scaled Agile, Inc.

5-9

## Building the CDP with DevOps

- The CDP represents the workflows, activities, and automation needed to deliver new functionality more frequently
- Each ART builds and maintains, or shares a pipeline
- Organizations map their current pipeline into this new structure to remove delays and improve the efficiency of each step



SCALED AGILE® © Scaled Agile, Inc.

© Scaled Agile, Inc.

5-10



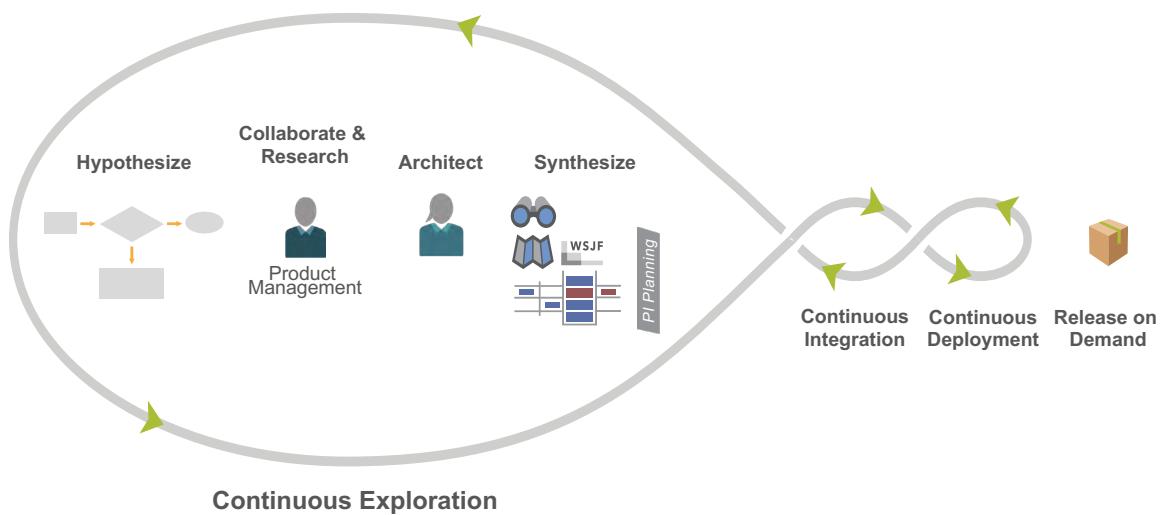
## Discussion: Continuous delivery culture



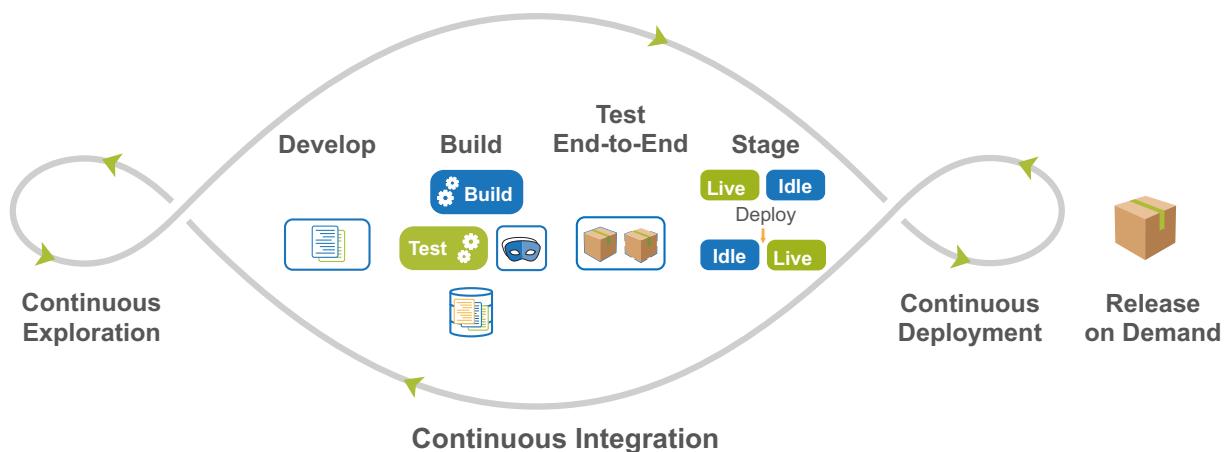
- ▶ **Step 1:** Working in your group, discuss the following:
  - Is your organizational culture or environment ready for continuous delivery?
  - What are the biggest factors influencing your readiness?
  - What does “continuous” mean to you and your group?
- ▶ **Step 2:** Be prepared to share with the class



## Continuous Exploration – Understand Customer needs



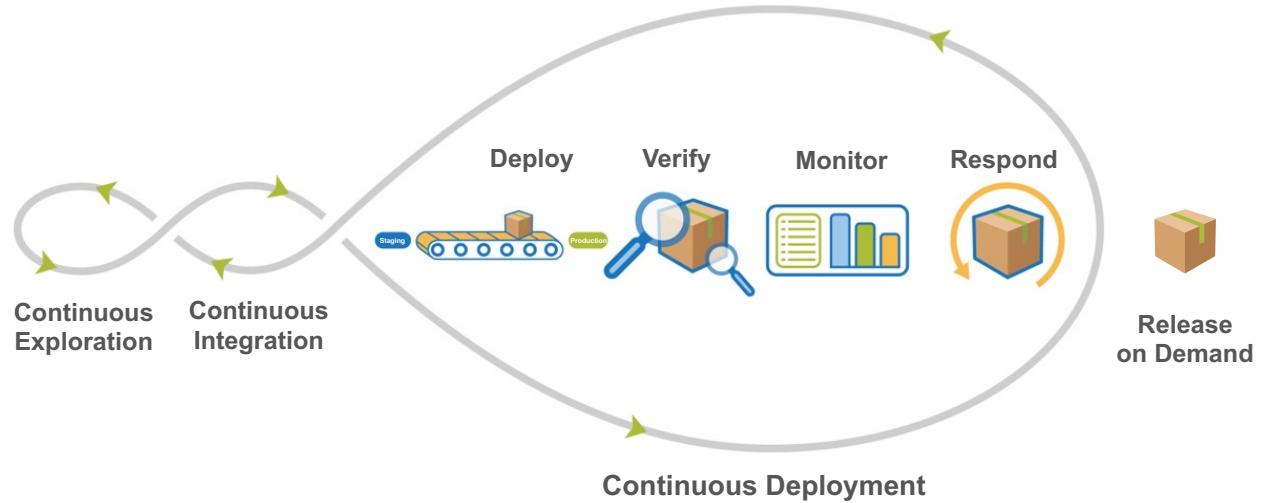
## Continuous Integration – Get fast feedback on small changes



SCALED AGILE® © Scaled Agile, Inc.

5-13

## Continuous Deployment – Get to production early for verification

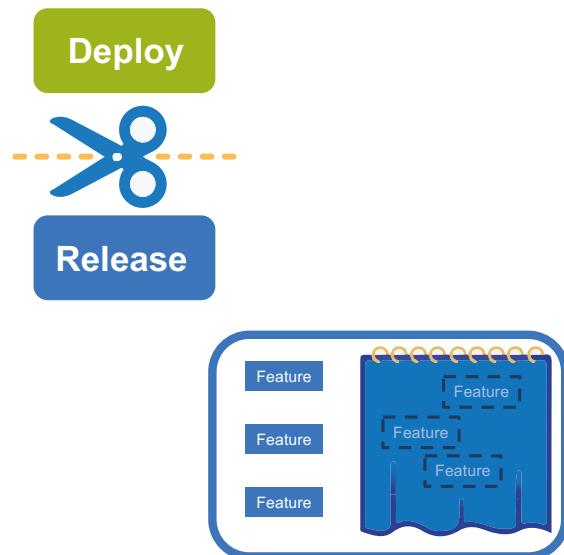


SCALED AGILE® © Scaled Agile, Inc.

5-14

## Release on Demand

- ▶ Separate deployment from release
- ▶ Hide new functionality under feature toggles
- ▶ Enable testing background and foreground processes in the actual production environment before exposing new functionality to users
- ▶ Consider timing of the release a business decision



SCALED AGILE® © Scaled Agile, Inc.

5-15

## Discussion: Continuous Exploration, Integration, and Deployment challenges

- ▶ **Step 1:** Considering the various aspects of environment, culture, tools, and people that influence continuous delivery culture, discuss the following:
  - What are the challenges to continuously exploring?
  - What are the challenges to continuously integrating?
  - What are the challenges to continuously deploying?
- ▶ **Step 2:** As a group, create a poster with the challenges you discussed in Step 1. What may be some ways to solve them?
- ▶ **Step 3:** Be prepared to share with the class.



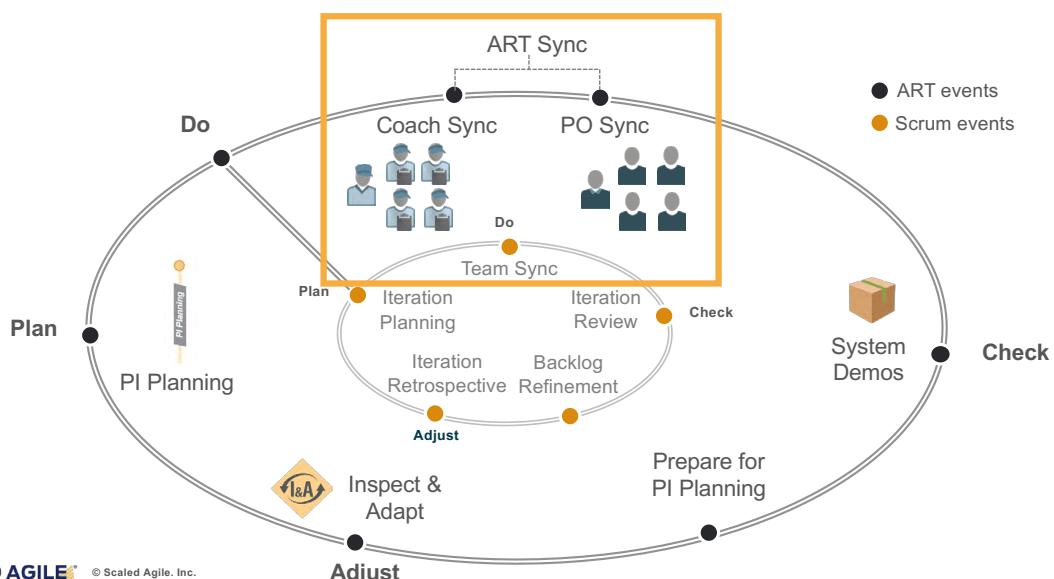
SCALED AGILE® © Scaled Agile, Inc.

5-16

## 5.2 Syncs align delivery

### Sync events drive the delivery of value

Utilize syncs to maintain connectivity as you deliver and overcome challenges together.



## Communication and synchronization with the Team Sync

### Basic daily Team Sync agenda

Each person answers:

1. What did I do yesterday to advance the Iteration Goals?
2. What will I do today to advance the Iteration Goals?
3. Are there any impediments that will prevent the team from meeting the Iteration Goals?

### The meet-after agenda

1. Review topics captured on the meet-after board
2. Involved parties discuss and uninvolved people may leave



### Activity: Reenact a Team Sync



You will participate in or observe a reenactment of a Team Sync. A group of volunteers will play the role of team members. Your trainer will play the role of the Scrum Master / Team Coach (SM/TC).

- ▶ **Step 1:** Volunteers receive a persona card. Do not show your card to others.
- ▶ **Step 2:** Run a Team Sync, playing the role assigned by the card.
- ▶ **Step 3:** The rest of the class observes and considers the following:
  - How long do you think the meeting should be?
  - Where should it take place?
  - What is the main purpose of the Team Sync?

## Reenact the Team Sync

**How long do you think the meeting should be?**

**Where should it take place?**

**What is the main purpose of the Team Sync?**

## ART Sync is used to coordinate progress



### Coach Sync

- ▶ Visibility into progress and impediments to flow
- ▶ Facilitated by RTE
- ▶ Participants: SM/TCs, other select team members, and SMEs as 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 Product Manager
- ▶ Participants: Product Managers, POs, other stakeholders, and SMEs as necessary
- ▶ Weekly or more frequently, 30 – 60 minutes
- ▶ Timeboxed and followed by a meet-after

## 5.3 Building quality in

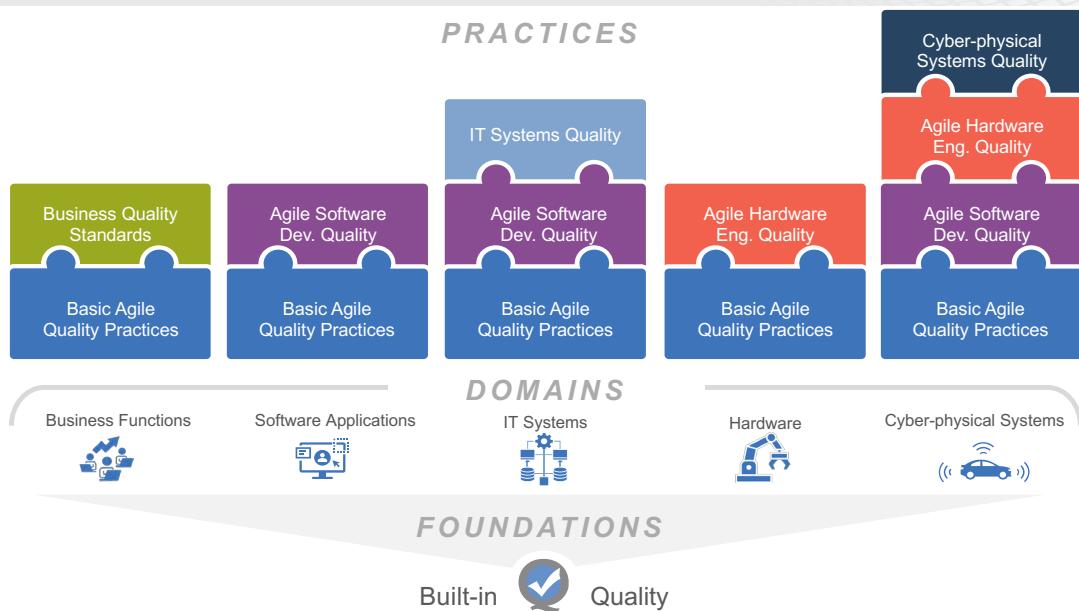


## Discussion: Express “Build quality in”



- ▶ **Step 1:** As a class, discuss what the phrase “build quality in” means to you
- ▶ **Step 2:** Identify how your teams build quality in and what value it brings to your Customer, your organization, and your Agile Teams

## Built-In Quality



## Foundations of Built-in Quality

- ▶ Think shift-left
- ▶ Shorten the define-build-test cycle
- ▶ Remove barriers to collaboration
- ▶ Design for quality
- ▶ Deliver continuously for fast Customer feedback



SCALED AGILE® © Scaled Agile, Inc.

5-25

## Basic Agile quality practices

Agile quality practices apply to every team, whether business or technology.

- ▶ **Establish flow:** avoid stopping and starting as work moves through the system
- ▶ **Peer review and pairing:** multiple viewpoints enhance knowledge and work quality
- ▶ **Collective ownership and standards:** reduce bottlenecks and ensure consistency
- ▶ **Automation:** minimize manual processes to enable smaller batches
- ▶ **Definition of done:** ensure consistent quality measures for each work product



SCALED AGILE® © Scaled Agile, Inc.

5-26

## Agile software development quality practices

- ▶ **Continuous integration:** continually check many small changes for conflicts and errors
- ▶ **Test-first practices:** define and execute many tests early, often, and at multiple levels of integration
- ▶ **Refactoring:** continuously modify the system to provide a foundation to efficiently deliver value in the future
- ▶ **Continuous delivery:** apply continuous exploration, integration, deployment, and release on demand
- ▶ **Agile architecture:** evolve the architecture continuously, while supporting the needs of current users

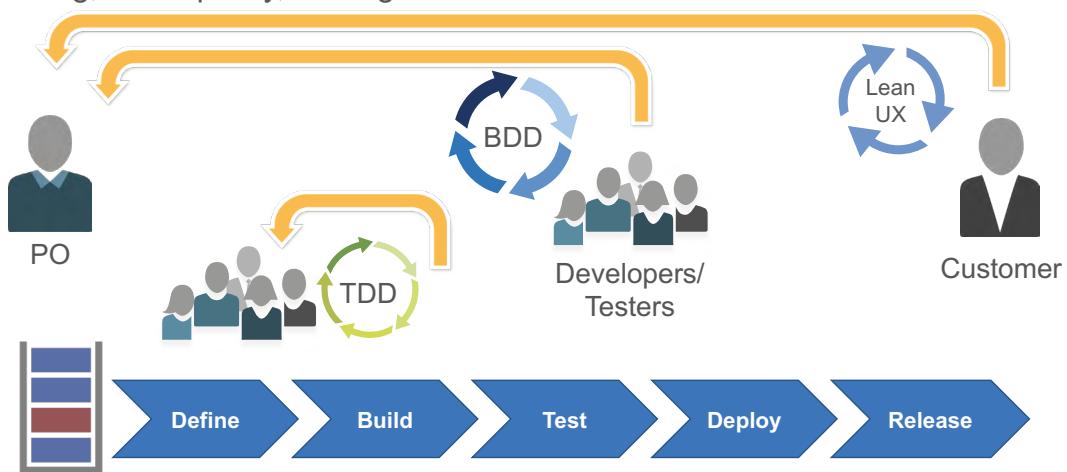
Agile Software Dev. Quality

SCALED AGILE® © Scaled Agile, Inc.

5-27

## Test-first software practices

Software quality practices, most inspired by Extreme Programming (XP), like Agile testing, behavior-driven development (BDD), test-driven development (TDD), refactoring, code quality, and Agile architecture.

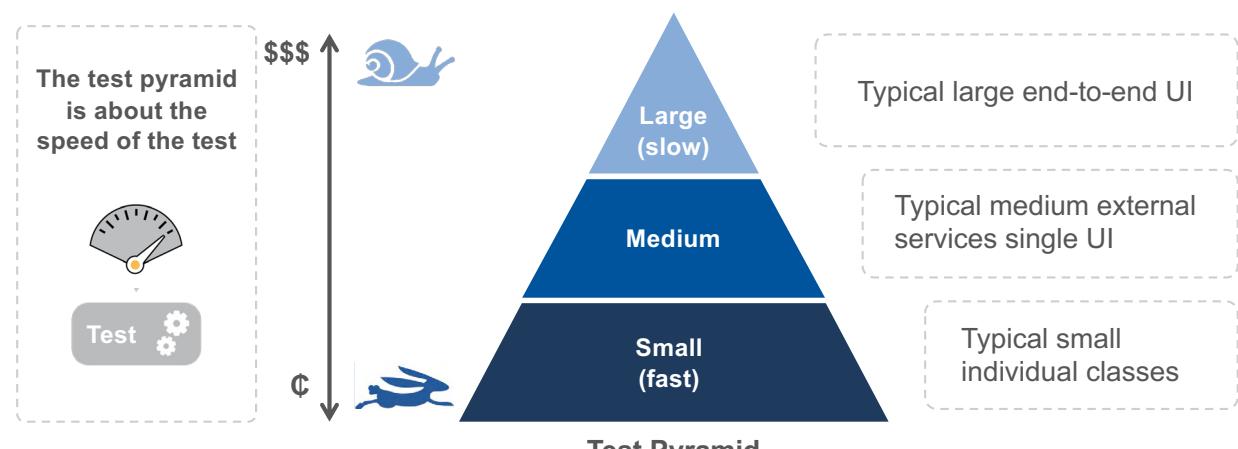


SCALED AGILE® © Scaled Agile, Inc.

5-28

## A test-first approach creates a balanced Agile testing pyramid

A balanced portfolio of tests contains many small, low-level, automated tests and fewer large, manual tests.

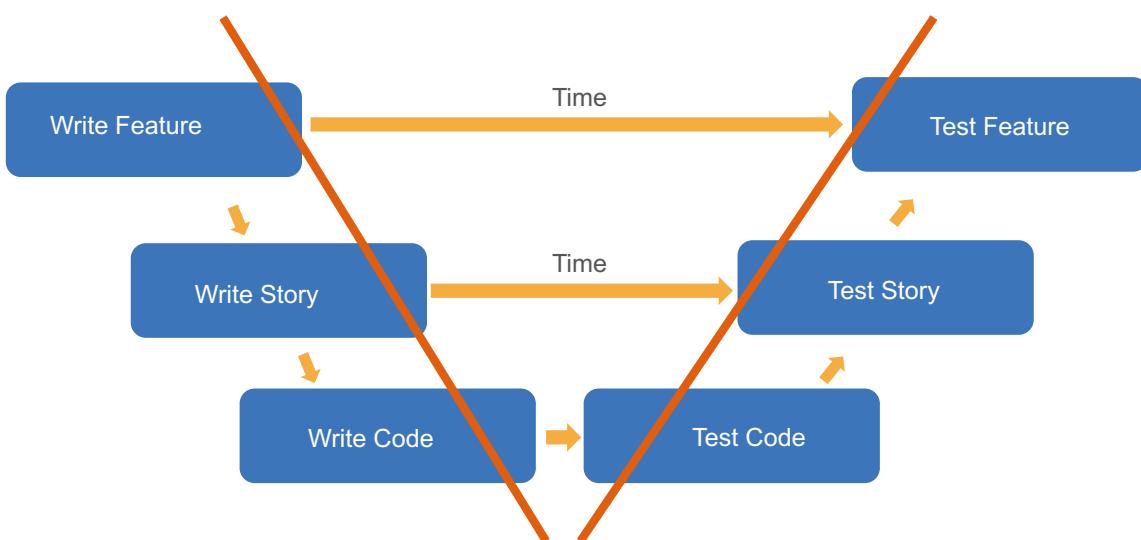


Reference: Cohn, *Succeeding with Agile*

SCALED AGILE® © Scaled Agile, Inc.

5-29

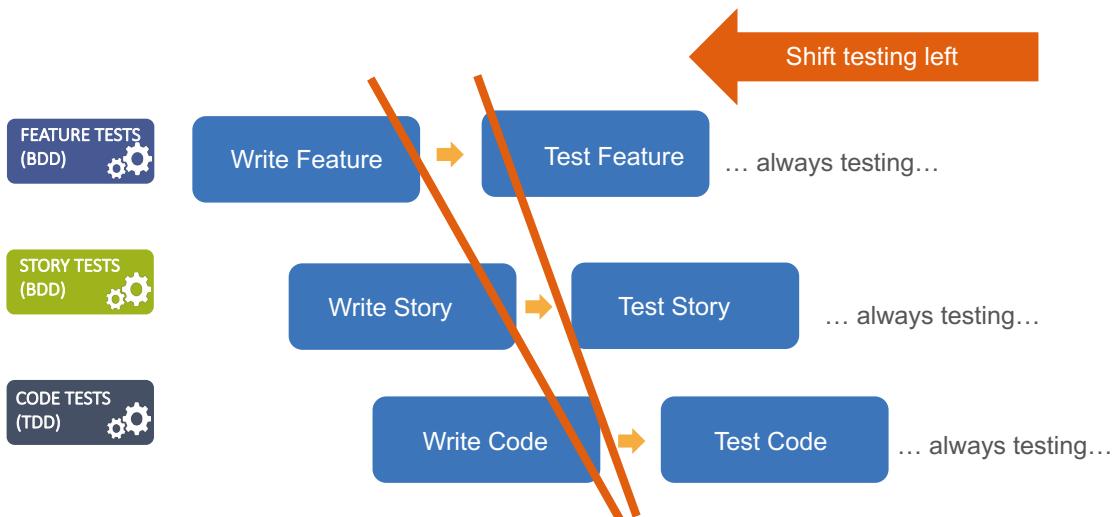
## Traditional testing (V-Model) delays feedback



SCALED AGILE® © Scaled Agile, Inc.

5-30

## Shift testing left for fast and continuous feedback



SCALED AGILE® © Scaled Agile, Inc.

5-31

## Business quality practices

- ▶ **Address specific functional quality standards:** Standards have historical and legacy context that require adjustment when applying them to a flow-based context.
- ▶ **Apply standards in an Agile manner:** apply standards early and continually (shift-left) and strive to automate them where possible



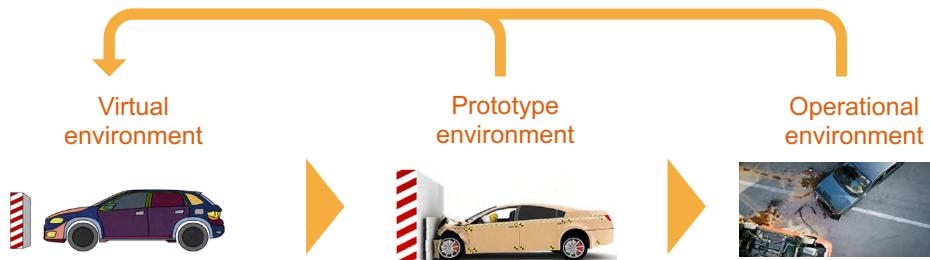
SCALED AGILE® © Scaled Agile, Inc.

5-32

## Agile hardware engineering quality practices

- **Modeling and simulation:** analyze and simulate in a virtual environment to test changes quickly and cost-effectively
- **Rapid prototyping:** integrate mockups (for example, 3D-printed parts) for higher-fidelity feedback

Agile Hardware  
Eng. Quality



SCALED AGILE® © Scaled Agile, Inc.

5-33

## Scalable definition of done



Team Increment	System Increment	Solution Increment	Release
<ul style="list-style-type: none"><li>Stories satisfy acceptance criteria</li><li>Acceptance tests passed (automated where practical)</li><li>Unit tests</li><li>Cumulative unit tests passed</li><li>Assets under version control</li><li>Engineering standards followed</li><li>NFRs met</li><li>No must-fix defects</li><li>Stories accepted by PO</li></ul>	<ul style="list-style-type: none"><li>Stories completed by all teams in the ART and integrated</li><li>Completed features meet acceptance criteria</li><li>NFRs met</li><li>No must-fix defects</li><li>Verification and validation of key scenarios</li><li>Included in build definition and deployment process</li><li>Increment demonstrated, feedback achieved</li><li>Accepted by Product Management</li></ul>	<ul style="list-style-type: none"><li>Capabilities completed by all ARTs and meet acceptance criteria</li><li>Deployed/installed in the staging environment</li><li>NFRs met</li><li>System end-to-end integration, verification and validation done</li><li>No must-fix defects</li><li>Included in build definition and deployment/transition process</li><li>Documentation updated</li><li>Solution demonstrated, feedback achieved</li><li>Accepted by Solution Management</li></ul>	<ul style="list-style-type: none"><li>All capabilities done and meet acceptance criteria</li><li>End-to-end integration and solutions V&amp;V done</li><li>Regression testing done</li><li>NFRs met</li><li>No must-fix defects</li><li>Release documentation complete</li><li>All standards met</li><li>Approved by Solution and Release Management</li></ul>

5-34



## Activity: What is your definition of done?



- ▶ **Step 1:** As a group, craft a definition of what it means for you to finish a team increment
- ▶ **Step 2:** Considering the elements of a team increment in the previous slide, discuss what is included in your definition of done
- ▶ **Step 3:** Discuss how you might mature your definition of done to improve the quality of your Solution
- ▶ **Step 4:** Be prepared to share with the class



SCALED AGILE® © Scaled Agile, Inc.

5-35



## Action Plan: Deliver value



- ▶ **Step 1:** With your group, brainstorm actions you could take into the upcoming PI regarding one of the below:
  - Continuous Exploration
  - Continuous Integration
  - Continuous Deployment
- ▶ **Step 2:** As a group, select one action you agree to take in the coming PI
- ▶ **Step 3:** Be prepared to share with the class



SCALED AGILE® © Scaled Agile, Inc.

5-36



# Action Plan

Deliver Value

## Lesson review

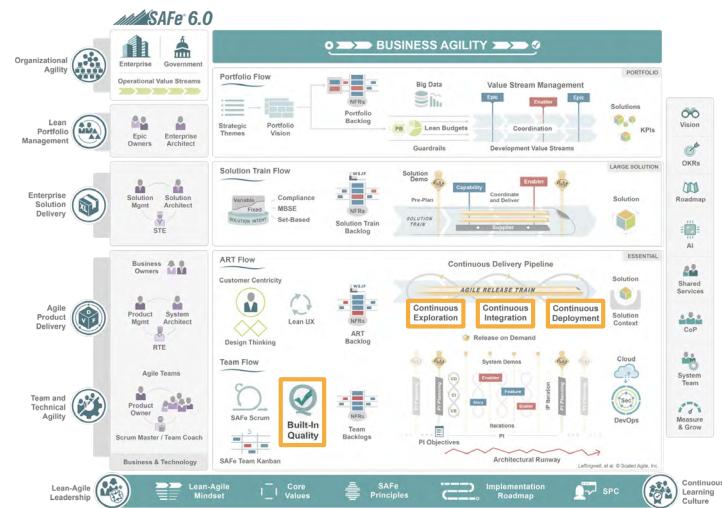
In this lesson you:

- ▶ Explained the importance of a CDP
- ▶ Described the communication and synchronization benefits of SAFe sync events
- ▶ Summarized different practices to build quality into Solutions
- ▶ Created action items as a team to take into the upcoming PI around improving your CDP and Built-in Quality

## Articles used in this lesson

Read these Framework articles to learn more about topics covered in this lesson

- ▶ “Continuous Exploration”  
<https://www.scaledagileframework.com/continuous-exploration/>
- ▶ “Continuous Integration”  
<https://www.scaledagileframework.com/continuous-integration/>
- ▶ “Continuous Deployment”  
<https://www.scaledagileframework.com/continuous-deployment/>
- ▶ “Built-in Quality”  
<https://www.scaledagileframework.com/built-in-quality/>



## Continue your SAFe journey with the following resources:

<p>Use the SAFe Collaborate template “Determine the Team’s Definition of Done” with your team to agree upon Guardrails, rules, and processes that ensure Built-in Quality for your Solutions.</p> <p><a href="https://bit.ly/Template-DetermineDoD">https://bit.ly/Template-DetermineDoD</a></p>	<p>Watch this sixty-minute video, <i>Community Webinar: DevSecOps in Real Life</i>, to delve into the most important practices that fuel the CDP and how they are implemented in real life.</p> <p><a href="https://bit.ly/Community-DevSecOpsWebinar">https://bit.ly/Community-DevSecOpsWebinar</a></p>
<p>Download “The Facilitator’s Guide to SAFe: Team Sync” for support facilitating your Team Sync events and ideas for overcoming potential issues you might encounter.</p> <p><a href="https://bit.ly/Community-FGDailyStand-Up">https://bit.ly/Community-FGDailyStand-Up</a></p>	<p>Download "The Facilitator’s Guide to SAFe: PO Sync" to ensure your PO Sync events stay on track, achieve their goals, and stay fresh to ensure strong, consistent participation.</p> <p><a href="https://bit.ly/Community-FGPOSync">https://bit.ly/Community-FGPOSync</a></p>
<p>Download "The Facilitator’s Guide to SAFe: Coach Sync" for guidance on preparation and execution of the Coach Sync, along with tips to keep the format engaging for attendees.</p> <p><a href="https://bit.ly/Community-FGSoS">https://bit.ly/Community-FGSoS</a></p>	<p>Download "The Facilitator’s Guide to SAFe: ART Sync" for an overview of the purpose, preparation, and execution details to ensure this meeting is valuable for the attendees and the organization.</p> <p><a href="https://bit.ly/Community-FGARTSync">https://bit.ly/Community-FGARTSync</a></p>

## References

Cohn, Mike. *Succeeding with Agile: Software Development Using Scrum*. Boston: Pearson Education Inc., 2013. 312

Oosterwal, Dantar P. *The Lean Machine: How Harley-Davidson Drove Top-Line Growth and Profitability with Revolutionary Lean Product Development*. New York: AMACOM, 2010. 143.

## Lesson notes

Enter your notes below. If using a digital workbook, save your PDF often so you don't lose any of your notes.

# Lesson 6

## Get Feedback

SAFe® Course - Attending this course gives learners access to the SAFe® Practitioner exam and related preparation materials.



SCALED AGILE®

### Lesson Topics

- 6.1 Getting Customer Feedback
- 6.2 Demonstrating value with the Iteration Review
- 6.3 System Demo



## Learning objectives

At the end of this lesson, you should be able to:

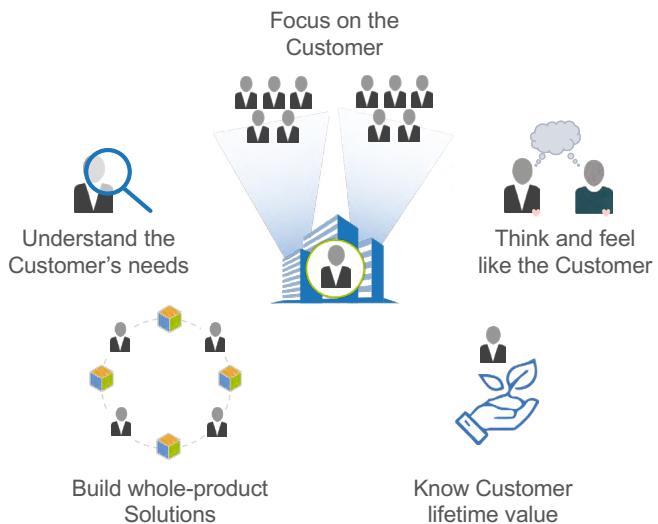
- ▶ Summarize techniques to maintain Customer focus
- ▶ Identify Iteration Review steps for the Agile Team
- ▶ Recognize the importance of integrating and demonstrating together with the System Demo
- ▶ Identify one or more actions your team can take to develop or maintain a Customer focus in the upcoming PI

## 6.1 Getting Customer Feedback

## Connect to the Customer

Agile Teams use multiple techniques to maintain Customer focus:

- ▶ Product roles as a proxy
- ▶ Gemba walks
- ▶ Empathy mapping
- ▶ Solution telemetry
- ▶ System Demo



SCALED AGILE® © Scaled Agile, Inc.

6-5

## Product roles as a proxy

- ▶ Focus on how value is determined by the Customer
- ▶ Bring the Business Owners, Customers, and end users as needed to ART and team events
- ▶ Identify the need and plans for implementing Design Thinking activities such as Gemba walks and empathy interviews



Understand the Customer's needs

SCALED AGILE® © Scaled Agile, Inc.

6-6

## Gemba and empathic design

### Gemba walks

Gemba walking refers to observing Customers in their day-to-day, using Solutions, or observing problem spaces

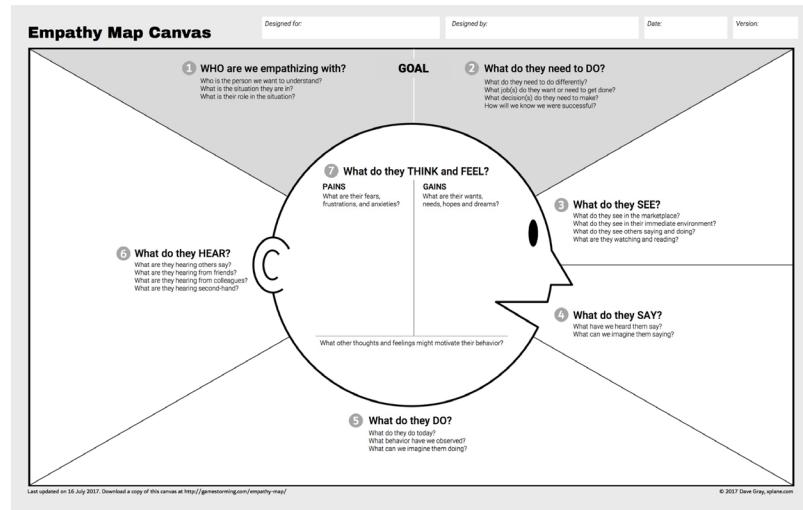
### Empathic design

Empathic design refers to our ability to put aside our preconceived ideas and develop Solutions from the perspective of our Customers



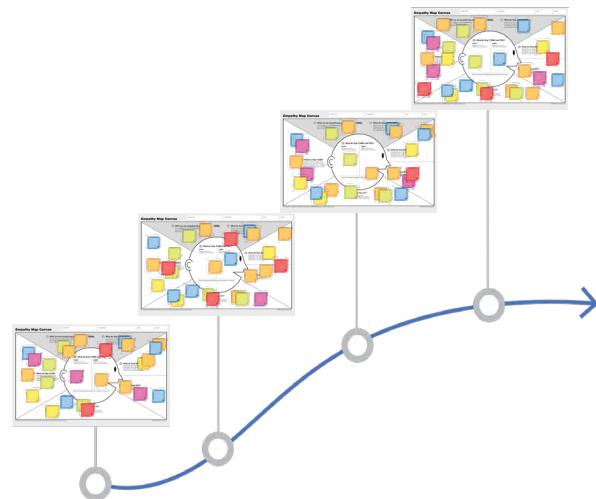
## Empathy map

- The empathy map is a tool that helps teams develop deep, shared understanding and empathy for others
- The empathy map can be used to design better experiences and Value Streams



## Empathy maps evolve

- ▶ What do our Customers see, think, and feel now?
  - More Gemba walks after the Solution is in use can build on previous innovation
- ▶ How do we hope our planned releases change what is seen, considered and felt?
- ▶ When should we update our persona and empathy map based on these changes?



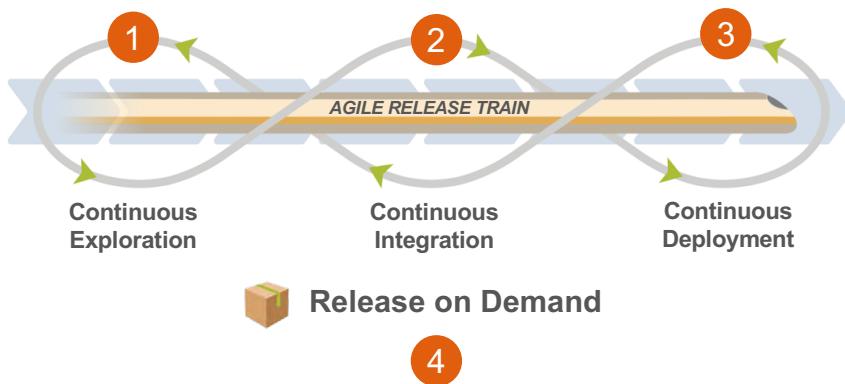
SCALED AGILE® © Scaled Agile, Inc.

6-9

## Enable feedback by building telemetry into solutions

- 1 Architect for operations by designing logging and telemetry into Enterprise solutions
- 2 Build telemetry into each application to help determine the results of relevant hypotheses
- 3 Use telemetry to monitor for problems across the entire stack
- 4 Collect and manage the telemetry data used to track and measure hypotheses

### Continuous Delivery Pipeline



SCALED AGILE® © Scaled Agile, Inc.

6-10



## Discussion: Keeping the Customer focus



- ▶ **Step 1:** Individually, consider the following:
  - What techniques do the ART and teams already utilize to get Customer feedback?
  - What techniques may be valuable to add in?
- ▶ **Step 2:** Discuss as a class

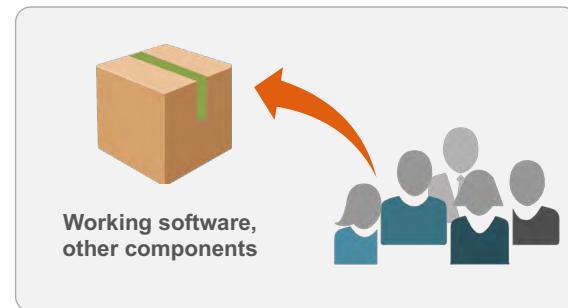
## 6.2 Demonstrating value with the Iteration Review

## The Iteration Review

The Iteration Review provides the true measure of progress by showing working software functionality and business outcome progression.

- ▶ Preparation starts with planning
- ▶ Teams demonstrate every Story, spike, refactor, and NFR
- ▶ Attended by the team and its stakeholders

Demonstrating a working, tested team increment



## Iteration Review guidelines

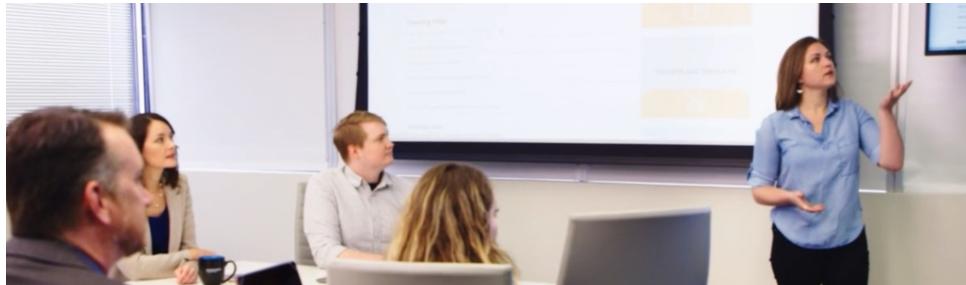
- ▶ **Timebox:** Approximately one-to-two hours long.
- ▶ **Preparation:** Review preparation should be limited to one-to-two hours. Minimize presentation. Work from the repository of Stories.
- ▶ **Attendees:** If a major stakeholder cannot attend, the PO should follow up individually.

### Sample Iteration Review Agenda

1. Review business context and Iteration Goals
2. Demo and solicit feedback of each Story, spike, refactor, and NFR
3. Discuss Stories not completed and why
4. Identify risks, impediments
5. Revise team backlog and Team PI Objectives as needed

## Two views from the Iteration Review based on a working system

- ▶ **How did we do in the Iteration?**
  - Did we meet the goal?
  - Story-by-Story review
- ▶ **How are we doing in the PI?**
  - Review of PI Objectives
  - Review of remaining PI scope and reprioritizing if necessary



SCALED AGILE® © Scaled Agile, Inc.

6-15

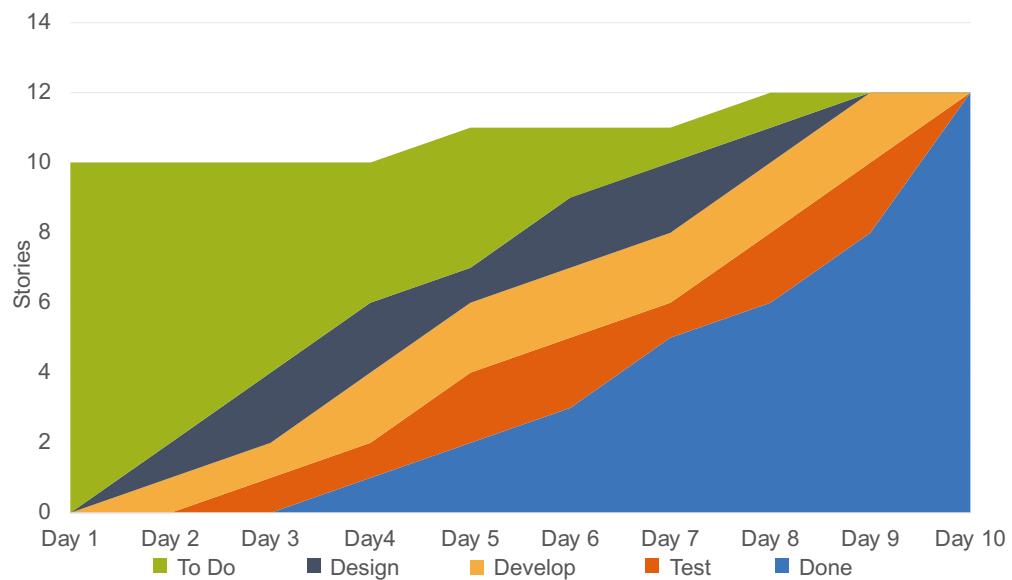
## Team Reviews within SAFe Team Kanban

- ▶ **How did we do in the Iteration?**
  - Did we meet the dependencies other teams relied on us for?
  - Have we been meeting our Team DoD?
  - Is our Flow Time stabilized or improving?
- ▶ **How are we doing in the PI?**
  - Review of Team PI Objectives
  - Review of remaining known PI scope and reprioritizing if necessary
  - Alignment on Kanban board

SCALED AGILE® © Scaled Agile, Inc.

6-16

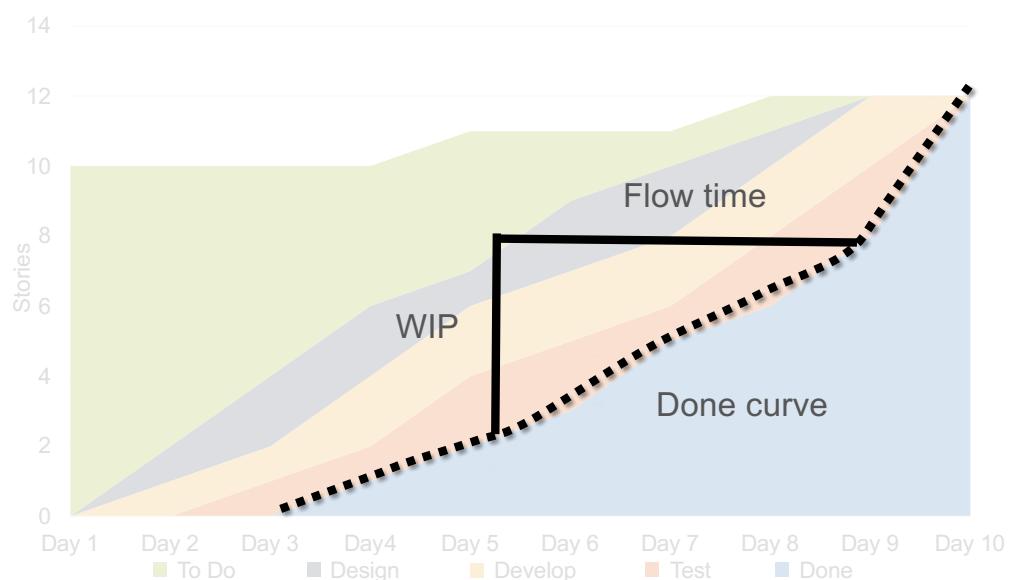
## Understand cumulative flow diagrams (CFDs)



SCALED AGILE® © Scaled Agile, Inc.

6-17

## What can you learn from a CFD?



SCALED AGILE® © Scaled Agile, Inc.

6-18

## 6.3 System Demo

## SAFe Lean-Agile Principles

#1 Take an economic view

#2 Apply systems thinking

#3 Assume variability; preserve options

#4 Build incrementally with fast, integrated learning cycles

### **#5 Base milestones on objective evaluation of working systems**

#6 Make value flow without interruptions

#7 Apply cadence, synchronize with cross-domain planning

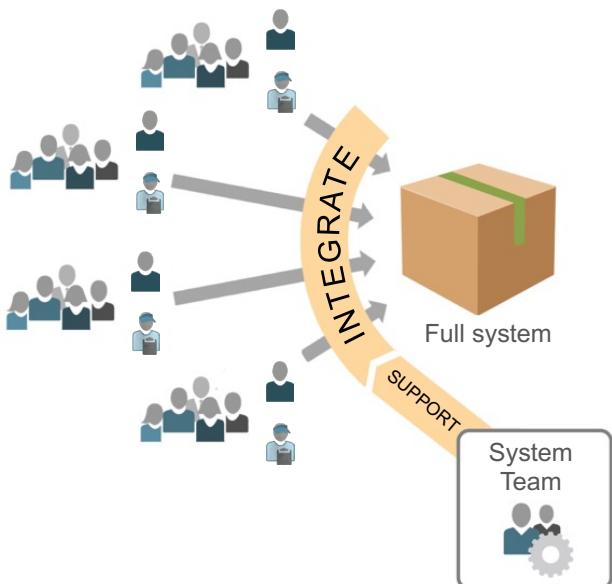
#8 Unlock the intrinsic motivation of knowledge workers

#9 Decentralize decision-making

#10 Organize around value

## Base Milestones on objective evaluation of working systems

- ▶ Demos happen after the Iteration Review and may lag by one Iteration at most
- ▶ Features are functionally complete or ‘toggled’ so as not to disrupt demonstrable functionality
- ▶ New Features work together and with existing functionality
- ▶ Demos happen from a staging environment that resembles production as much as possible



SCALED AGILE® © Scaled Agile, Inc.

6-21

## The System Demo demonstrates value at each iteration

System Demo tests and evaluates the full solution in a production-like context, often through staging, to receive feedback from stakeholders.

### Sample Agenda:

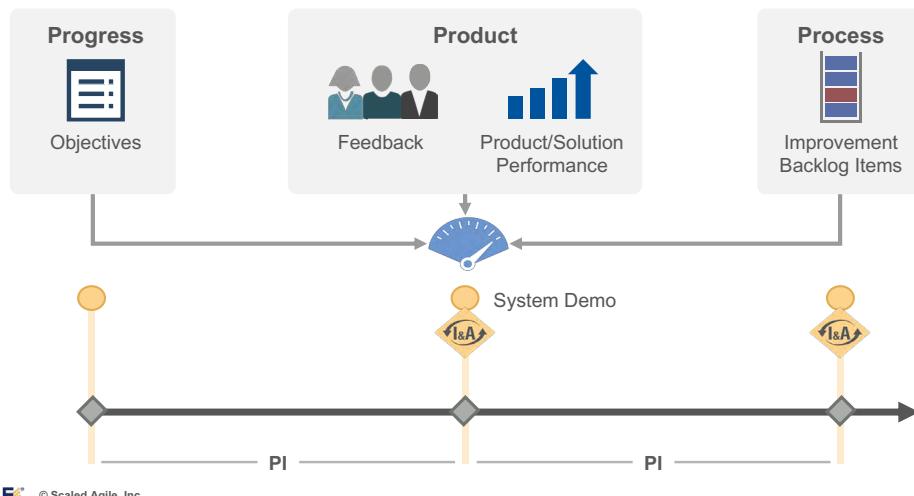
- Review the business context and the PI Objectives (around 5 – 10 minutes)
- Briefly describe each new Feature before demoing (around 5 minutes)
- Demo each new Feature in an end-to-end use case (around 20 – 30 minutes total)
- Open discussion of questions and feedback
- Summarize progress, feedback, and action items

SCALED AGILE® © Scaled Agile, Inc.

6-22

## The PI System Demo evaluates the full PI

PI System Demos are orchestrated to deliver objective progress, product, and process Metrics.



SCALED AGILE® © Scaled Agile, Inc.

6-23



### Discussion: System Demo challenges



- ▶ **Step 1:** In your group, discuss the challenges present in demonstrating a new system increment every two weeks
- ▶ **Step 2:** Brainstorm some ways to solve these challenges
- ▶ **Step 3:** Be prepared to share with the class

SCALED AGILE® © Scaled Agile, Inc.

6-24



## Action Plan: Get feedback



- ▶ **Step 1:** With your group, discuss and select one Customer feedback collection technique you would like to use in the upcoming PI
- ▶ **Step 2:** Identify any other teams or stakeholders you may need to work with to collect Customer feedback
- ▶ **Step 3:** Plan for how your team will collaborate with other teams and stakeholders during PI Planning to plan your feedback campaign
- ▶ **Step 4:** Add your plan to the Action Plan in your workbook





## Action Plan

Get Feedback

## Lesson review

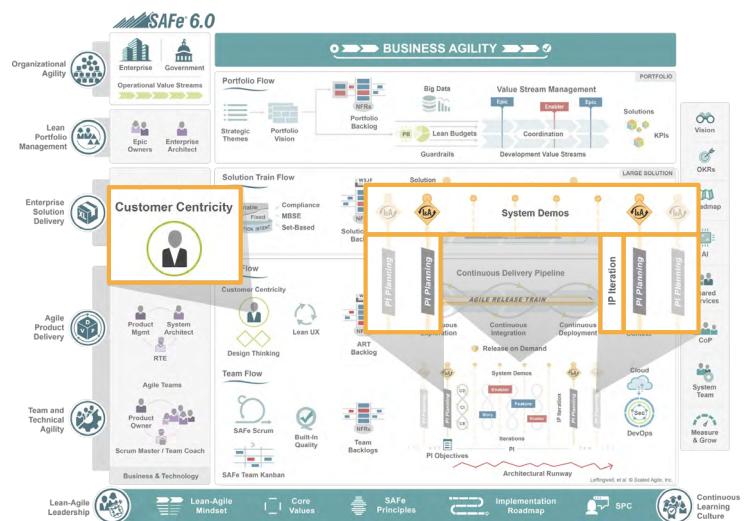
In this lesson you:

- ▶ Summarized techniques to maintain Customer focus
- ▶ Identified Iteration Review steps for the Agile Team
- ▶ Recognized the importance of integrating and demonstrating together with the System Demo
- ▶ Identified one or more actions your team can take to develop or maintain a Customer focus in the upcoming PI

## Articles used in this lesson

Read these Framework articles to learn more about topics covered in this lesson:

- ▶ "System Demo"  
<https://www.scaledagileframework.com/system-demo/>
- ▶ "Iteration Review"  
<https://www.scaledagileframework.com/iteration-review/>
- ▶ "Customer Centricity"  
<https://www.scaledagileframework.com/customer-centrality/>



## Continue your SAFe journey with the following resources:

Download the "Facilitator's Guide to SAFe: Iteration Review and Demo" for guidance in preparing to facilitate Iteration Review and Demo events.  
<https://bit.ly/Community-FGIterationReview>

Watch this five-minute video, *How To Run An Effective SAFe Iteration Review*, to get an overview of the Iteration Review event along with practical tips on how to facilitate the event.  
<https://bit.ly/Video-IterationReview>

Download and use the "Facilitator's Guide to SAFe: System Demo" for practical guidance on how to prepare for and execute the System Demo.  
<https://bit.ly/Community-FGSytemDemo>

Watch and share this two-minute video, *ART Events: System Demo*, to learn about the purpose and benefit of this critical ART event.  
<https://bit.ly/Video-SystemDemo>

## References

Gray, Dave. "Empathy Map." Game Storming. Updated November 12, 2009.  
<https://gamestorming.com/empathy-Dmap>.

## Lesson notes

Enter your notes below. If using a digital workbook, save your PDF often so you don't lose any of your notes.

# Lesson 7

## Improve Relentlessly

SAFe® Course - Attending this course gives learners access to the SAFe® Practitioner exam and related preparation materials.



SCALED AGILE®

### Lesson Topics

- 7.1 Improving competency
- 7.2 Improving flow
- 7.3 Improving outcomes
- 7.4 Starting the improvement journey

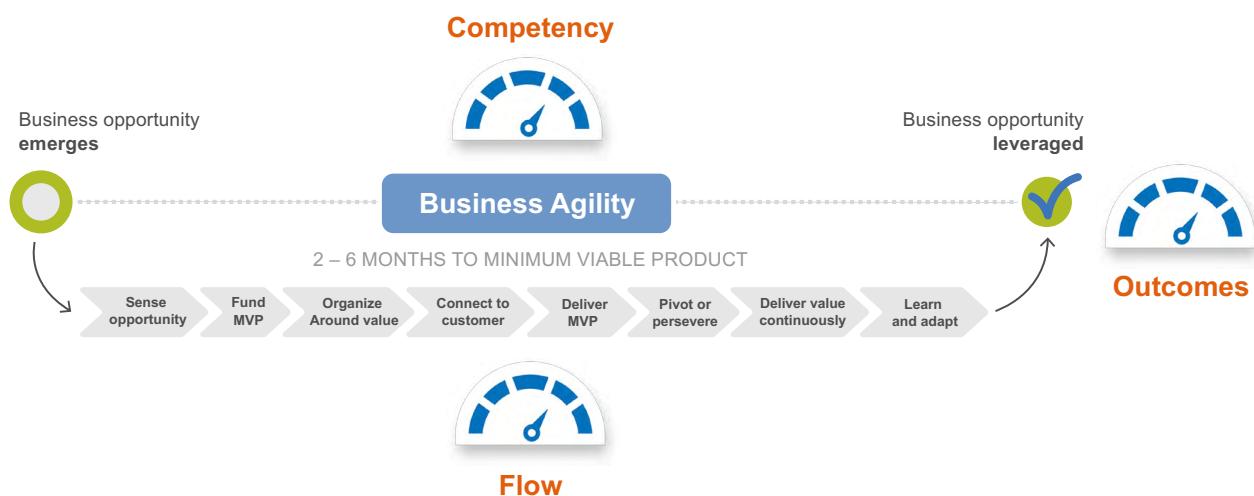


## Learning objectives

At the end of this lesson, you should be able to:

- ▶ Summarize the steps of a retrospective
- ▶ Identify the steps of Inspect & Adapt (I&A)
- ▶ Explain the eight flow accelerators
- ▶ Summarize how to apply flow Metrics to assess the organization's ability to make value flow without interruption
- ▶ Describe the purpose of a balanced Metric approach
- ▶ Identify the top three actions your team can take into the upcoming PI to develop and grow your SAFe Agile Team practices

## Business Agility requires improvement in three domains



# 7.1 Improving competency

**SAFED AGILE** © Scaled Agile, Inc.

7-5

## Delivering business agility requires technical competency

**Competency**

Competency improvement activities include:

- ▶ Retrospectives
- ▶ SAFe Assessments
- ▶ I&A

**Team and Technical Agility**

**Agile Product Delivery**

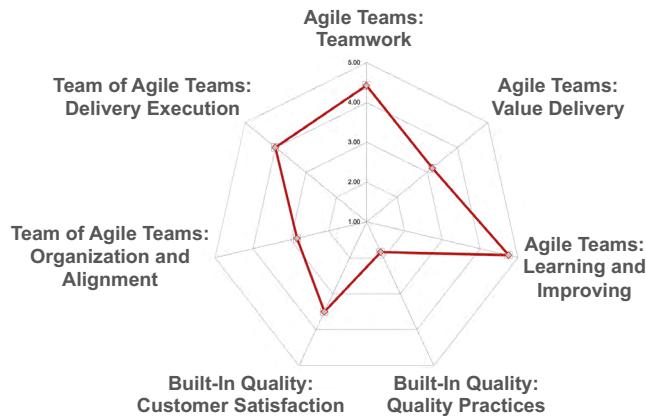
**SAFED AGILE** © Scaled Agile, Inc.

7-6

## Competency assessments

- ▶ Periodically measure the progress being made toward the three dimensions of each competency
- ▶ Identify specific practices for potential improvement
- ▶ Reassess periodically to observe trends
- ▶ Make growth recommendations available as you assess

### Example: Team and Technical Agility Self-Assessment



<https://bit.ly/Community-MeasureAndGrow>

SCALED AGILE® © Scaled Agile, Inc.

7-7

## Iteration Retrospective

- ▶ **Timebox:** Should be an hour or less.
- ▶ **Purpose:** Align on items that can be improved upon in next Iteration.
- ▶ **Outcome:** Enter the improvement items in the Team Backlog.
- ▶ **Tip:** Action items take time and energy. Prioritize them alongside other work so the work to be done is visible.

### Sample Agenda

- Part 1: Quantitative**
1. Review the improvement backlog items targeted for this Iteration. Were they all accomplished?
  2. Did the team meet the goals (yes/no)?
  3. Collect and review the agreed-to Iteration metrics.
- Part 2: Qualitative**
1. What went well?
  2. What didn't?
  3. What can we do better next time?

SCALED AGILE® © Scaled Agile, Inc.

7-8



## Activity: Simulate a retrospective



In your group, run a retrospective of this course so far.

- ▶ **Step 1:** Pick someone in your group to play the role of the Scrum Master / Team Coach (SM/TC). They will facilitate the retrospective.
- ▶ **Step 2:** As a group, participate in the retrospective by discussing the following:
  - What went well?
  - What didn't go so well?
  - What can be done better?
- ▶ **Step 3:** Share some of your group's insights with the class.

## SAFe Lean-Agile Principles

#1 Take an economic view

#2 Apply systems thinking

#3 Assume variability; preserve options

#4 Build incrementally with fast, integrated learning cycles

### #5 Base milestones on objective evaluation of working systems

#6 Make value flow without interruptions

#7 Apply cadence, synchronize with cross-domain planning

#8 Unlock the intrinsic motivation of knowledge workers

#9 Decentralize decision-making

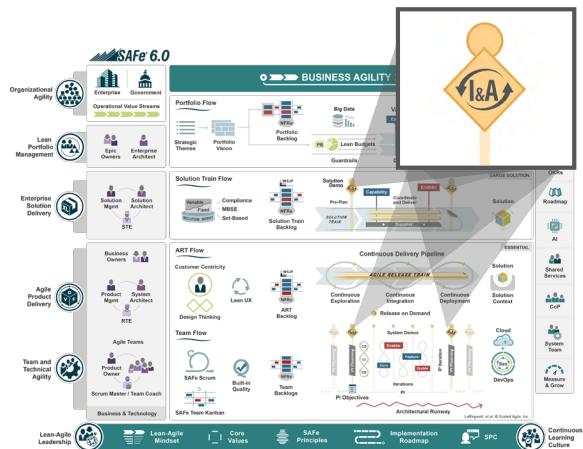
#10 Organize around value

## Improving with I&A

There are three parts of I&A:

1. The PI System Demo
2. Quantitative and Qualitative Measurement
3. Problem-Solving Workshop

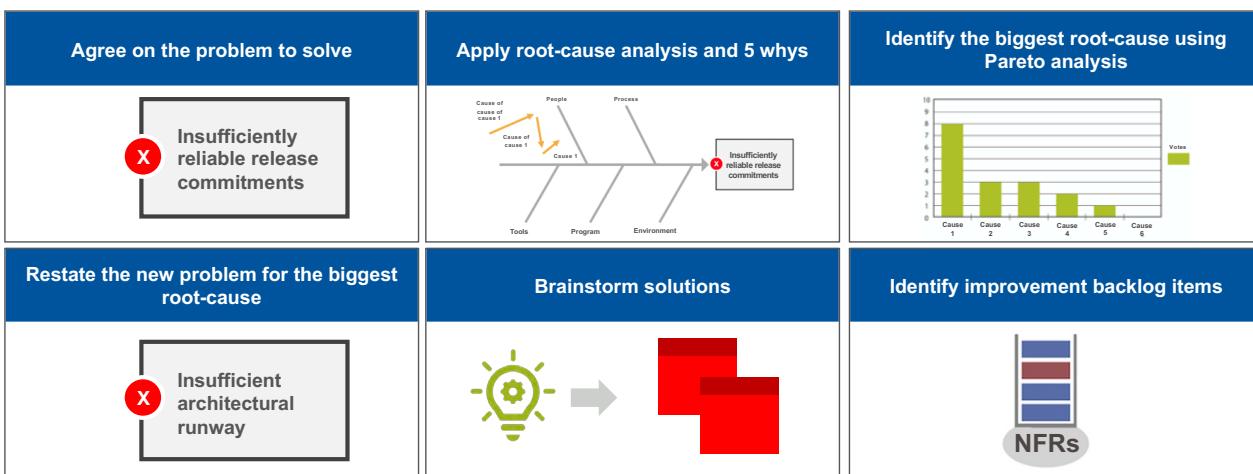
- **Timebox:** Three-to-four hours per PI
- **Attendees:** Teams and stakeholders



SCALED AGILE® © Scaled Agile, Inc.

7-11

## The Problem-Solving Workshop



SCALED AGILE® © Scaled Agile, Inc.

7-12

## 7.2 Improving flow

### SAFe Lean-Agile Principles

#1 Take an economic view

#2 Apply systems thinking

#3 Assume variability; preserve options

#4 Build incrementally with fast, integrated learning cycles

#5 Base milestones on objective evaluation of working systems

**#6 Make value flow without interruptions**

#7 Apply cadence, synchronize with cross-domain planning

#8 Unlock the intrinsic motivation of knowledge workers

#9 Decentralize decision-making

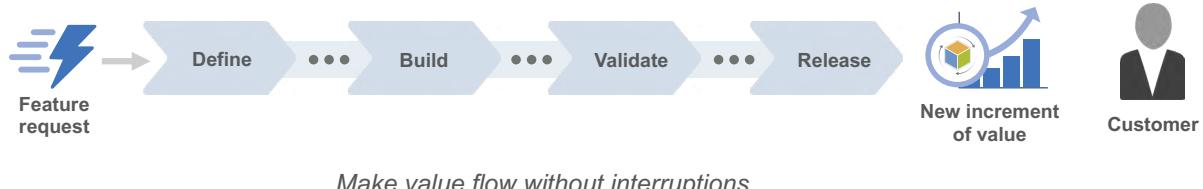
#10 Organize around value

## What is flow?

"To enable fast and predictable lead times in any value stream, there is usually a relentless focus on creating a smooth and even flow of work...."

—Kim et al., *The DevOps Handbook*

- ▶ Delivering a continuous flow of value to customers in the shortest sustainable lead time is the central theme of SAFe
- ▶ Accomplishing this requires dedication to working towards an entirely new way of working



*Make value flow without interruptions*

## The Eight Flow Accelerators

1. Visualize and limit WIP
2. Address bottlenecks
3. Minimize handoffs and dependencies
4. Get faster feedback
5. Work in smaller batches
6. Reduce queue lengths
7. Optimize time 'in the zone'
8. Remediate legacy policies and practices



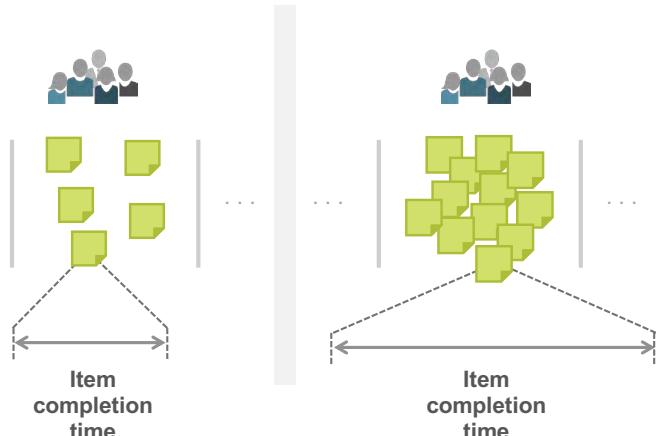
## 1. Visualize and limit WIP

### ► Why it matters:

- Excessive WIP decreases team productivity and impedes the flow of value
- Excessive WIP confuses individual and team priorities, causes frequent context switching, and increases waste and overhead

### ► What to do about it:

- Make current WIP visible
- Set WIP limits to balance WIP against available capacity



SCALED AGILE® © Scaled Agile, Inc.

7-17

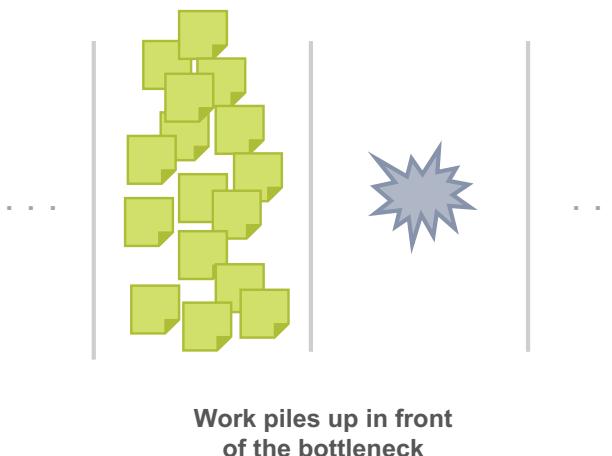
## 2. Address bottlenecks

### ► Why it matters:

- A bottleneck constrains the productivity of the entire ART
- Teams must address them to improve flow and continuously optimize

### ► What to do about it:

- Identify bottlenecks and understand their impact.
- Identify the resolution: bypass? Increase skills towards the bottleneck? Prioritize enablers?
- Assess the improvement impact on the system.



SCALED AGILE® © Scaled Agile, Inc.

7-18

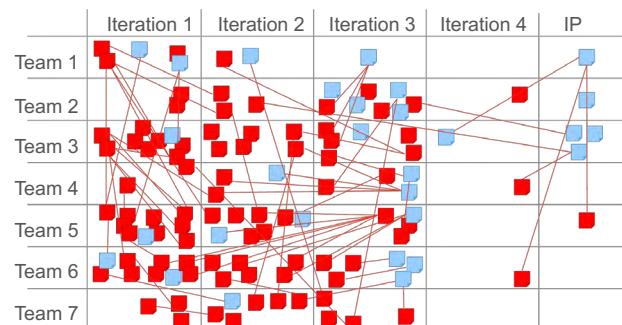
### 3. Minimize handoffs and dependencies

#### ► Why it matters:

Excessive and unnecessary dependencies and handoffs disrupt team flow, create delays, and increase context switching overhead

#### ► What to do about it:

- Use the planning board to visualize dependencies within and external to the ART
- Foster incremental execution of dependencies
- Organize around value, and reorganize when necessary



### 4. Get faster feedback

#### ► Why it matters:

When feedback is delayed or missing mistakes pile up quickly, leading to substantial rework for multiple teams, slow delivery, and unsatisfied Customers

#### ► What to do about it:

- Determine what types of feedback are missing or inadequate
- Shift reviews left
- Provide Solution telemetry
- Frequently engage with the Customer and Business Owner



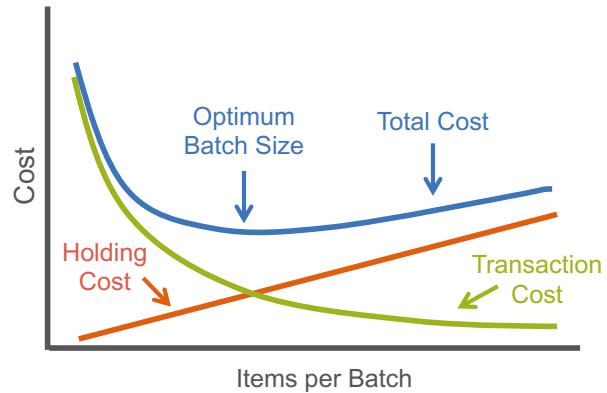
## 5. Work in smaller batches

### ► Why it matters:

Operating in large batches leads to delayed feedback, significant rework, and high variability. Teams have many different types of batches in play including feedback batches, integration batches and deployment batches.

### ► What to do about it:

- Use recommended cadence and team size
- Ensure enablers support automating the delivery pipeline
- Provide Solution telemetry
- Explicitly plan for small batches



SCALED AGILE® © Scaled Agile, Inc.

7-21

## 6. Reduce Queue Lengths

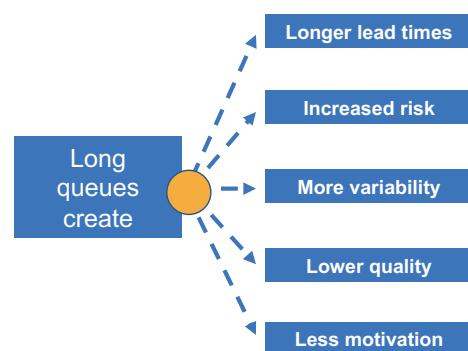
### ► Why it matters:

Queues represent committed work. The longer the queue, the longer the wait time for new functionality to be delivered to the Customer.

### ► What to do about it:

- Short Iteration time boxes and crisp Iteration and PI Objectives bring focus
- Ensure all work goes through the backlog
- Enable Product Managers and POs to exert positive yet firm ways to prioritize
- Leave capacity for emergent priorities

### Long queues: All bad



SCALED AGILE® © Scaled Agile, Inc.

7-22

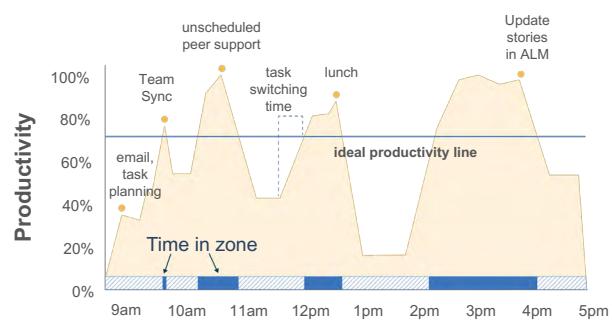
## 7. Optimize time 'in the zone'

### ► Why it matters:

Solution development relies heavily on creativity, focus, and deep, intellectual effort. It may take up to 20 minutes to fully immerse in work, and a simple external factor may instantly interrupt it.

### ► What to do about it:

- Continually optimize the efficiency of all meetings
- Refine what productive collaboration patterns mean to the team
- Maintain Solution health
- Make WIP visible, update limits across teams as needed



$$\frac{2.4 \text{ hrs (time in zone)}}{8 \text{ hrs (full day)}} = 30\% \text{ of work day spent in zone}$$

SCALED AGILE® © Scaled Agile, Inc.

7-23

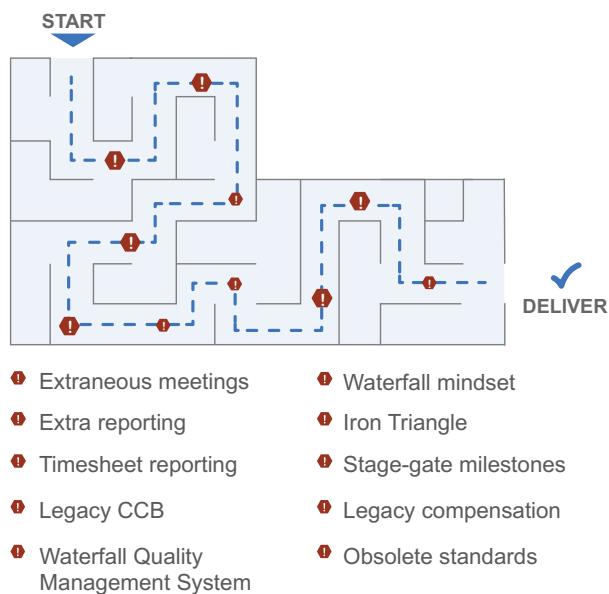
## 8. Remediate legacy policies and practices

### ► Why it matters:

Teams find themselves in a difficult predicament: pretend to conform to conflicting policies and reduce transparency, or fight the system while development is in process, slowing progress and creating friction. We want to be fully Agile, not half.

### ► What to do about it:

- Know that these impediments likely exist
- Grant time and space to raise issues
- Actively solve the issues you can within the teams and ART
- Raise the visibility of the effects of those out of ART control



SCALED AGILE® © Scaled Agile, Inc.

7-24

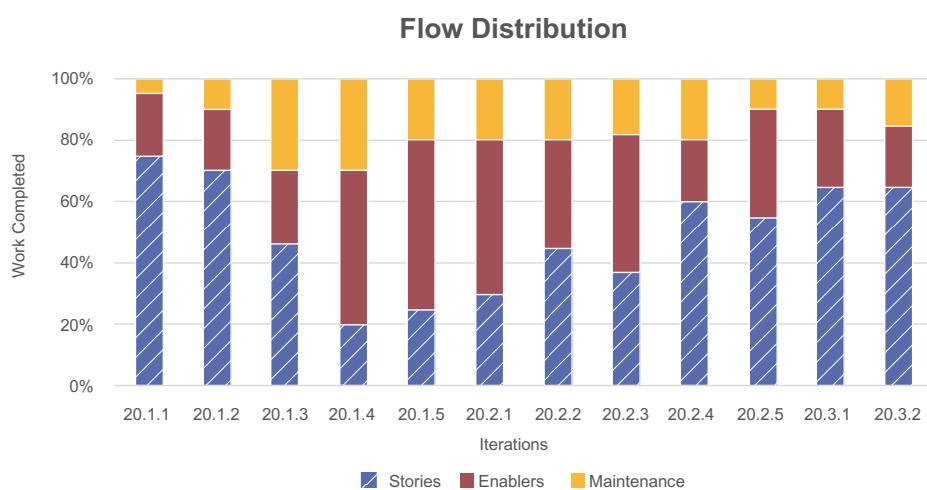
## Measure flow with six flow Metrics

Metric	Description
Flow Distribution	The amount of each type of work in the system over time
Flow Velocity	The average number of work items that can be completed in a given timeframe
Flow Time	How long it takes for a work item to go through the system
Flow Load	The overall amount of WIP in the system
Flow Efficiency	How much of the overall flow time is spent in value-added work activities vs. waiting between steps
Flow Predictability	Overall planned vs. actual business value achieved

Reference: Kersten, *Project to Product*

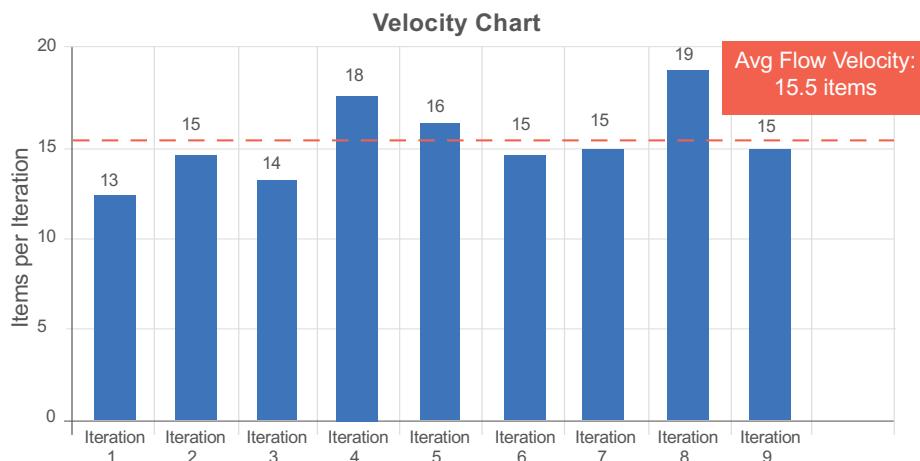
## Flow distribution ensures a healthy balance of work item types

**What does it measure?** Flow distribution measures the amount of each type of work in the system over time.



## Flow velocity tracks team performance each Iteration

**What does it measure?** Flow velocity measures the number of backlog items (Stories, Features, Capabilities, Epics) completed in a given timeframe; this is also known as the system's throughput.

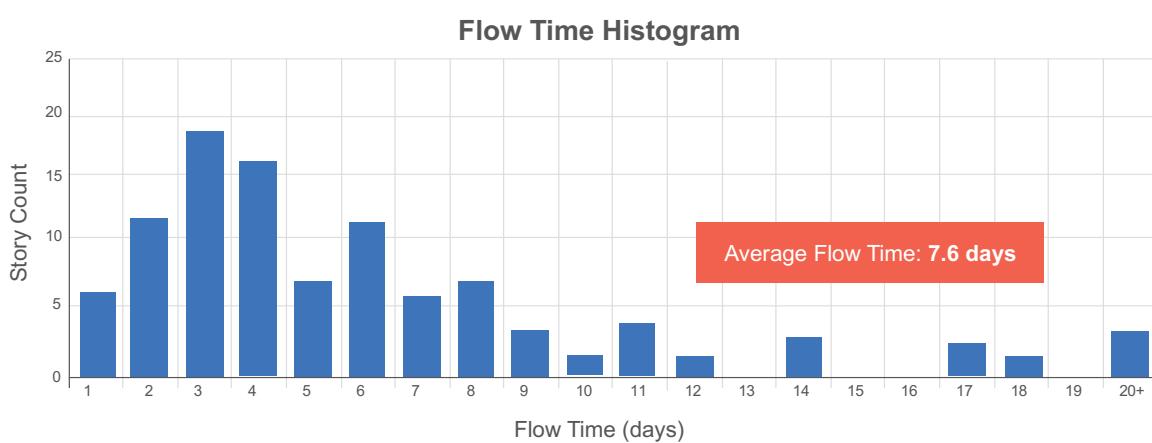


SCALED AGILE® © Scaled Agile, Inc.

7-27

## Flow time ensures teams deliver value in the shortest possible time

**What does it measure?** Flow time measures the elapsed time from when an item enters the system to the moment it is delivered to the Customer.

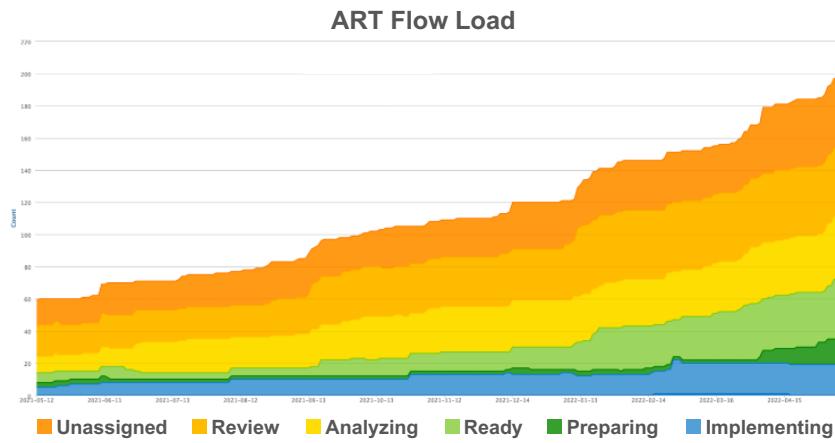


SCALED AGILE® © Scaled Agile, Inc.

7-28

## Flow load is a leading indicator of excess WIP

**What does it measure?** Flow load indicates how many items are currently in the system, and helps to optimize the team and ART throughput by limiting demand to the capacity.

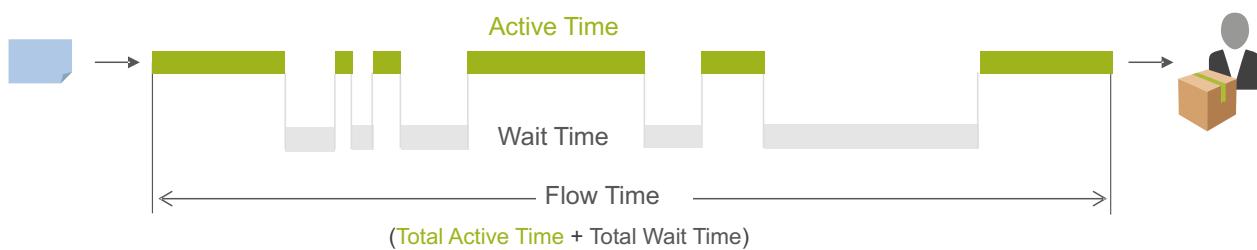


SCALDED AGILE® © Scaled Agile, Inc.

7-29

## Flow efficiency highlights waste, bottlenecks, and delays in the system

**What does it measure?** Flow efficiency measures how much of the overall flow time is spent in value-added work activities vs. waiting between steps.



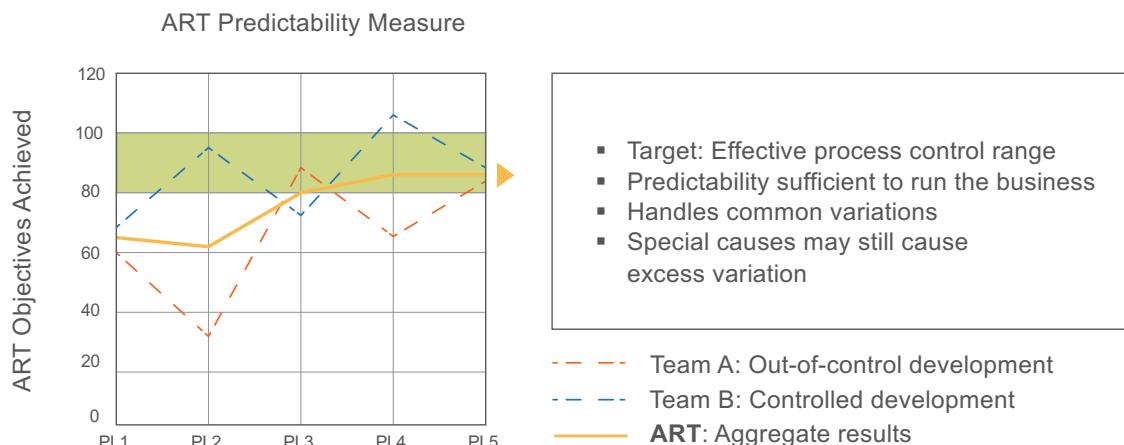
$$\text{Flow Efficiency} = \frac{\text{Total Active Time}}{\text{Flow Time}}$$

SCALDED AGILE® © Scaled Agile, Inc.

7-30

## Measure ART Predictability

The ART Predictability Measure compares actual business value to planned business value.





## Activity: Which accelerators assist in what kind of problems?

Prepare  
8 min

Share  
3 min

- ▶ **Step 1:** In your group, review the example problems provided in your workbook and on the next slide. Select a problem that most closely matches one you experience in your team or ART.
- ▶ **Step 2:** Select which Flow Accelerators and Flow Metrics would apply to that problem.
- ▶ **Step 3:** Determine if any additional Flow Accelerators and Flow Metrics would be helpful in addressing the problem.
- ▶ **Step 4:** Be prepared to share with the class.



## Activity: Which accelerators assist in what kind of problems?

### Example Problems Surfaced

Too much focus on business Features, leading to Solution health degradation and slow development

Underlying problems with productivity; unpredictable velocity from one time period to the next

Slow time to market, causing the Customer to wait and our business to incur the cost of delay

Excess WIP, leading to increased flow time as queues build up in the system

Large amounts of waste in the system, along with bottlenecks and delays that need addressing

Low or erratic predictability, highlighting underlying problems in technology, planning, or organization performance that need addressing

# Which fundamentals assist in what kind of problems?

## Example Problems Surfaced

Too much focus on business Features leading to Solution health degradation, slowing development

Underlying problems with productivity; unpredictable velocity from one time period to the next

Slow time to market causing Customer to wait and our business to incur a cost of delay

Excess WIP leading to increased flow time as queues build up in the system

Large amounts of waste in the system along with bottlenecks and delays that need addressing

Low or erratic predictability highlights underlying problems in technology, planning, or organization performance that need addressing

## Problems

## Flow Accelerators

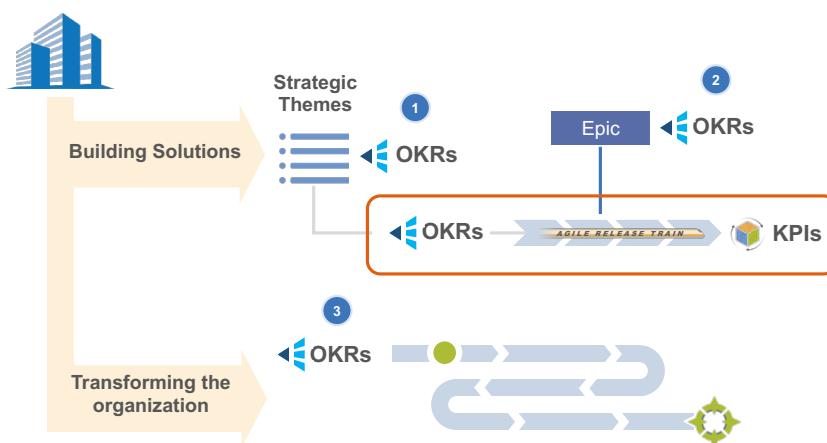
## Flow Metrics

## Notes

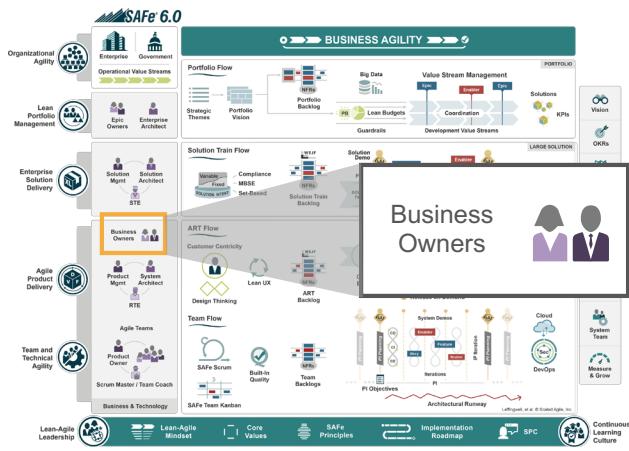
## 7.3 Improving outcomes

OKRs measure progress towards achieving outcomes

Objectives and key results (OKRs) create goals towards achieving business outcomes



## The Business Owners guide the ART towards outcomes



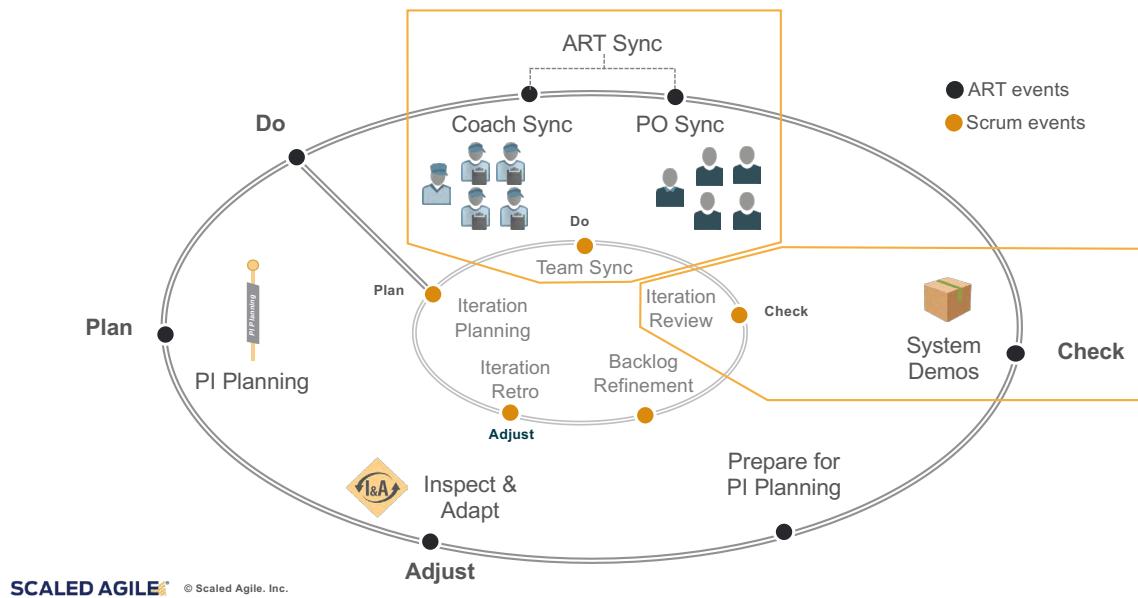
- ▶ Guide the ART toward business outcomes
- ▶ Are responsible for understanding and communicating the strategic themes that influence the train
- ▶ Identify OKRs and maintain, with ART leaders, KPI alignment
- ▶ Communicate the business context at ART events

SCALED AGILE® © Scaled Agile, Inc.

7-36

## SAFe events provide continual outcome focus

Utilize syncs alongside reviews and demos to maintain alignment on outputs leading towards outcomes.



7-37

## 7.4 Starting the improvement journey

SCALED AGILE® © Scaled Agile, Inc.

7-38

### Creating a balanced Metrics dashboard

	SAFe Portfolio	Agile Release Train	Agile Team
Outcomes	Value Stream KPIs Employee NPS	ART PI Objectives	Team PI Objectives Iteration Goals
Flow	Flow Load Flow Distribution	Flow Time Flow Efficiency Flow Predictability	Flow Velocity Flow Distribution
Competency	BA Assessment	APD Assessment DevOps Health Radar	TTA Assessment

SCALED AGILE® © Scaled Agile, Inc.

7-39

## Four critical success factors



1. Use measurement in conjunction with other discovery tools



2. Apply Metrics where they support improved decision making



3. Understand the effect of Metrics on behavior



4. Interpret Metrics carefully

SCALED AGILE® © Scaled Agile, Inc.

7-40



## Action Plan: How will we improve together?

Prepare  
15 min

Share  
10 min

- ▶ **Step 1:** As a team, discuss and answer the following prompts:
  - Identify the top three actions you will bring to the upcoming PI
  - Identify how you will emphasize relentless improvement in the team and the ART in the next PI
  - Determine what Metrics and methods you will start with
  - Discuss how you will visualize team improvements
  - Decide how you would prefer to get feedback from other teams on opportunities to improve together
- ▶ **Step 2:** Create a two-minute recap of your discussion and your decisions



SCALED AGILE® © Scaled Agile, Inc.

7-41



## Action Plan

How will we  
improve together?

## Lesson review

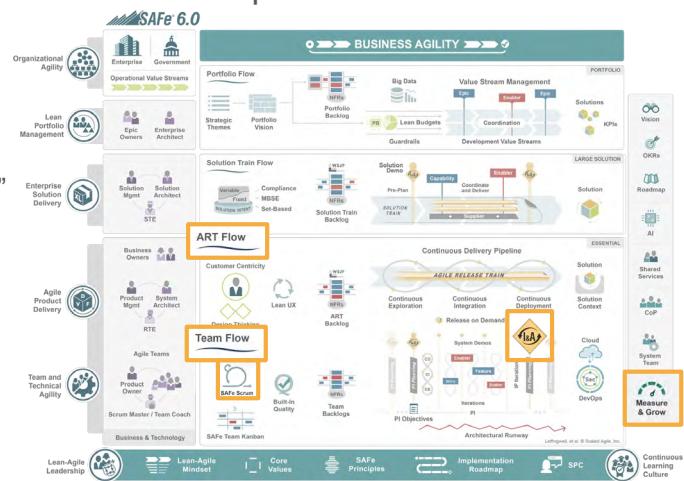
In this lesson you:

- ▶ Summarized the steps of a retrospective
- ▶ Identified the steps of I&A
- ▶ Explained the eight flow accelerators
- ▶ Summarized how to apply flow Metrics to assess the organization's ability to make value flow without interruption
- ▶ Described the purpose of a balanced Metric approach
- ▶ Identified the top three actions your team can take into the upcoming PI to develop and grow your SAFe Agile Team practices

## Articles used in this lesson

Read these Framework articles to learn more about topics covered in this lesson

- ▶ “Measure and Grow”  
<https://www.scaledagileframework.com/measure-and-grow/>
- ▶ “Make Value Flow Without Interruptions”  
<https://www.scaledagileframework.com/make-value-flow-without-interruptions/>
- ▶ “Inspect & Adapt”  
<https://www.scaledagileframework.com/inspect-and-adapt/>
- ▶ “Iteration Retrospective”  
<https://www.scaledagileframework.com/iteration-retrospective/>



## Continue your SAFe journey with the following resources:

Download "The Facilitator's Guide to SAFe: Iteration Retrospectives" for guidance in preparing to facilitate Iteration Retrospectives. <a href="https://bit.ly/Community-FGRetrospective">https://bit.ly/Community-FGRetrospective</a>	Access the Collaborate template, "Start – Stop – Continue & I Wish Retrospective," for two types of retrospectives to help teams reflect on the previous Iteration. <a href="https://bit.ly/Template-StartStopContinuelWishRetro">https://bit.ly/Template-StartStopContinuelWishRetro</a>
Download "The Facilitator's Guide to SAFe: Inspect & Adapt" for support with agenda setting, preparation guidance, and tips and tricks for facilitating I&A events. <a href="https://bit.ly/Community-FGInspectandAdapt">https://bit.ly/Community-FGInspectandAdapt</a>	Access the SAFe Collaborate template, "Root Cause Analysis and Problem-Solving Board," to gather data and visualize the problem-solving process. <a href="https://bit.ly/Template-RootCause">https://bit.ly/Template-RootCause</a>
Watch this 25-minute video, <i>Measure What Matters for Business Agility: Outcomes, Flow and Competency</i> , for a comprehensive overview of the various tools SAFe prescribes for measuring success. <a href="https://bit.ly/Video-MeasureWhatMatters">https://bit.ly/Video-MeasureWhatMatters</a>	Use the "SAFe Assessment Workshop Toolkit" to run a workshop to identify growth opportunities for your team as you work toward mastering SAFe and Business Agility. <a href="https://bit.ly/Community-ToolkitsandTemplates">https://bit.ly/Community-ToolkitsandTemplates</a>

## References

- AOE. 4. Don Reinertsen: *The Economics of Batch Size and the "Father-Egg" Story*. YouTube. April 12, 2017.  
[https://www.youtube.com/watch?v=zVASqSj\\_kvc](https://www.youtube.com/watch?v=zVASqSj_kvc).
- Kersten, Mik. *Project to Product: How to Survive and Thrive in the Age of Digital Disruption with the Flow Framework*. IT Revolution: Portland, 2018. Kindle Edition.
- Kim, Gene, Jez Humble, Patrick Debois, and John Willis. *The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organization, Second Edition*. IT Revolution: Portland, 2021. Kindle Edition.

## Lesson notes

Enter your notes below. If using a digital workbook, save your PDF often so you don't lose any of your notes.

# Lesson 8

## Practicing SAFe

SAFe® Course: Attending this course gives learners access to the SAFe® Practitioner exam and related preparation materials.

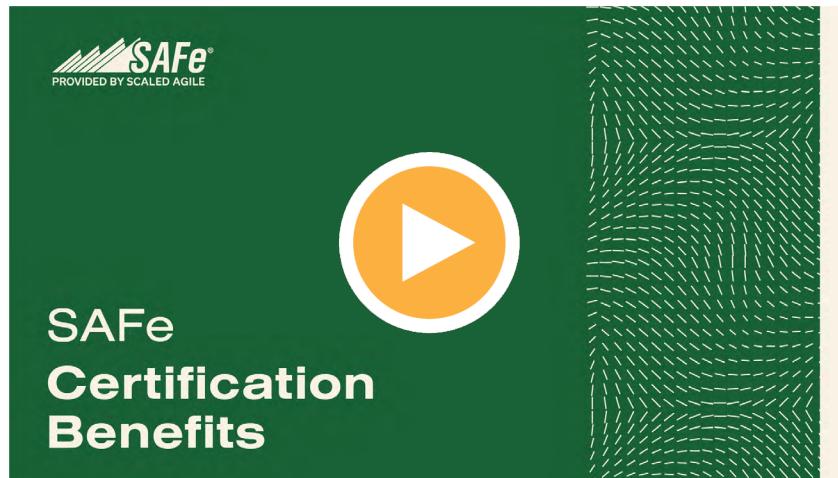


SCALED AGILE®



Video: SAFe Certification Benefits

Duration  
3 min



<https://bit.ly/Video-SAFeCertificationBenefits>

SCALED AGILE® © Scaled Agile, Inc.

8-2

## The Learning Plan is YOUR path to certification

Next Steps:



Download the workbook and access the  
**exam study guide** and **practice test**



Take the **Certification Exam** and showcase your  
**Digital Badge** to get recognized as a Certified  
SAFe Practitioner

## Feedback is a gift

Help us improve by  
completing the Course  
Feedback Survey





## Activity: My Learning Plan

Duration  
7 min

- ▶ **Step 1:** Individually, navigate to your learning plan via:
  - <https://safe.scaledagile.com/>
- ▶ **Step 2:** In your Learning Plan, navigate to and complete the course feedback survey. The survey results will remain anonymous to the instructor.
- ▶ **Step 3:** After completing the survey, review the rest of the certification prep materials. Take special note of your exam deadline and plan your timeline to prepare for and complete the exam!

SCALED AGILE® © Scaled Agile, Inc.

8-5



## Video: Welcome to SAFe Studio

Duration  
3 min



<https://bit.ly/Video-WelcomeSAFeStudio>

SCALED AGILE® © Scaled Agile, Inc.

8-6

## SAFe Studio Resources



Online Learning



Videos



Toolkits



My SAFe Events



SAFe FAQs



SAFe Assessments



SAFe Forums

SCALED AGILE® © Scaled Agile, Inc.

8-7



### Activity: SAFe Studio Exploration

Duration  
7 min

- ▶ **Step 1:** Using the sidebar navigation, explore each area of SAFe Studio to locate the following:
  - Identify a video about a SAFe event you're curious to learn more about.
  - Select an online learning of interest.
- ▶ **Step 2:** Navigate to the SAFe Forum and make a post. You could introduce yourself, ask a question from the class parking lot, or respond to a post from another SAFe Studio user.

SCALED AGILE® © Scaled Agile, Inc.

8-8



**Good luck on your  
SAFe Practice  
with SAFe Studio**

<https://safe.scaledagile.com/>

™

8-9

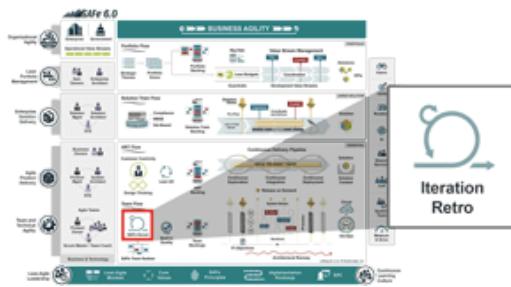
# SAFe Glossary



## SAFe Glossary:

Visit the Scaled Agile Framework site ([www.scaledagileframework.com/glossary/](http://www.scaledagileframework.com/glossary/)) to download glossaries translated into other languages.

## + Framework



“ At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

—Agile Manifesto

# Iteration Retrospective



**Note:** For more on SAFe Scrum, please read the additional Framework articles in the Scrum series, including [SAFe Scrum](#), [Scrum Master/Team Coach](#), [Iterations](#), [Iteration Planning](#), [Iteration Goals](#), and [Iteration Review](#).

The Iteration Retrospective is a regular event where the team members discuss the results of the iteration, review their practices, and identify ways to improve.

Agile teams that apply Scrum hold a retrospective at the end of each iteration. Each retrospective seeks to uncover what's working well, what's not, and what the team can do better next iteration. Agile Teams applying Kanban hold retrospectives whenever needed, but they often use the same cadence as Scrum teams.

## Details

[Agile Teams](#) reflect on the completed iteration and derive new ideas to improve the team's process. This reflection helps instill the concept of relentless improvement—a fundamental tenet of the SAFe [Core Values](#). And it helps ensure that every retrospective yields some minor improvements.

# Inputs and Outputs of the Iteration Retrospective

Inputs to iteration retrospective may include:

- [Iteration goals](#)
- The team's increment
- List of improvement stories identified and the actions taken since the last retrospective
- Collection of agreed-to iteration metrics

A successful iteration retrospective event delivers the following outputs:

- Creation of a few improvement [Stories](#)
- Updated [Team Backlog](#)

## Preparation

The [Scrum Master/Team Coach](#) should offer the team a way to give feedback on the techniques used at the last retrospective to improve the event's facilitation.

## Process

The entire team participates in the retrospective. The Scrum Master/Team Coach starts the event by introducing the goals of the retrospective, the agenda, and the facilitation format for the meeting.

The team reviews and discusses the metrics the team has agreed to and determines any actions to take, such as further investigation or analysis.

Team members may write their thoughts on a flip chart or the digital tool designated for the retrospective. Following are several popular formats for getting started with qualitative feedback for the iteration (also see [1], [3], [4], [5]):

- **Individual.** Individually write post-it notes and then group similar stickies
- **Appreciation.** Note whether someone has been helpful to the team
- **Conceptual.** Choose one word to describe the iteration
- **Rating.** Rate the iteration on a scale of one to five, and then brainstorm how to make the next one a five
- **Simple.** Open a discussion and record the results under the three headings described next.

The last format is the most conventional. The team creates stickies to record 1) *what went well*, 2) *what did not*, and 3) what to *do better next time* and then facilitates an open brainstorming session.

Of course, teams can examine more than three questions, match the questions to themes, and can use this time to explore new ways of getting feedback and improvement items. The team can adopt the simple format quickly, making all accomplishments and challenges visible. Optionally, the team can use different headings for the simple retrospective, as shown in Figure 1.

The last part of the process is to hold a team vote on the action items and suggestions for improvement to add to the [Team Backlog](#). If a significant issue is identified, the team may perform a root cause analysis, discuss potential corrective actions, and enter improvement stories into the team backlog. (See the '*problem-solving*' section in the [Inspect & Adapt](#) article for more information on root cause analysis.)



Figure 1. One team's iteration retrospective results using a simple 4Ls format [1]

## Attendees

Attendees of the iteration retrospective event include:

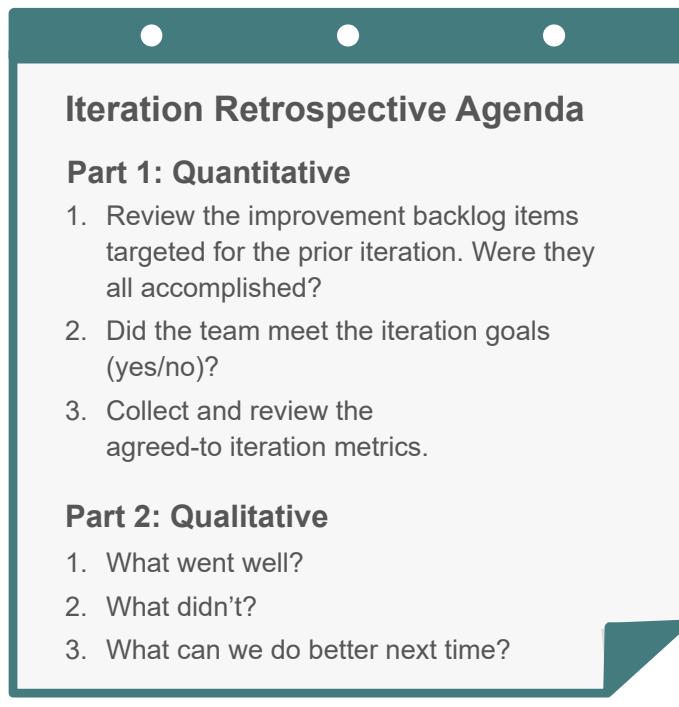
- The [Product Owner \(PO\)](#)
- Scrum Master/Team Coach
- All team members and other stakeholders or subject matter experts

- Other stakeholders, which may include representatives from other Agile Teams or trains.

Scrum Masters/Team Coaches or POs typically facilitate iteration retrospectives for the team, ensuring they stay within the agreed event timebox.

## Agenda

The timebox for the event is a maximum of one hour for a two-week iteration. An *example* iteration retrospective agenda (Figure 2) and a description of each item follow.



© Scaled Agile, Inc.

Figure 2. Example retrospective agenda

The retrospective generally has two parts:

1. **Quantitative review.** The team assesses if the iteration goals were met using a binary (yes or no) measure. Agile Teams collect and review iteration metrics for transparency and to assist with process improvement. Examples include [flow metrics](#), such as flow velocity, load and distribution, defects addressed, and automated test coverage. This data also provides the context for the qualitative section that follows.
2. **Qualitative review.** During the qualitative portion, the team reviews the improvement stories they identified in the last retrospective. They then analyze the current process, focusing on finding one or two things they can do better in the next iteration. Since many improvement items have a significant scope, the team can divide them into smaller improvement stories to be implemented incrementally in subsequent iterations.

# Other Improvement Opportunities

As organizations begin implementing [DevOps](#) and a [Continuous Delivery Pipeline](#), Agile Teams will have many improvement opportunities, including:

- Applying [Built-In Quality](#)
- Enhancing test automation (including [test-driven development](#) and [behavior-driven development](#)) and [Continuous Integration](#)
- Automating the deployment process
- Decoupling deployment from release (see [Release on Demand](#))
- Building telemetry and recovery techniques into systems

[Lean-Agile Leaders](#) support the time teams need during each iteration to focus on cultivating new skills and addressing improvement opportunities. The [Innovation and Planning \(IP\) Iteration](#) also offers opportunities for teams to advance their skills.

## Keeping Team Members Engaged

Team members are more likely to remain more engaged when retrospective formats are new and varied. For example, teams may choose to rotate the responsibility for facilitating retrospectives. One fun practice is allowing each person to select the retrospective format when it's their turn to lead. This practice creates shared ownership of the process and keeps the retrospective interesting. Having different themes and templates to focus the retrospective on specific topics can also be fun and valuable. The SAFe Collaborate tool, accessed from the community platform, provides many retrospective templates.

---

## Learn More

[1] Sprint retrospective techniques. <https://waynedgrant.wordpress.com/2014/02/09/sprint-retrospective-techniques-3/>

[2] Derby, Esther, and Diana Larson. *Agile Retrospectives: Making Good Teams Great*. Pragmatic Bookshelf, 2006.

[3] Leffingwell, Dean. *Scaling Software Agility: Best Practices for Large Enterprises*. Addison-Wesley, 2007.

[4] Fun Retrospectives. [www.funretrospectives.com](http://www.funretrospectives.com)

[5] TastyCupcakes.org. <http://tastycupcakes.org/tag/retrospective/>

Last update: 14 March 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm

Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



[Home](#) » Inspect and Adapt

## Inspect and Adapt

**“** Kaizen is about changing the way things are. If you assume that things are all right the way they are, you can't do kaizen. So change something!

—Taiichi Ohno

**Definition:** The Inspect and Adapt (I&A) is a significant event held at the end of each PI, where the current state of the Solution is demonstrated and evaluated. Teams then reflect and identify improvement backlog items via a structured problem-solving workshop.

## Summary

The Inspect and Adapt (I&A) is an Agile Release Train (ART) event. It engages all ART stakeholders alongside the Agile Teams in reflecting on progress and identifying improvements. It is structured into three parts, the PI System Demo, a quantitative and qualitative measurement review, and a retrospective and problem-solving workshop. The goal of the Inspect and Adapt is to make the ART better by identifying and solving big issues. This is done by focusing on what has actually happened over the last PI rather than what was supposed to happen.

## What is Inspect and Adapt in SAFe?

Inspect and Adapt (I&A) is a significant event for continuous improvement. Held at the end of each PI, this event is structured to demonstrate progress, review current and meaningful measurements, and create actionable improvements.

The Agile Manifesto emphasizes the importance of continuous improvement through the following principle: “At regular intervals, the team reflects on how to become more effective, then

tunes and adjusts its behavior accordingly." The I&A amplifies the original principle and creates this time for the entire ART and key stakeholders to engage in continuous improvement.

In addition, SAFe includes 'relentless improvement' as one of the four SAFe Core Values. While opportunities for improvement occur continuously throughout the PI, the I&A creates structure, cadence, and synchronization to help ensure time is set aside to identify improvements across the Agile Release Train (ART).

The I&A includes the active involvement of all ART members and stakeholders. The result is a set of improvement backlog items that go into the ART Backlog for consideration in the next PI Planning event. In this way, every ART improves every PI. When multiple ARTs are involved in building a large solution, a similar event can be held to improve their alignment.

The I&A empowers the Agile Teams to take ownership of their work, celebrating achievements whilst also sharing challenges that create learning and change. It creates an important feedback loop, ensuring the organization moves towards its business and customer goals.

Read more about relentless improvement as a Core Value of SAFe:

Core Values

## How to run an Inspect and Adapt event?

The I&A event consists of three parts:

1. PI System Demo
2. Quantitative and qualitative measurement
3. Retrospective and problem-solving workshop

Participants in the I&A should be, wherever possible, all the people involved in building the solution. For an ART, this includes:

- The Agile Teams
- Release Train Engineer (RTE)
- System and Solution Architects
- Product Management, Business Owners, and other stakeholders

Additionally, Solution Train stakeholders may also attend this event.

## Part 1: PI System Demo

The PI System Demo showcases all features developed during the PI. It is formal and requires preparation but typically does not exceed one hour.

Additionally, before or during the demo, Business Owners and Agile Teams assess the actual business value of the team's PI Objectives to calculate an achievement score. This score is used over time as part of the predictability measure for the ART. This is assessed by comparing each team's planned versus actual business value, aiming for a reliability range of 80-100 percent to ensure effective planning for the business and stakeholders. This measure excludes uncommitted objectives from planning but includes them in actual achievements.

Objectives for PI 3		Business Value	
		Plan	Actual
▪ Structured locations and validation of locations		7	7
▪ Build and demonstrate a proof of concept for context images		8	8
▪ Implement negative triangulation by: tags, companies and people		8	6
▪ Speed up indexing by 50%		10	5
▪ Index 1.2 billion more web pages		10	8
▪ Extract and build URL abstracts		7	7
<b>Uncommitted Objectives</b>			
▪ Fuzzy search by full name		7	0
▪ Improve tag quality to 80% relevance		4	4
<b>Totals</b>		<b>50</b>	<b>45</b>
<b>% Achievement: 90%</b>			

© Scaled Agile, Inc.

Figure 1. Scoring actual business value for each team

Read more tips for facilitating System Demo throughout the PI:

System Demo

## Part 2: Quantitative and Qualitative Measurement

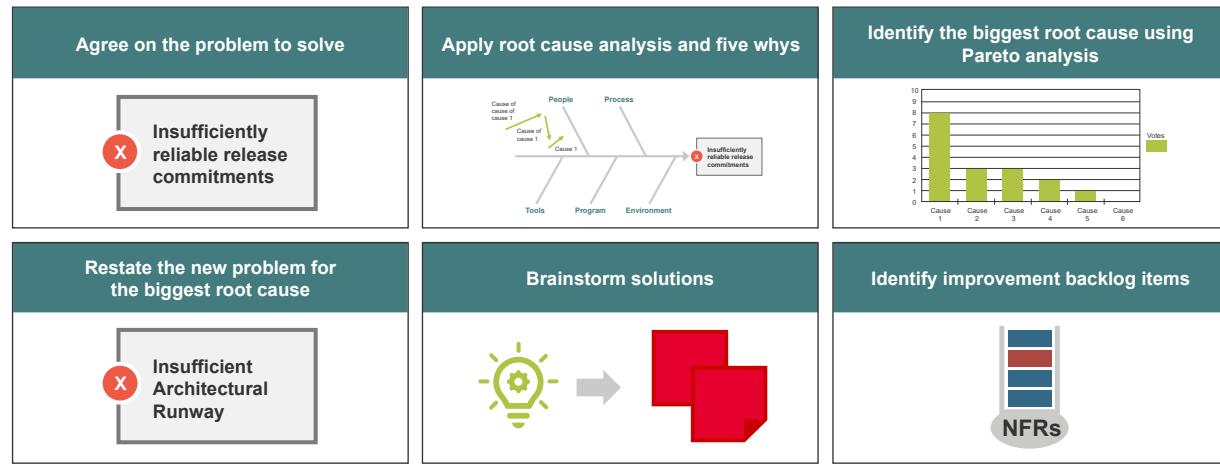
In the second part of the I&A event, the RTE gathers and analyzes relevant performance metrics, preparing to present them to the ART to evaluate teams' effectiveness. One of these metrics is the ART's predictability measure. Additionally, ARTs review measures related to the flow of value across the ART and other measures that are important within the context of the ART. Common examples include the SAFe flow metrics, value stream KPIs, and eNPS. These measures enable a meaningful Part 3 of Inspect and Adapt.

## Part 3: Problem-solving workshop

As the final part of the I&A, the ART holds a structured, root-cause problem-solving workshop to address systemic problems. Root cause analysis provides a set of problem-solving tools used to identify the actual causes of a problem rather than just fixing the symptoms. The RTE typically facilitates the session in two hours or less.

A brief retrospective is used to help attendees identify key issues they want to tackle in the problem-solving workshop. This often leads to the formation of new, cross-functional groups that bring diverse perspectives to the issues they want to spend time resolving. All ART stakeholders, such as Business Owners, available customers, and leadership, participate in the groups. They can help remove obstacles beyond the ART's control, ensuring a comprehensive approach to problem-solving.

Figure 2 illustrates the steps in the problem-solving workshop.



© Scaled Agile, Inc.

Figure 2. Problem-solving workshop format

The following sections describe each step of the process.

- **Agree on the problem(s) to solve** – The groups have self-selected the problem they want to address. Each group spends a few minutes clearly stating the problem, highlighting the 'what,' 'where,' 'when,' and 'impact' as concisely as possible. Figure 3 illustrates a well-written problem statement.

What

When

Impact

Where

We discovered three significant design problems in the October deployment of the new EMV vehicles at the Thrills Amusement Park.

The design flaws caused us to recall the vehicles and invest three months in materials, redesign, and testing. We delivered late, paid substantial penalties, and lost credibility with the customer.

*Concept contributed by Beth Miller*

© Scaled Agile, Inc.

Figure 3. Example problem statement

- **Perform root cause analysis** – Also known as an Ishikawa Diagram, a fishbone diagram is a visual tool for exploring the causes of specific events or sources of variation in a process. Team members brainstorm potential causes of a problem, using the fishbone categories seen below, to consider the problem from different points of view.

They then use the 5 Whys technique to explore each cause's root by asking 'why' five times until a suitable root cause is identified.

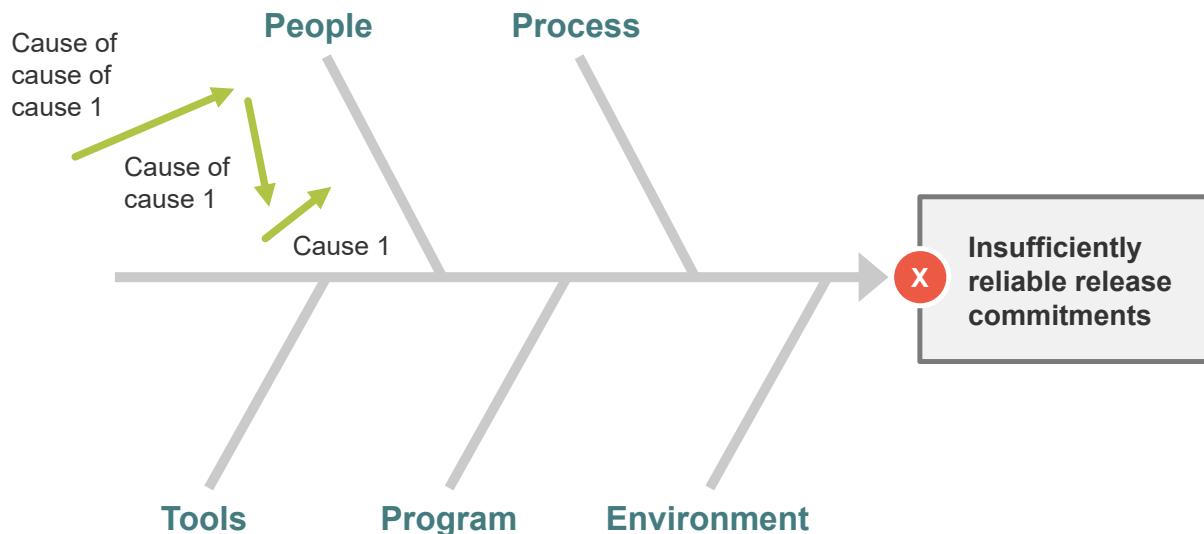


Figure 4. Fishbone diagram with primary sources identified

**People** – Problems that stem from the way people operate

**Process** – Problems that stem from the processes in place

**Tools** – Problems that stem from the way tools are used

**Program** – Problems that stem from the way the teams and ARTs are structured

**Environment** – Problems that stem from the organizational culture and environment external to the ART

- **Identify the biggest root cause** – Pareto Analysis, or the 80/20 rule, helps identify the few causes (20%) that lead to the majority (80%) of a problem. It is useful when dealing with complex issues and many potential actions. Team members vote on the main causes, usually through dot voting, and results are displayed in a Pareto chart to highlight the most significant cause.
- **Restate the new problem** – The next step is to pick the cause with the most votes and restate it clearly as a problem. Restating it should take only a few minutes, as the teams clearly understand the root cause.
- **Brainstorm solutions** – At this point, the restated problem will start to imply some potential solutions. The team brainstorms as many possible corrective actions as possible within a short timebox. When brainstorming, aim to generate numerous ideas without criticism, letting imagination fly to explore and combine possibilities.
- **Create improvement backlog items** – Each group votes on the top three solutions. These solutions are shared with everyone, written up as improvement backlog items, and added to the ART backlog.

The combined process of I&A makes problem-solving a regular, systematic activity, driving continuous improvement.

**Note on large solutions:** To efficiently address problems, large solutions may necessitate a separate I&A event attended by selected key stakeholders, including primary stakeholders and representatives from various ARTs and suppliers.

Read more about the different applications of measurement in SAFe:

Measure and Grow

## How to apply the improvements identified in the Inspect and Adapt?

The I&A represents one component of the ART's continuous improvement journey. To benefit from it, the ART must implement the improvements it identifies. Participants often identify multiple positive and realistic actions for improvement. It isn't possible to do them all at once. The ART leadership and Agile Teams must prioritize these actions during the preparation for PI Planning and in regular ART and team backlog reviews. They do this by looking at what will have

the most positive impact on achieving the desired PI Objectives. The feasibility, the ART's future direction, and the existing backlog of work are all considered.

Plans are developed for prioritized improvements, which become work artifacts for teams or stakeholders. Designated responsibilities and a communication plan for how progress will be shared are also developed. Any ART member, including key stakeholders, can assume the roles of owners for these improvements. This approach guarantees the execution of improvements and keeps the ART and wider organization informed.

Implementing changes to workflows, tools, or practices often demands communication and training, not just implementation. Establishing metrics to measure the impact of the improvements may be necessary. This validates the effectiveness of what initially seems like a great idea, ensuring efforts focus on ideas that achieve desired outcomes.

The Inspect and Adapt, as well as the actions taken following the event, drive continuous improvement and collaboration. The practice ultimately delivers better business outcomes and happier employees.

## Login to access SAFe Studio practical tools



PI Execution Toolkit

### In this article

[What is Inspect and Adapt in SAFe?](#)

[How to run an Inspect and Adapt event?](#)

[How to apply the improvements identified in the Inspect and Adapt?](#)

### Key Takeaways

- The Inspect and Adapt (I&A) event supports relentless improvement across the Agile Release Train (ART) and its stakeholders.
- The I&A event has three main parts that work together to achieve the outcome of actionable improvements.

- ART members are responsible for identifying improvements inside of a problem-solving workshop.
- The ART takes ownership of the improvement actions and plans them into upcoming PIs.

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

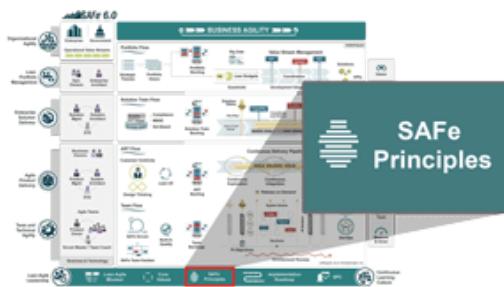
[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



## + Framework



“ When we start thinking about ways to line up all of the essential steps needed to get a job done into a steady, continuous flow, it changes everything.

— James P. Womack and Daniel T. Jones, *Lean Thinking*

## Principle 6 – Make Value Flow without Interruptions



### Note: About the Flow Article Series

SAFe is a flow-based system. As such, any interruptions to flow must be identified and addressed systematically to enable continuous value delivery. While flow-based guidance is embedded throughout SAFe, a special collection of eight articles directly addresses impediments to flow. These are [Value Stream Management](#), Principle #6- Make value flow without interruptions, [Team Flow](#), [ART Flow](#), [Solution Train Flow](#), [Portfolio Flow](#), the extended Guidance articles [Accelerating Flow with SAFe](#), and [Coaching Flow](#).

Enterprises must respond quickly to market changes to remain competitive in the digital age. Delivering a continuous flow of value to customers in the ‘shortest sustainable lead time’ is the central theme of SAFe. Doing so requires moving new system features through the development value stream as quickly as possible. Achieving continuous flow requires a new way of working that eliminates the traditional start-stop-start project cycle and the waterfall phase gates that hinder flow.

The principles and practices that enable the uninterrupted flow of value in SAFe are integral to the [Lean-Agile Mindset](#), [Value Stream Management](#), and Lean Thinking [1], which can be summarized as:

- Precisely specify value by product
- Identify the value stream for each product
- *Make value flow without interruptions*
- Let the customer pull value from the producer
- Pursue perfection

This article, SAFe Principle 6, describes how to *make value flow without interruptions* by introducing 'eight flow accelerators' that can be used to improve flow at any level of SAFe.

## What is Flow?

First, it's essential to understand what SAFe means by *flow*. Flow occurs when there is a smooth, linear, and fast movement of work product from step to step in a relevant value stream.

While the details of any flow system are based on its context, all flow systems have eight common properties, as illustrated in Figure 1.

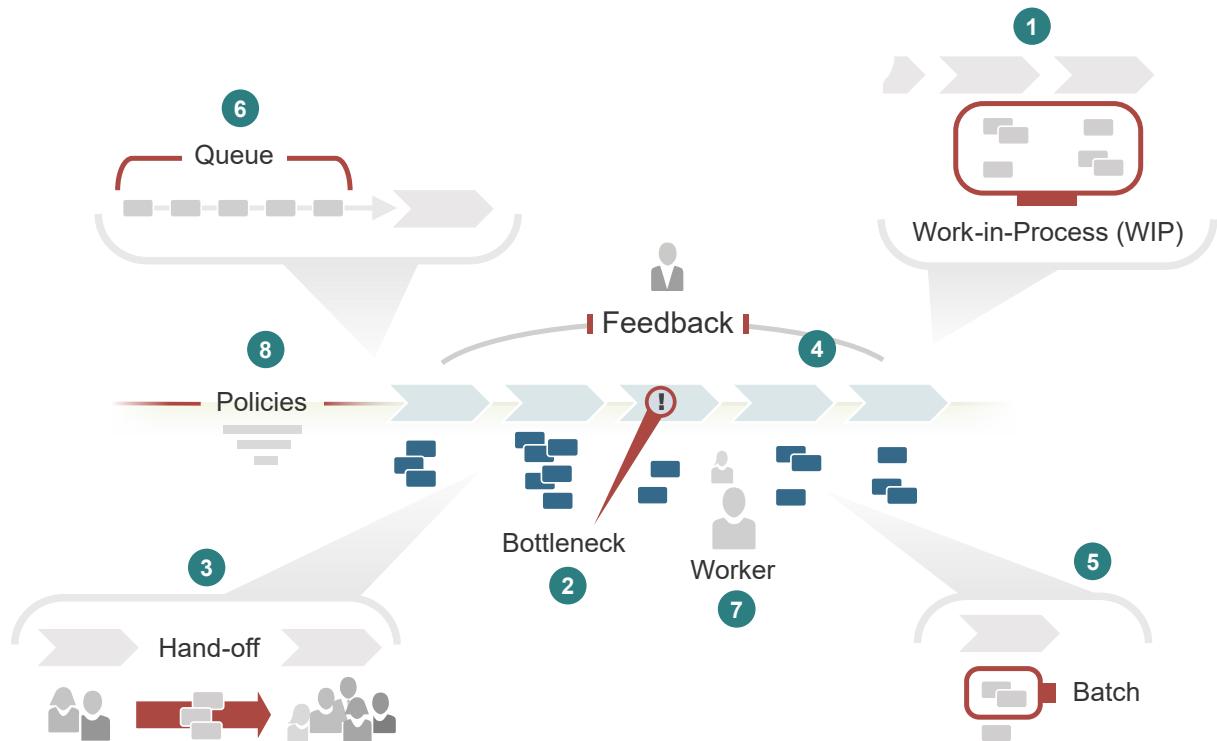


Figure 1. Eight properties of a flow system

Each is described briefly below:

1. **Work in process.** There is always some work in process in the system; if there weren't, there could be no flow of value.
2. **Bottlenecks.** In every flow system, one or more bottlenecks effectively limit the flow through the entire system.
3. **Handoffs.** Handoffs wouldn't be necessary if one person could do all the work. But in any material flow system, different individuals and teams will have different skills and responsibilities. Each plays its part in moving a work item through the system.
4. **Feedback.** Customer and stakeholder feedback is integral to efficient and effective outcomes. Ideally, feedback happens throughout the entire process.
5. **Batch.** As any system has a finite capacity, all the work can't be done at once. Therefore, work through the system occurs in batches designed to be as efficient as possible.
6. **Queue.** It all starts with a set of work items to be done. In addition, each value stream needs a prioritizing mechanism to sequence the work for the best value.
7. **Worker.** People do the critical work of moving work items from one state to another.
8. **Policies.** Policies are integral to flow. They may be local policies — like team-based policies that determine how a work item moves from step to step— or global policies like those that govern how work is performed within the company.

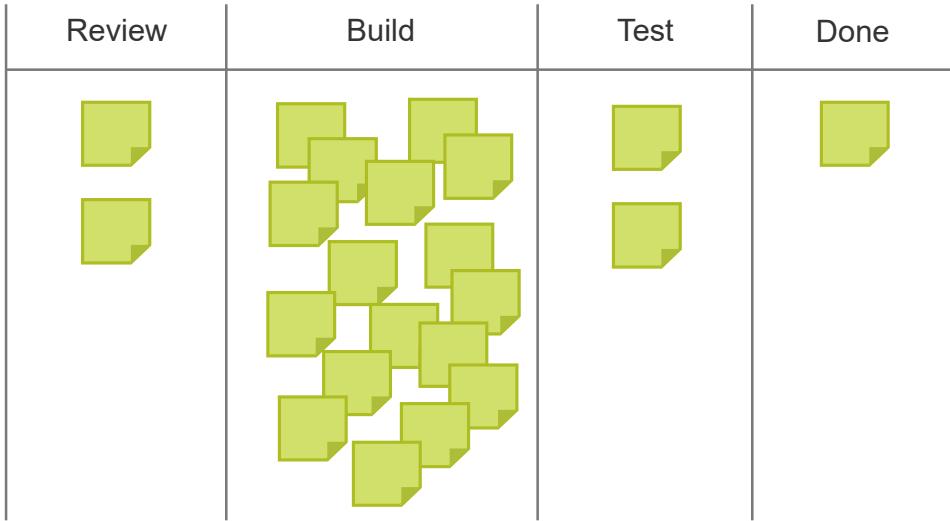
## The Eight Flow Accelerators

Each flow property is subject to optimizations, and often many steps encounter unnecessary delays, bottlenecks, and other impediments to flow. *Making value flow without interruptions* can best be achieved by adopting the eight 'flow accelerators' described in this article. These powerful accelerators of value are relevant to all Framework levels, but the challenges differ for each. An individual SAFe article discusses how these accelerators apply to each flow domain: [Team Flow](#), [ART Flow](#), [Solution Train Flow](#), and [Portfolio Flow](#).

### #1 Visualize and Limit WIP

Overloading teams and ARTs with more work than can be reasonably accomplished is a common and pernicious problem. Too much work in process (WIP) confuses priorities, causes frequent context switching, and increases overhead. It overloads people, scatters focus on immediate tasks, reduces productivity and throughput, and increases wait times for new functionality. Like a highway at rush hour, there is simply no upside to having more work in a system than the system can handle.

The first corrective action is to make the current WIP visible to all stakeholders. Figure 2 shows a simple Kanban board that illustrates the total amount of WIP and the process state of each work item. This Kanban serves as an initial process diagnostic, showing the current bottlenecks. Often, simply visualizing the current volume of work is the wake-up call that causes the organization to address the systemic problems of too much work and too little flow.



© Scaled Agile, Inc.

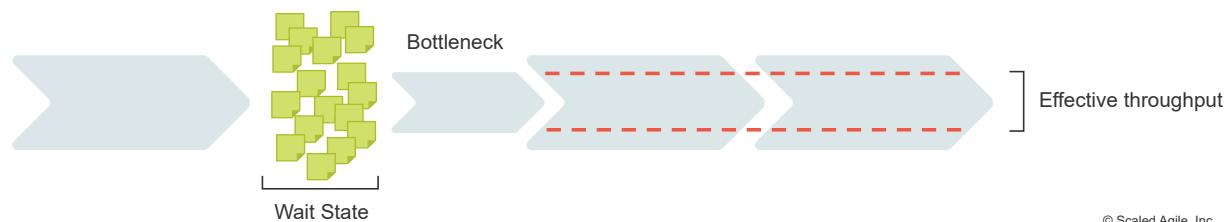
**Figure 2. Kanban boards make excessive work-in-process (WIP) visible**

The following action is balancing the amount of WIP against the available development capacity. This is done by establishing—and continually adjusting—WIP limits for the relevant states. No new work is started when any workflow state reaches its WIP limit. This matches demand to capacity and increases flow through the system.

Limiting WIP, however, requires knowledge, discipline, and commitment. It may even seem counterintuitive to those who believe that the more work you put into the system, the more you get out. That can be true up to a point, but when the system becomes overloaded, throughput *decreases* dramatically. Indeed, there is no substitute for effectively managing WIP.

## #2 Address Bottlenecks

Bottlenecks occur wherever people or resources (systems, materials, and so on) in the flow of value experience demand greater than the available capacity. Examples include a shortage of a specialized skill (such as a data scientist), insufficient processing power for the build servers in the CI/CD pipeline, or a silicon supply shortage for building the integrated circuits of a cyber-physical system. Work piles up at a bottleneck and limits the effective throughput of value, as Figure 3 illustrates.



**Figure 3. Bottlenecks reduce the flow of value through the value stream**

Upstream processes are blocked from moving value. Downstream processes are starved and waiting. Bottlenecks cause the value stream to operate slowly and uneconomically, far below its potential capacity. This is emphasized in the Theory of Constraints (Goldratt [2], [3]), which posits that the throughput of any flow system is limited by the capacity of a dominant constraint, or *bottleneck*. By this theory, investment in optimizing the system at any other point than the dominant constraint is waste, as it will not improve the throughput of the system.

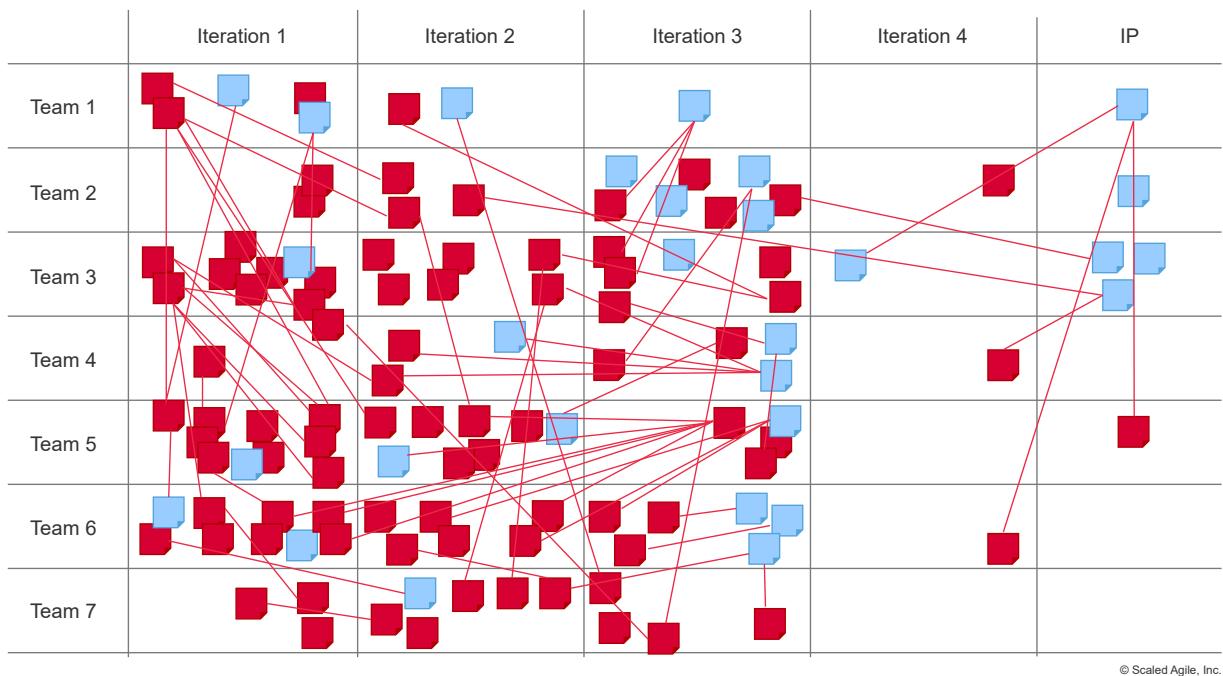
No matter the theory, bottlenecks must be addressed, by adding additional skills, people, or other resources at the bottleneck step. While that is not always easy to do, (there are reasons why the bottleneck is a bottleneck), eliminating dominant bottlenecks must be a primary focus as throughput will not increase until the bottleneck is addressed.

In the meantime, however, the complex workflows in solution development provide other options. It's often the case that not all the work is single-threaded (must pass through the bottleneck step). There is often other work (different features, etc.) that does not have to pass through the bottleneck. In this case, teams can selectively pick and deliver other valuable work, thus increasing value throughput, while the dominant bottleneck is being addressed. In addition, relentless improvement drives us to improve wherever we can do so, and knowledge workers take pride in improving the processes that are under their control. In the end, the efficiency of every step matters.

But the fact is that bottlenecks limit throughput and negatively affect economic outcomes.

### #3 Minimize Handoffs and Dependencies

Handoffs occur whenever there is a separation between knowledge, responsibility, action, and feedback [4]. For example, dependencies happen between teams when the work of one team cannot continue until related work by another team is completed (see Figure 4). Both result in development waste in the form of wait states in the flow of value. They can also lead to rework as the knowledge transfer is likely imperfect, causing further delays.



**Figure 4. Excessive handoffs and dependencies made visible on the ART planning board**

The best solution to overcome handoffs and dependencies is to create teams and ARTs with all the knowledge, resources, skills, and decision-making authority to create an end-to-end flow of value. However, unhealthy dependencies and handoffs can still occur even when teams and trains have all the skills to deliver end-to-end value. This can happen when there is a series of handoffs between Agile Team members or when there are delays while waiting on decisions to be made outside the ART. Activities like value stream mapping, retros, and the I&A problem-solving workshop can help identify the root causes and potential solutions.

## #4 Get Faster Feedback

Learning is the foundation of improvement and the engine that powers product development [4]. Doing this as fast as possible speeds up and improves the overall development process. The goal is to get positive and negative feedback into the development process as early as possible.

However, we often discover that getting early feedback can be difficult for a variety of reasons, for example:

- Lack of direct access to customers
- Delays in the development value stream
- Late or infrequent integration results in discovering hidden work and defects
- Developing more functionality than what's needed
- Building the wrong things or more functionality than what's needed

Fast feedback is generally achieved by applying the basic Plan-Do-Check-Adjust (PDCA) learning cycle. However, to accelerate flow further, we've found that more needs to be done, for example:

- Applying customer centricity and design thinking as part of product development and engaging with customers frequently
- Making improvements to the continuous delivery pipeline, including build and test automation, test-first practices, and continuous integration
- Keeping work items small results in faster working increments of value.
- Using built-in quality practices, mob work, pairing, and swarming to increase team cohesion and focus on finishing one backlog item at a time
- Upholding a solid Definition of Done (DoD) to help teams work together to finish Increments of value and share knowledge
- Use “stop-the-line” to fix problems when they occur so they don’t pile up.

Generally, solution builders need two types of feedback from each PDCA cycle (Figure 5):

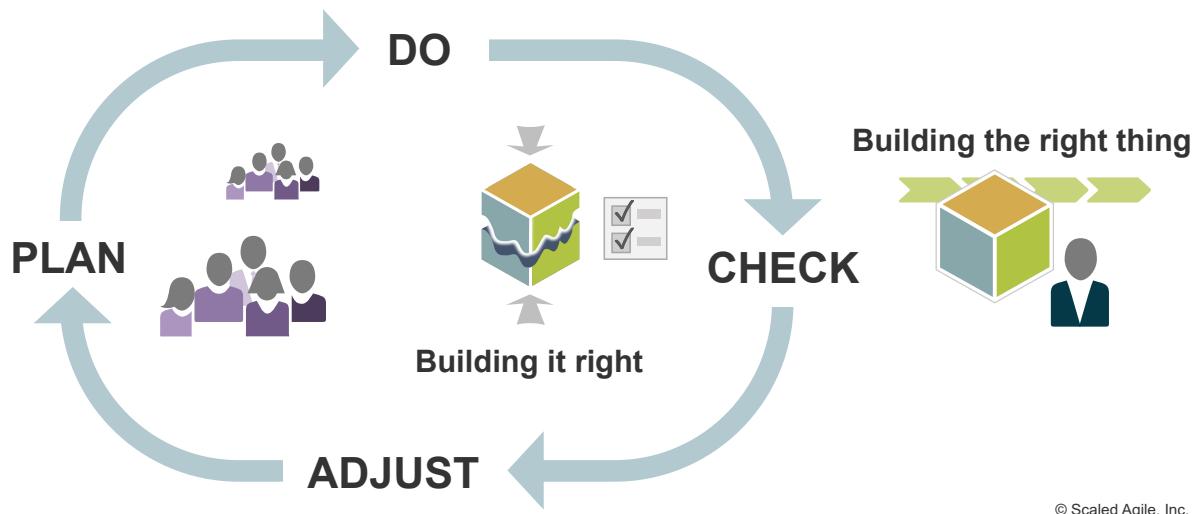


Figure 5: Every PDCA cycle collects two kinds of feedback

1. **Feedback about building the right thing.** This feedback can only come from those users, customers, and economic stakeholders who can measure a solution’s actual value. Each PDCA cycle is an opportunity for this learning, from early mockups and storyboards to system demos during development to feedback on pre-releases and deployed systems in production.
2. **Feedback about building it right.** Innovative systems constantly push the bounds of technology and the developers’ skills. Each PDCA cycle also evaluates if the right technology is applied to optimally solve the customer’s problem and meet the critical nonfunctional requirements (system ‘ilities’) that characterize robust and effective solutions.

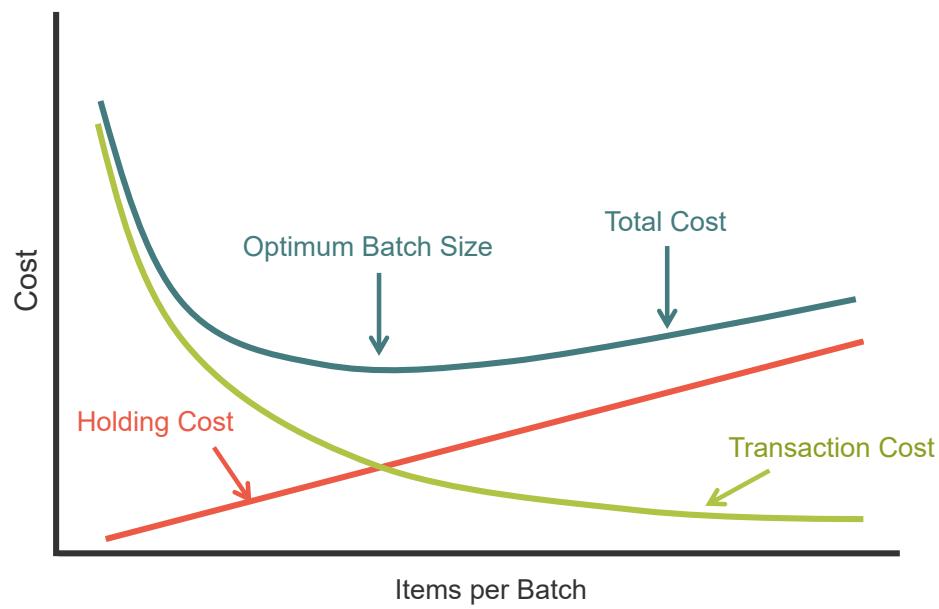
Creating the mechanisms and processes to collect a wide range of data is the critical first step to promoting flow with faster feedback, but it doesn’t stop there. The information should be quickly

analyzed and evaluated to make effective adjustments and initiate the next PDCA cycle based on these learnings.

## #5 Work in Smaller Batches

Faster feedback is one of the primary reasons for working in smaller batches. The smaller the size, the faster teams can collect and evaluate the feedback to adjust. In addition, smaller batches reduce WIP by limiting the number of requirements, designs, code, tests, and other work items moving through the system at any point. Smaller batches go through the system faster and with less variability, fostering faster learning. Moreover, since each item in the batch has some variability, larger batches accumulate more variability.

The economically optimal batch size depends on the holding cost (the cost for delayed feedback, inventory decay, delayed value delivery, and so on) and the transaction cost (the cost of preparing and implementing the batch). Figure 6 illustrates the tradeoff u-curve optimization for batch size [5].



© Scaled Agile, Inc.

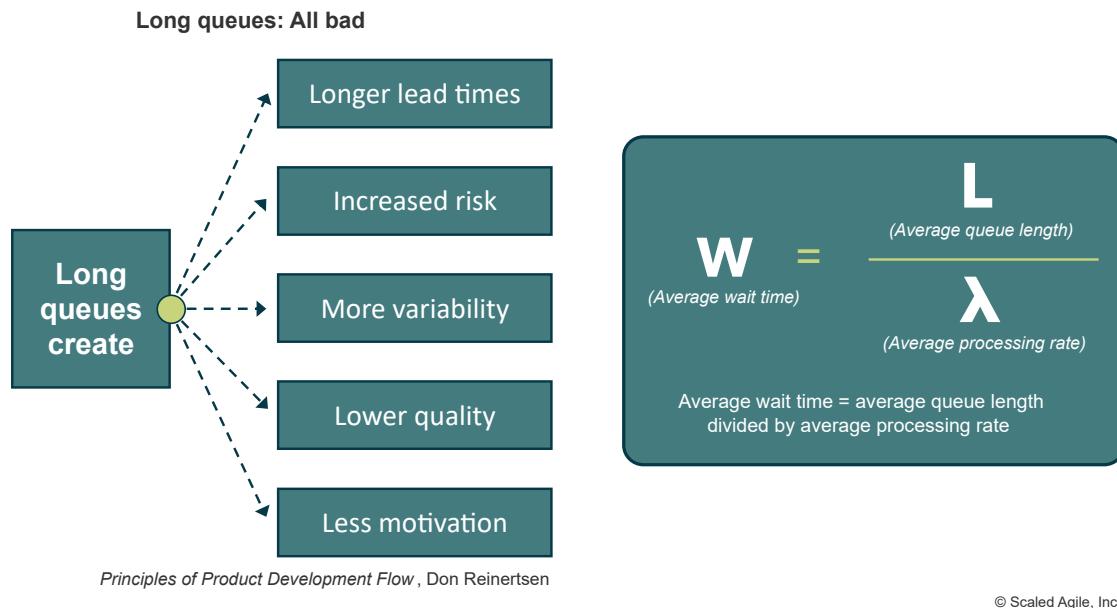
Figure 6. U-curve optimization for batch size showing tradeoffs between holding cost and transaction cost

To improve the economics of processing smaller batches—teams should focus on reducing the transaction costs—resulting in higher throughput for a batch of a given size. *Reducing batch size typically involves investment in automating the Continuous Delivery Pipeline*, including infrastructure and automation, continuous integration, builds, regression testing, and more. Shorter iterations and PIs also help to reduce batch size.

## #6 Reduce Queue Length

Reducing the length of the queue that is feeding the system is another critical way to accelerate flow. As we have all experienced, long queues are fundamentally bad. They introduce waste,

delays, and information decay. In addition, Little's Law (Figure 7) informs us that the *average wait time* equals the *average queue length* divided by the *average processing rate*. (While this might sound complicated, even the line at Starbucks illustrates that.) Therefore, assuming any average processing rate, the longer the queue, the longer the wait.



**Figure 7. Long queues slow service and flow; Little's Law predicts the average wait time**

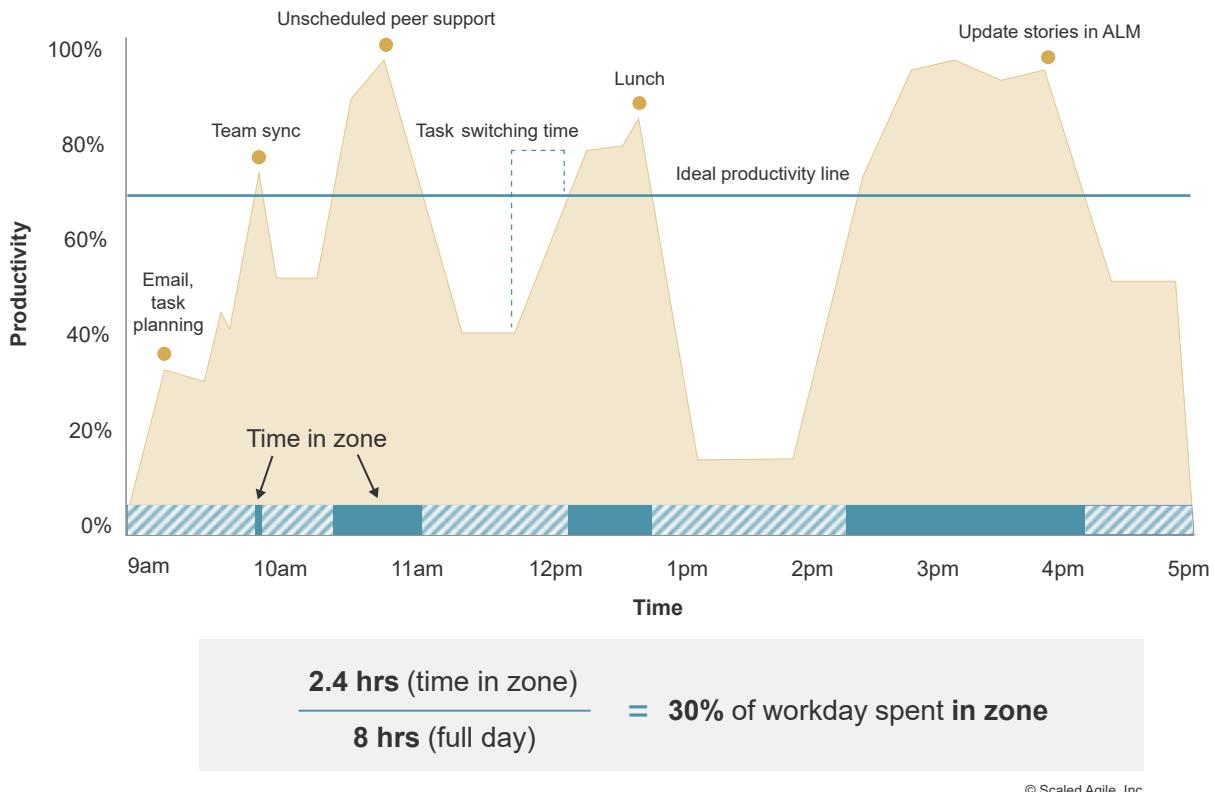
For solution development, the longer the queue of committed work awaiting implementation, the longer the wait time for new features, regardless of the team's efficiency. For example, suppose an ART has an average flow velocity of 10 features per quarter and a committed backlog of 30. In that case, the customer may have to wait as long as three quarters before any new features can start developing. This example explains why queues are fundamentally bad and can significantly delay the ability to respond to customer needs.

Reducing queue length decreases delays, reduces waste, increases flow, and improves predictability. It's a requisite for faster service and a more consistent flow of value.

## #7 Optimize Time 'In the Zone'

Being 'in the zone' (also described as being in a 'flow state') is an engaged mental state of extreme focus on an activity where the work feels effortless and time passes quickly. People and teams in the zone demonstrate higher creativity, productivity, happiness, and fulfillment. Getting into this mental state requires uninterrupted focus time, autonomy, competence, and connectedness to others to engender self-actualization and intrinsic motivation. [5]

Contrast this to the conditions in a typical work environment where work occurs in functional silos in a batch-queue-handoff system. These frequent interruptions (emergency requests, ad hoc status reports, constant communication alerts, and so on) are the norm, and excessive WIP drives frequent task switching. (Figure 8).



© Scaled Agile, Inc.

Figure 8. Time' in the zone' is often a fraction of the total workday

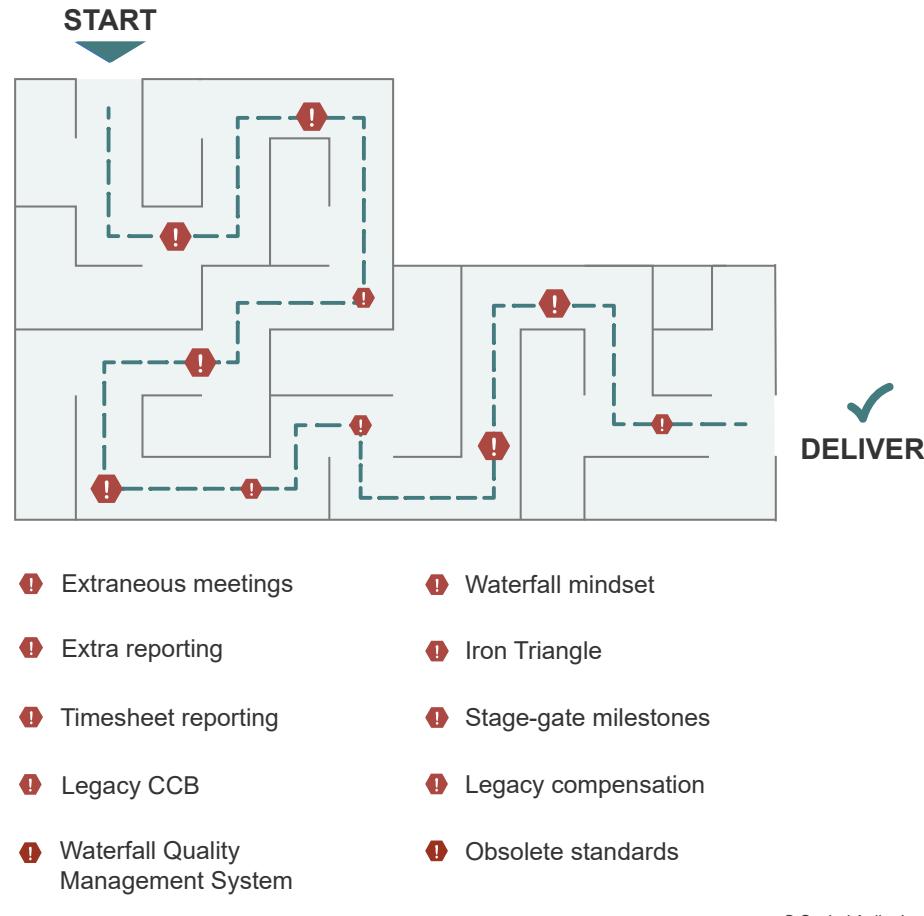
There is an essential connection between creating a continuous flow of value and creating a working environment where individuals and teams can maximize their time in the zone. Knowledge workers also need the time and space free from interruption essential for complex tasks involving application, analysis, evaluation, and creativity, and ultimately the personal satisfaction that completion engenders.

## #8 Remediate Legacy Policies and Practices

During or after a Lean-Agile transformation, enterprises must constantly look out for legacy policies and practices that inhibit flow (Figure 9). Many of these practices became part of the culture and are described as “we’ve always done it this way,” even when they are no longer fit for purpose. Examples are many, some legendary:

- Continued reliance on phase-gate milestones and the iron triangle of fixed scope, resources, and time. In reality, all three components become fixed instead of tradeoffs among constraints.
- Obsolete or unnecessary change control boards, including extraneous oversight and reporting
- Waterfall-based quality management systems for regulations and compliance
- Obsolete tech standards—design specifications, audit practices, and the like—in environments where they are not mandated or required for quality

- Continuation of timesheet reporting *in addition to* Agile Lifecycle Management (ALM) tooling, requiring double recording of time
- Traditional HR performance reviews and compensation policies that cause unhealthy competition
- Agile is adopted only by teams; the mindset of management and portfolio governance remain unchanged



**Figure 9. The maze of legacy policies and practices**

And that's not an exhaustive list. While many of these patterns may well have solved problems in the past, they now create *new* problems that become ongoing impediments to flow. They must be proactively or reactively discovered, eliminated, modified, or mitigated.

## Measuring Flow

It's difficult to improve what isn't measured. And historically, it has been challenging to measure the development process as the work items are mostly intangible and invisible, living mainly in the minds of the knowledge workers who design and build the systems. But the physics and tooling of flow (timeboxes, Kanban, value stream mapping, CD pipeline, stories, features, and more) provide

a new basis for measuring the flow of value through the development value stream. The SAFe [Measure and Grow](#) article describes six metrics —flow *distribution*, *velocity*, *time*, *load*, *efficiency*, and *predictability*—for measuring flow (see Figure 10).

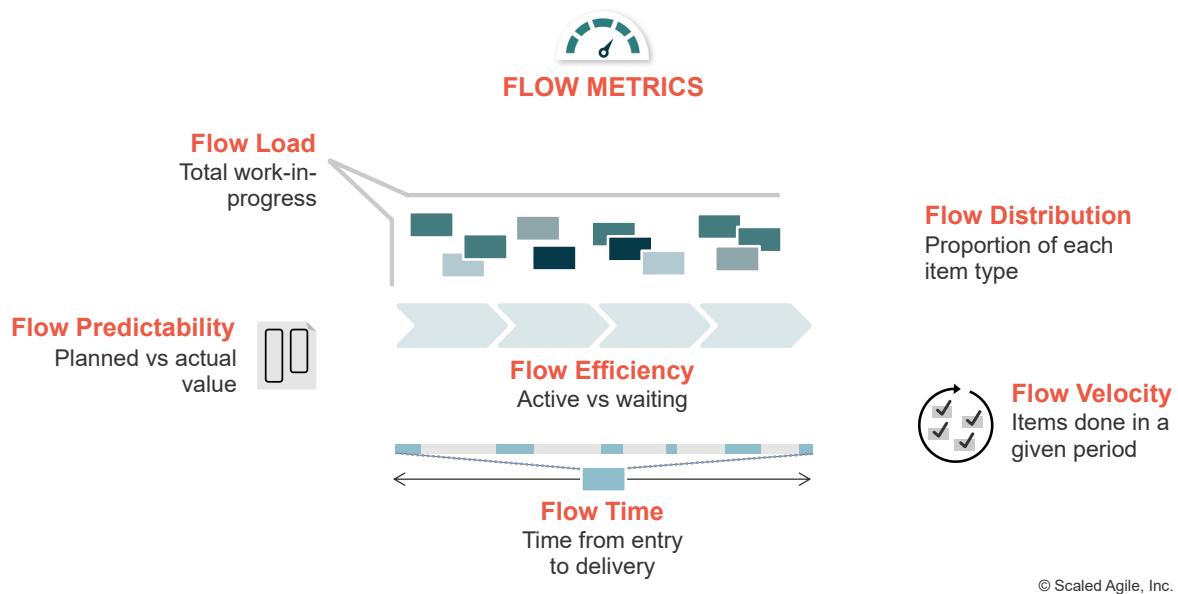


Figure 10. SAFe flow metrics

- **Flow Distribution** is a measure of the proportion of work items by type in a system.
- **Flow Velocity** measures the number of completed work items over a time period.
- **Flow Time** is a measure of the time elapsed from start to completion for a given work item.
- **Flow Load** is a measure of the number of work items currently in progress (active or waiting).
- **Flow Efficiency** is the ratio of the total time spent in value-added work activities divided by the flow time.
- **Flow Predictability** is a measure of how consistently teams, ARTs, and portfolios are able to meet their commitments.

Together, these metrics provide a comprehensive view as new value flows through the development value stream. These measures need to be relatively easy to collect, maintain, and visible to be valuable and actionable. Fortunately, they can be automated using most modern Agile Lifecycle Management (ALM) tooling. In addition, organizations should complement the flow metrics with qualitative data to ensure their delivery flow creates the right solutions for customers when needed.

## Summary

These eight flow accelerators help teams increase throughput and deliver value faster. As an added benefit, implementing them gives people a sense of control over the process and triggers

fast and measurable improvements in customer satisfaction and employee engagement.

---

## Learn More

- [1] Womack, James P., and Jones, Daniel T. *Lean Thinking: Banish Waste and Create Wealth in Your Organization*. Free Press, 2003.
- [2] Goldratt, Eliyahu M. *The Goal: A Process of Ongoing Improvement*. The North River Press Publishing Corporation, 1986
- [3] Goldratt, E. M. *What is this Thing called Theory of Constraints and How should it be Implemented?* North River Press, Inc, 1990
- [4] Oosterwal, Dantar P. *The Lean Machine*. AMACOM, 2010
- [5] Reinertsen, Donald G. *The Principles of Product Development Flow: Second Generation Lean Product Development*. Celeritas, 2009.
- [6] Ward, Allen, and Durward Sobeck. *Lean Product and Process Development*. Lean Enterprise Institute, 2014.
- [7] Csikszentmihalyi, Mihaly. *Flow*. HarperCollins, 1990
- [8] Kersten, Mik. *Project to Product: How to Survive and Thrive in the Age of Digital Disruption with the Flow Framework*. IT Revolution Press, 2018.

Last update: 6 February 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

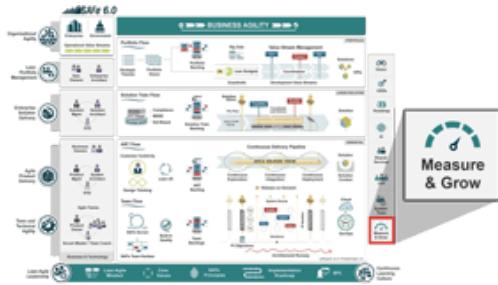
[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



## + Framework



“ The great thing about fact-based decisions is that they overrule the hierarchy.

—Jeff Bezos, founder of Amazon [1]

## Measure and Grow

Measure and Grow is an approach SAFe enterprises use to evaluate progress towards Business Agility and determine improvement actions.

Business Agility sets new performance standards for organizations, requiring fast, effective responses to emerging business opportunities. However, to improve speed and agility, leaders, teams, and business stakeholders need a way to reliably measure the current state and identify what they can do to improve. Therefore, choosing what and how to measure is a critical enabler of continuously improving business performance. This article describes a comprehensive approach that can be used to measure the performance of a SAFe portfolio or any of its elements.

## Details

When it comes to metrics, the first and most important thing is understanding *what* to measure. The goal of *Business Agility* is clear: quickly respond to market changes and emerging opportunities with innovative, digitally-enabled business solutions. The *Business Agility Value Stream*, shown in Figure 1, visualizes the steps needed to achieve this. SAFe's three measurement domains, *Outcomes*, *Flow*, and *Competency*, support this process and provide a comprehensive yet simple model for measuring progress toward this goal. The insights provided by these three

measurement domains support better decision-making and help to identify opportunities for improvement.

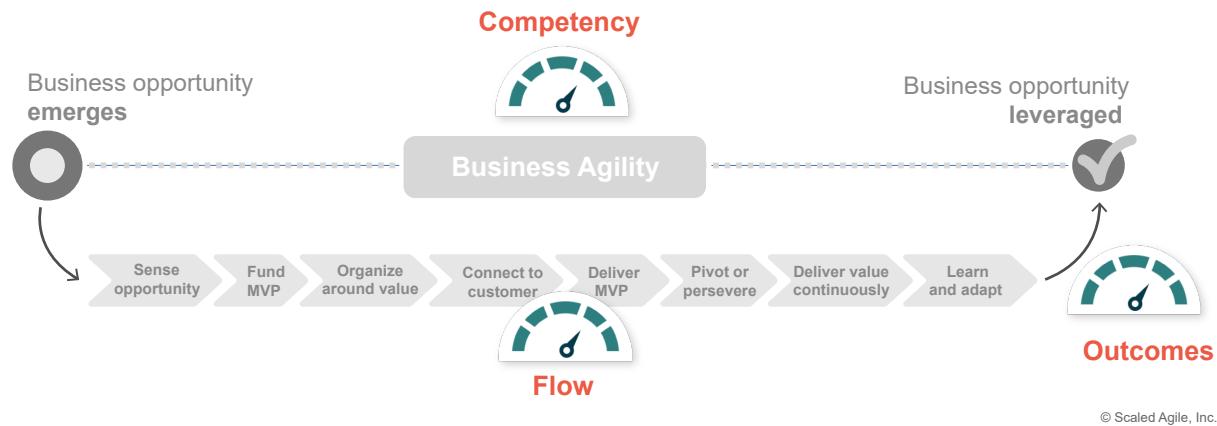


Figure 1. Three SAFe measurement domains support the goal of business agility

The three measurement domains are defined as follows:

- **Outcomes:** Do our solutions meet the needs of our customers and the business?
- **Flow:** How efficient is the organization at delivering value to the customer?
- **Competency:** How proficient is the organization in the practices that enable business agility?

Furthermore, these three measurement domains are applicable at every level of an organization. As Figure 2 illustrates, they can be used to measure performance within a SAFe portfolio, a Solution Train, an Agile Release Train, or even a single Agile Team.

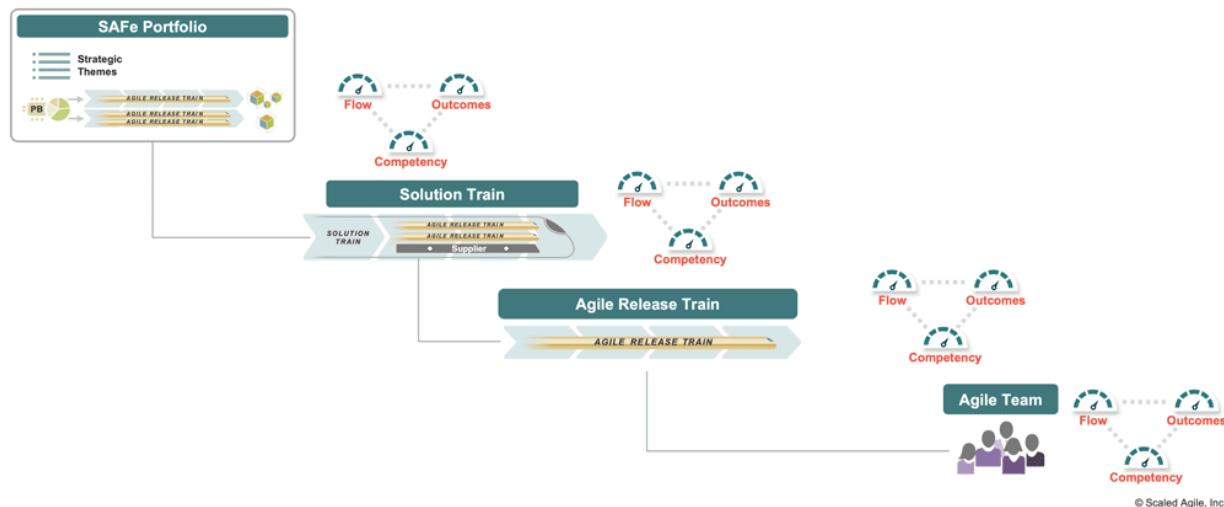


Figure 2. The three measurement domains are applicable at all levels of a SAFe Enterprise

Each measurement domain contains a set of specific metrics, which are described in the sections below.

# Measuring Outcomes

Outcomes help determine whether a development organization's efforts produce the desired business benefit. Outcomes may measure externally facing concerns such as increases in revenue, customer retention, etc., as well as internal considerations such as employee engagement.

## KPIs and OKRs

A SAFe Portfolio measures outcomes using *Key Performance Indicators (KPIs)* and *Strategic Themes*. Each KPI is a specific and quantifiable measure of business results for the value streams within that portfolio. Outcome metrics of this kind are typically context-specific and depend heavily on the organization, business model, and the nature of solutions delivered to the customer. For example, the customer conversion rate may be a meaningful metric for an eCommerce business but would be inapplicable to a microchip manufacturer. Some indicators, however, may be successfully applied across contexts, such as Net Promoter Score, for example.

The [Value Stream KPIs](#) article provides guidance for defining appropriate KPIs for that particular SAFe Portfolio. Examples of KPIs appear in Figure 3.

Operational Value Stream Type	Example KPIs
<b>Software product</b> (Consumer facing web site example)	AARRR ('Pirate Metrics'): Acquisition, Activation, Revenue, Retention, Referrals
<b>Fulfillment</b> (Consumer loan example)	Conversion funnel analytics, average time to decision, automated approval rate, Net Promoter Score, default rate, customer lifetime value
<b>Supporting</b> (Customer support example)	Tickets outstanding, Net Promoter Score, first response time, mean time to resolution, cost per ticket, customer experience score
<b>Manufacturing</b> (USB streaming microphone example)	Units sold, cost of goods sold, Supplier health, throughput, cycle time, inventory turns, cash-to-cash cycle time

© Scaled Agile, Inc.

**Figure 3. Value stream KPIs are specific to the type of operational value stream supported**

Of course, these KPIs are partly informed by the portfolio strategic themes since the strategy helps to determine the targets to be met. However, whereas KPIs represent ongoing 'health' metrics that can be used to measure overall business performance, the strategic themes, formulated as OKRs, define the specific outcomes that the portfolio is working towards to achieve future success. Therefore the key results associated with these objectives determine another set of critical outcome metrics that are typically measured quarterly, as shown in Figure 4 below.

Objective	Key Results	Q1	Q2	Q3	Q4
Achieve a dominant position within the autonomous delivery market	Increase serviceable market to 75% within 18 months	45%	↑ 55%	↓ 47%	↑ 52%
	Increase Net Promoter score from 35 to 60	35	↑ 49	↑ 54	↑ 57
	Improve repeat business rates from 60% to 80%	60%	↑ 64%	↑ 67%	↑ 72%
	Acquire 15% new customers over the next 12 months	2%	↑ 7%	↓ 4%	↑ 10%

© Scaled Agile, Inc.

Figure 4. Measuring progress of strategic themes using OKR metrics

Within a large portfolio, it can be useful to create specific OKRs for each value stream that align with the portfolio strategic themes. And further, for large value streams that contain multiple ARTs, this process can be repeated to create a set of OKRs that define the goals for each specific ART. This approach also allows those at each level of the organization to see the direct impact of their work against the key results of the OKRs they are aligning to. (This use case, and others, are described in more detail in the [OKRs article](#))

## Employee Engagement

Another important internal outcome metric is employee engagement. Employee engagement measures the amount to which individuals feel motivated and actively engaged in supporting the achievement of the organization's goals and values. Higher levels of employee engagement result in higher productivity, efficiency, and innovation levels. Consequently, lower levels of employee engagement can lead to poor motivation, lower-quality work, and higher staff turnover.

Different methods exist for measuring employee engagement, and each organization needs to determine what is right for them. Some organizations will use an annual employee engagement survey. Others use an employee Net Promoter Score (eNPS), which asks, 'How likely are you to recommend your employer to others as a place of work?' and is measured on a 10-point scale. Whichever approach is chosen, the resultant data should inform initiatives to improve employee engagement levels.

## Iteration Goals and PI Objectives

Localized metrics such as [Iteration Goals](#) and [PI Objectives](#) are used effectively by teams and trains to measure whether they are achieving their outcomes. These ensure their efforts are focused on the needs of the customer and the business, provide feedback on the progress they are making toward business results, inform the prioritization process, and facilitate acceptance of work.

Establishing effective outcome metrics requires a close collaboration of trains, value streams, and portfolios with their business partners, who can best define the business benefits resulting from solution investment.

# Measuring Flow

Flow measures are used to determine how effective an organization is at delivering value. The Flow Framework created by Mik Kersten [2] provides five metrics that can be used to measure different aspects of flow. As SAFe is a flow-based system, each metric is directly applicable. In addition, SAFe defines *Flow Predictability* to measure how Teams, ARTs, and Solution Trains deliver business value against their planned objectives. These six flow metrics are shown in Figure 5 and described further below.

Metric	Description
<b>Flow Distribution</b>	Proportion of work items by type in a system
<b>Flow Velocity</b>	Number of completed work items over a time period
<b>Flow Time</b>	Time elapsed from start to completion for a given work item
<b>Flow Load</b>	Number of work items currently in progress (active or waiting)
<b>Flow Efficiency</b>	Ratio of the total time spent in value-added work activities divided by the total flow time
<b>Flow Predictability</b>	How consistently teams, ARTs, and portfolios are able to meet their commitments

© Scaled Agile, Inc.

Figure 5. The six SAFe flow metrics

## Flow Distribution

**What does it measure?** Flow distribution measures the amount of each type of work in the system over time. This could include the balance of new business [Features](#) (or [Stories](#), [Capabilities](#), or [Epics](#)) relative to [Enabler](#) work, as well as the work to resolve defects and mitigate risks. Alternatively, a helpful view of portfolio flow distribution might illustrate the distribution of funding allocation across investment horizons.

**How is this measured?** One simple comparison is to count the number of each type of work item at any point in time. A more accurate measure might consider the size of each work item. Agile Teams may measure flow distribution per iteration, but PI boundaries are commonly used to calculate this at the ART level and above, as shown in Figure 6.

**Why is this important?** To balance both current and future velocity, it is important to be able to track the amount of work of each type that is moving through the system. Too much focus on new business features will leave little capacity for architecture/infrastructure work that addresses various forms of technical debt and enables future value. Alternatively, too much investment in

technical debt could leave insufficient capacity for delivering new and current value to the customers. Target capacity allocations for each work type can then be determined to help balance these concerns. Returning to the portfolio example, tracking the distribution of funding across investment horizons provides a means to ensure a balanced portfolio that ensures both near- and long-term health.

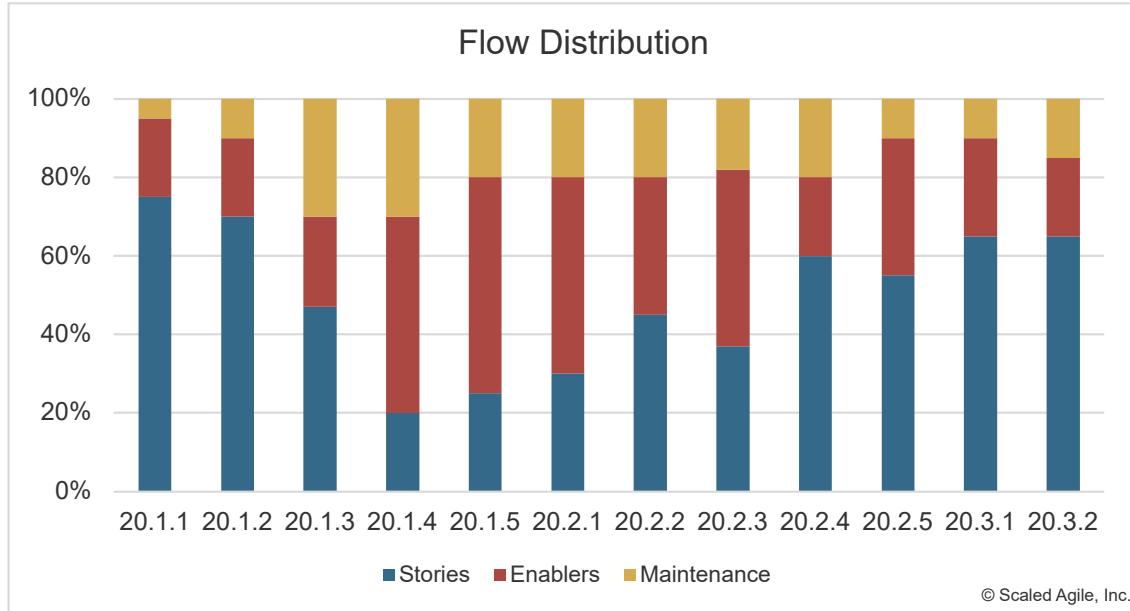


Figure 6. Flow distribution over time

## Flow Velocity

**What does it measure?** Flow velocity measures the number of backlog items (stories, features, capabilities, epics) completed in a given timeframe; this is also known as the system's throughput. (Figure 7).

**How is this measured?** As with flow distribution, the simplest measure of velocity is to count the number of work items completed over a time period such as an iteration or PI. Those items can be stories, features, capabilities, or even epics. However, since work items are not all the same size, a more common measure is the total number of completed story points for work items of a type over the timeframe.

**Why is this important?** All other things being equal, higher velocity implies a higher output and is a good indicator that process improvements are being applied to identify and remove delays from the system. However, the system's velocity will not increase forever, and over time stability of the system is important. Significant drops in velocity highlight problems that warrant investigation.

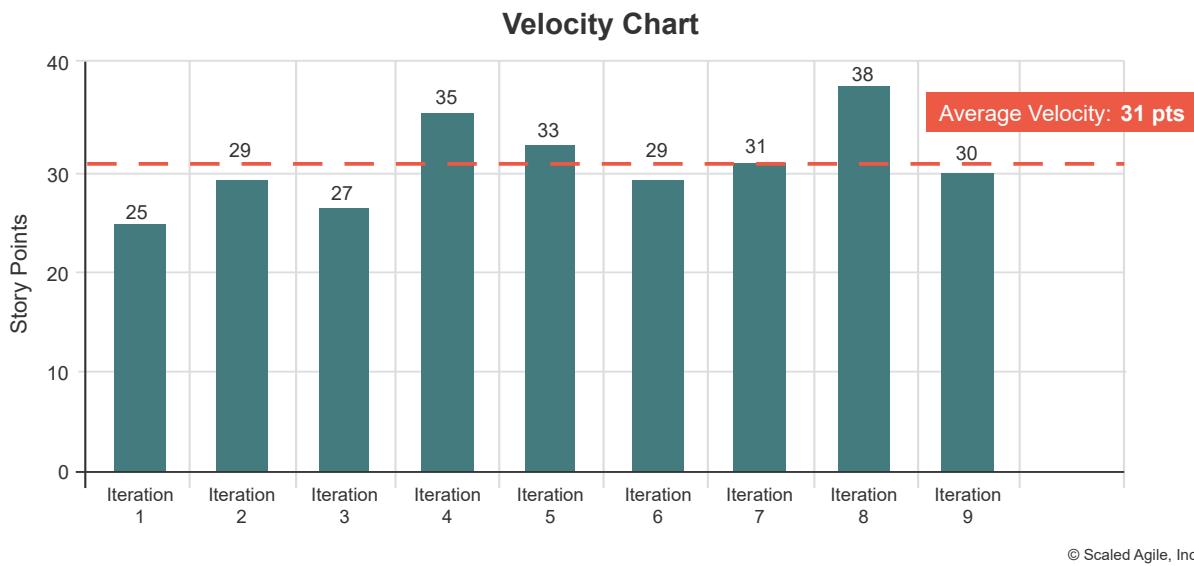


Figure 7. An example of an Agile team's flow velocity in story points per iteration

## Flow Time

**What does it measure?** Flow time measures the total time elapsed for all the steps in a workflow and is, therefore, a measure of the efficiency of the entire system. Flow Time is typically measured from ideation to production. Still, it can also be useful to measure Flow Time for specific parts of a workflow, such as code commit to deployment, to identify opportunities for improvement.

**How is this measured?** Flow time is typically measured by the average length of time it takes to complete a particular type of work item (stories, features, capabilities, epics). A histogram is a useful visualization of flow time (Figure 8) since it helps identify outliers that may need attention and supports the goal of reducing the overall average flow time.

**Why is this important?** Flow time ensures that organizations and teams focus on what is essential – delivering value to the business and customer in the shortest possible time. The shorter the flow time, the less time our customers spend waiting for new features and the lower the cost of delay incurred by the organization.

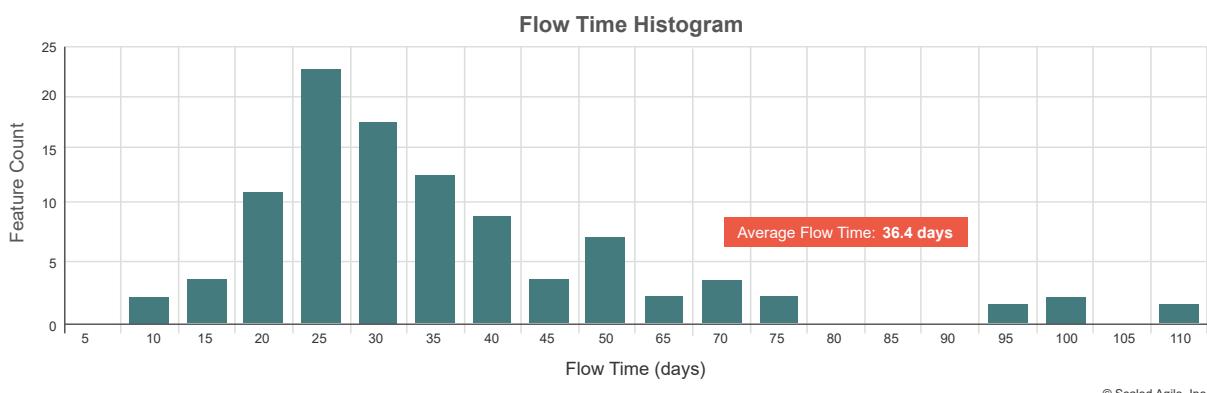


Figure 8. Measuring feature flow time with a histogram

## Flow Load

**What does it measure?** Flow load indicates how many items are currently in the system. Keeping a healthy, limited number of active items (limiting work in process) is critical to enabling a fast flow of items through the system ([SAFe Principle #6](#)).

**How is it measured?** A Cumulative Flow Diagram (CFD) is one common tool used to effectively visualize flow load over time (Figure 9). The CFD shows the quantity of work in a given state, the rate at which items are accepted into the work queue (arrival curve), and the rate at which they are completed (departure curve). At a given point in time, the flow load is the vertical distance between the curves.

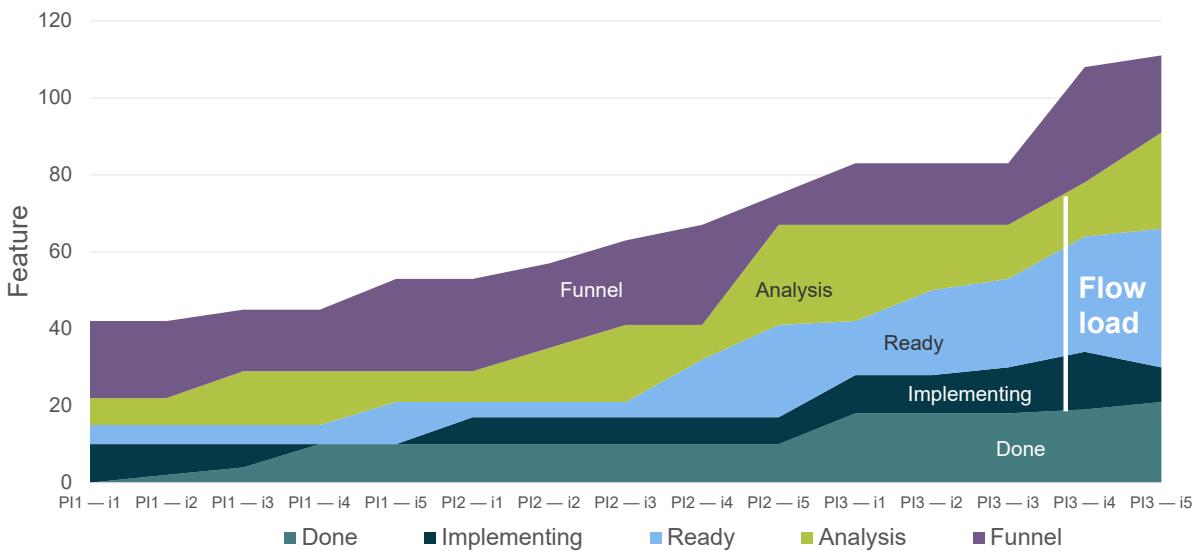


Figure 9. Visualizing flow load with a cumulative flow diagram.

**Why is this important?** Increasing flow load is often a leading indicator of excess work in process. All other things being equal, the likely result will be an increase in future flow times as queues start to build up in the system. For this reason, measuring and reducing flow load is of critical importance. Furthermore, it is easy to see how more frequent delivery lowers flow load while improving flow time and flow velocity.

## Flow Efficiency

**What does it measure?** Flow efficiency measures how much of the overall flow time is spent in value-added work activities vs. waiting between steps.

**How is it measured?** To correctly measure flow efficiency, the teams, trains, and value streams must clearly understand what the flow is in their case and what steps it passes through. This understanding is achieved with the help of Value Stream Mapping – a process of identifying workflow steps and delays in a system. (For more on Value Stream Mapping, see the [Continuous Delivery Pipeline](#) article and [3]. In addition, Scaled Agile's Value Stream Mapping Workshop and [SAFe DevOps course](#) provide comprehensive guidance on performing Value Stream Mapping.)

Once the steps have been mapped, flow efficiency is calculated by dividing the total active time by the flow time and is expressed as a percentage, as shown in Figure 10.

**Why is this important?** In a typical system that has not yet been optimized, flow efficiency can be extremely low, often in single digits. Low flow efficiency indicates excessive bottlenecks and delays in the system. Conversely, the higher the flow efficiency, the better the system can deliver value quickly.

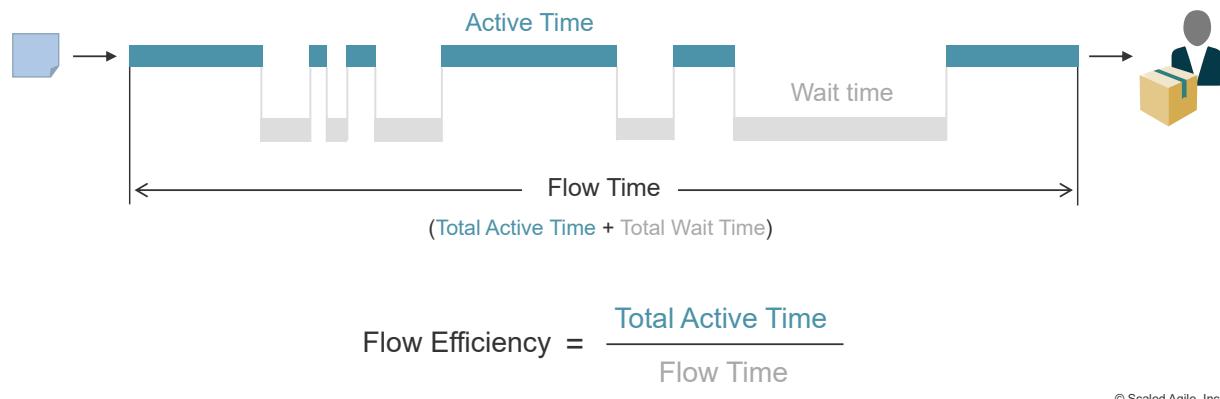


Figure 10. Flow efficiency is the ratio of total active time to flow time

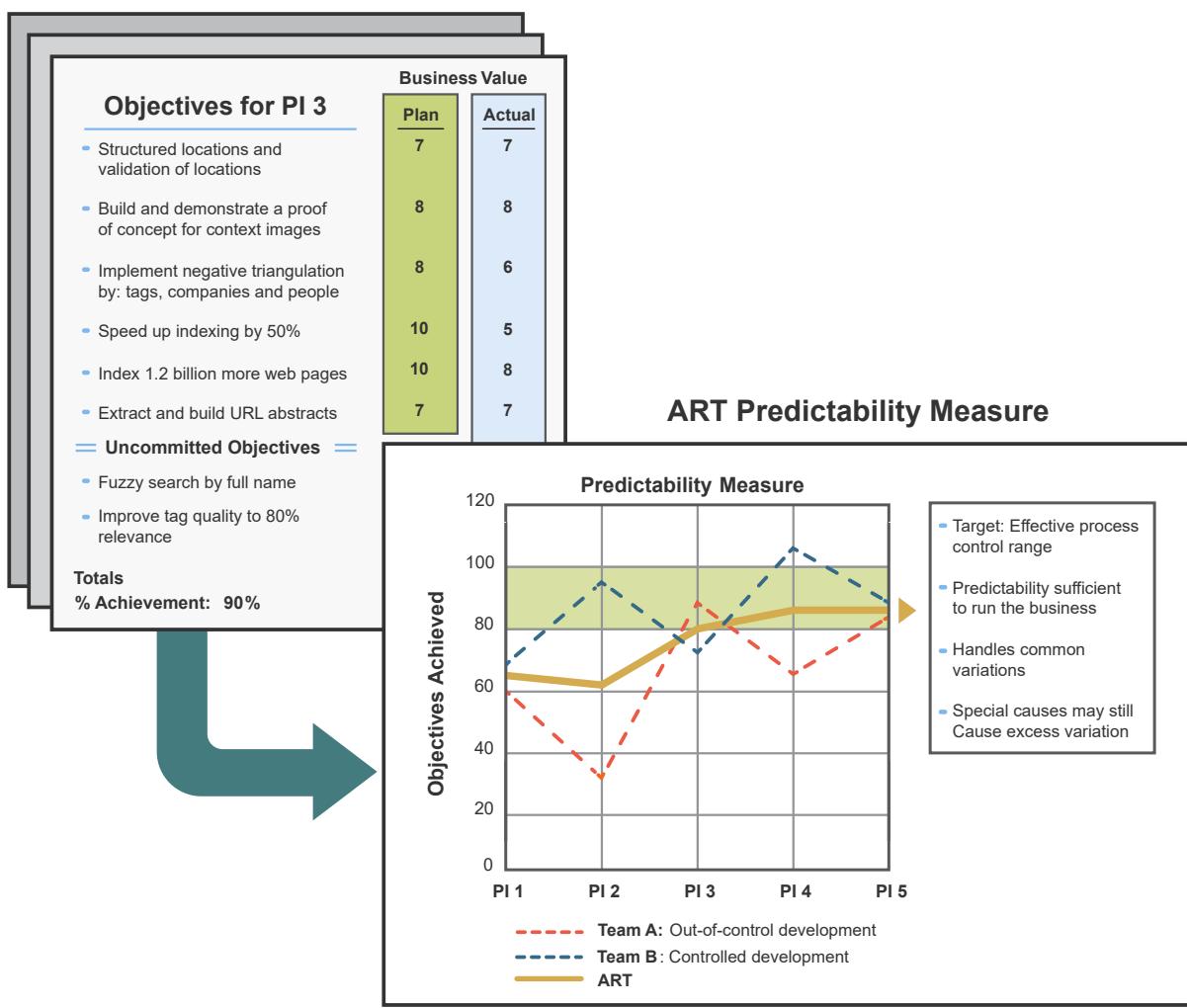
## Flow Predictability

**What does it measure?** Flow predictability measures how well teams, ARTs, and Solution Trains can plan and meet their PI objectives.

**How is it measured?** Flow Predictability is measured via the ART Predictability Measure, Figure 11. This measure calculates the ratio of actual business value delivered in a PI to the planned business value. For more information on calculating this important metric, see the [Inspect and Adapt](#) article.

**Why is this important?** Low or erratic predictability makes delivery commitments unrealistic and often highlights underlying problems in technology, planning, or organization performance that need addressing. Reliable trains should operate in the 80 – 100 percent range; this allows the business and its stakeholders to plan effectively.

## Team PI Performance Reports



© Scaled Agile, Inc.

Figure 11. ART predictability measure



**Note on DORA Metrics:** Within and across the three measurement domains, it can often be helpful to bring together complementary metrics to provide a specific view of performance. An example is the DORA metrics used to measure the performance of an organization's DevOps capabilities [4]. The four DORA metrics are 1)

deployment frequency, 2) lead time for changes, 3) time to restore service, and 4) change failure rate.

Each of these is an application of a flow metric designed for a particular use case. *Deployment frequency* is a productivity metric and an example of flow velocity. Instead of stories completed per iteration, it measures the number of deployments per given time period. Both *lead time for changes* and *time to restore service* are examples of flow time metrics, focusing on specific steps in the workflow. Finally, *change failure rate* represents the percentage of changes that require remediation after they have gone to production. In other words, how often does the work that arrives at the 'deploy to production' step in the workflow contain errors? When creating a

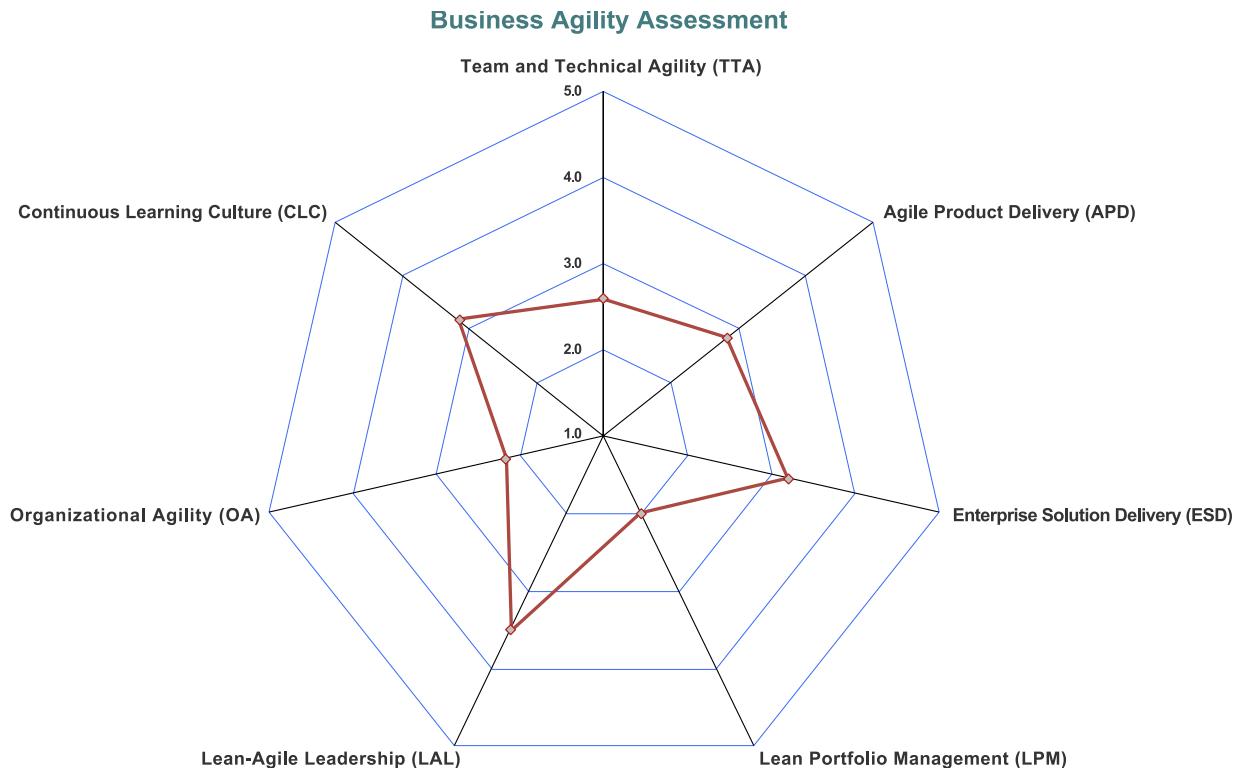
value stream map to measure flow efficiency, this is captured in the percent complete and accurate (%C&A) metric for each step i.e. the percentage of work that the next step can process without needing rework. High rates of change failure contribute significantly to low flow efficiency.

---

## Measuring Competency

Achieving business agility requires a significant degree of expertise across the [Seven SAFe Core Competencies](#). While each competency can deliver value independently, they are also interdependent in that true business agility can be present only when the enterprise achieves a meaningful state of mastery of all.

Measuring the level of organizational competency is accomplished via two separate assessment mechanisms designed for significantly different audiences and different purposes. The SAFe Business Agility Assessment is designed for the business and portfolio stakeholders to assess their overall progress on the ultimate goal of true business agility, as shown in Figure 12.



**Figure 12. A completed business agility assessment.**

The spreadsheet version of the assessment can be downloaded [here](#).

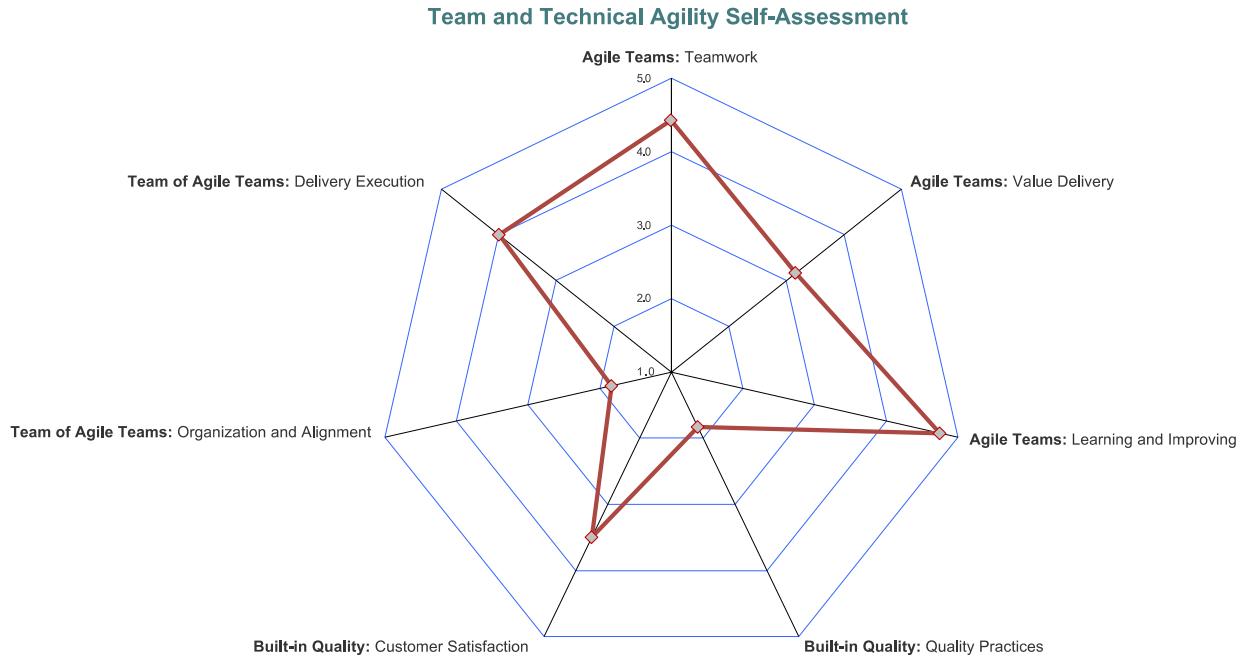
[Download the SAFe Business Agility Assessment](#) ▾

---



**Note for SAFe Studio Members:** All the SAFe assessments are available for SAFe Studio Members online through our partner Comparative Agility. This provides additional data collection, analysis, comparison, and trending capabilities that can be used to improve performance. Access these from the [Measure and Grow SAFe Studio page](#).

The SAFe Core Competency Assessments help teams and trains improve on the technical and business practices they need to help the portfolio achieve that larger goal. There is one for each of the seven core competencies. The Team and Technical Agility Assessment as an example, in Figure 13.



**Figure 13. A report from a team and technical agility competency assessment**

Each assessment follows a standard process pattern of running the assessment, analyzing the results, taking action, and celebrating successes. In addition, comparative analysis against the competition is achievable via online assessment tools available to SAFe community members. Additional information and guidance can be found in the Advanced Topic Article [Successfully Facilitating SAFe Assessments](#).

The following table provides download links for each of the core competency assessments.

#### Core competency assessments download

##### [Organizational Agility](#)

## [Core competency assessments download](#)

[Lean Portfolio Management](#)

[Enterprise Solution Delivery](#)

[Agile Product Delivery](#)

[Team and Technical Agility](#)

[Continuous Learning Culture](#)

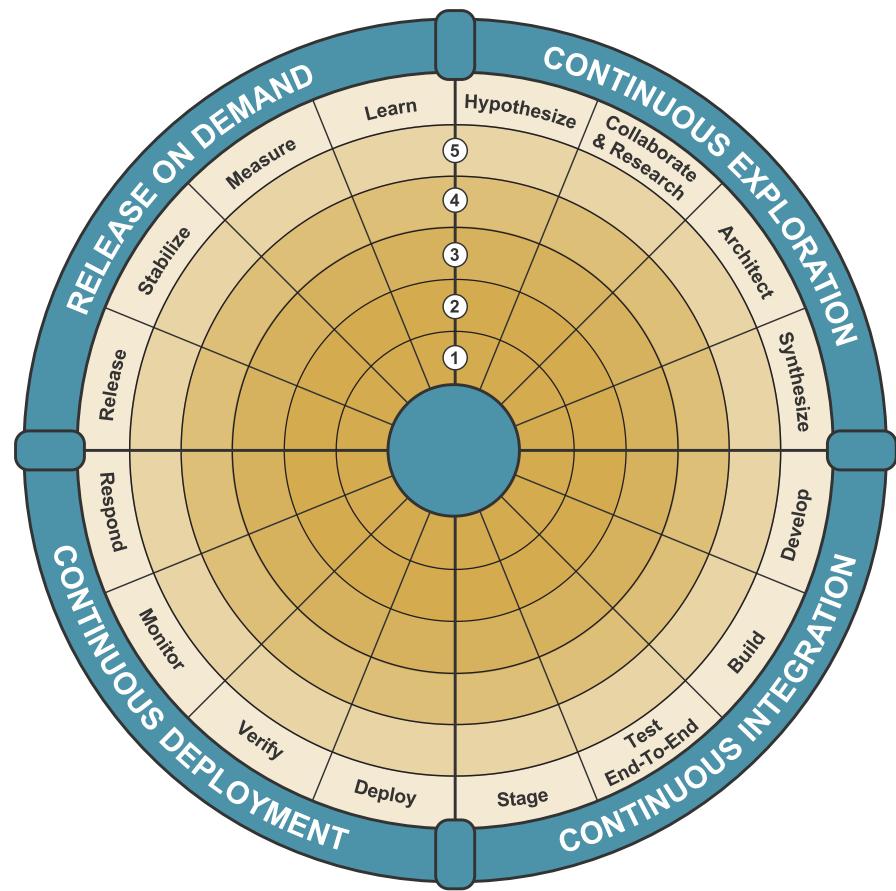
[Lean-Agile Leadership](#)

---

## Measuring and Managing DevOps Maturity

In addition to the business agility and core competency assessments, the SAFe DevOps Health Radar (Figure 14) is an assessment that helps ARTs and Solution Trains optimize their value stream performance. It provides a holistic DevOps health check by assessing the maturity of the four aspects and 16 activities of the continuous delivery pipeline. The Health Radar is used to measure baseline maturity at any point in a DevOps transformation and guide fast, incremental progress thereafter.

**Note:** The DevOps Health Radar should be used alongside the Agile Product Delivery assessment to ensure full coverage of all three dimensions of the APD core competency.



© Scaled Agile, Inc.

Figure 14. The SAFe DevOps Health Radar

The spreadsheet version of the DevOps Health Radar assessment can be downloaded [here](#).

## Four Critical Success Factors for Effective Measurement

Measuring organizational performance is one of the most sensitive areas in every business, often subject to politics and various dysfunctions. Additionally, since measurement inevitably involves the interpretation of data, it is subjected to cognitive bias, communication issues, and alignment disconnects. All this leads to a substantial danger in any measurement system: if not correctly implemented, *some measurements can do more harm than good*. The following success factors will help guide the enterprise to more effective measurements and more importantly, better business results.

### 1. Use measurement in conjunction with other discovery tools

However well-designed, any measurement system provides only a partial picture of reality, and adding more metrics does not necessarily improve visibility. There is a story behind every number, and that story often contains more important information than the number itself can convey. A powerful tool to be used in conjunction with measurement is *direct observation* (Gemba) – observation of the actual environment where value is created and where it meets the customer. Formal measures and informal observations reinforce one another. But used in isolation, ‘managing by just the numbers’ can lead to poor outcomes and even worse morale.

## 2. Apply metrics where they support improved decision-making

A common trap when applying metrics is over-measuring for fear of not measuring enough. Although many metrics can be automated, as the number of metrics and frequency of measurement increases, so will the effort needed to collect and analyze the data. When considering whether to include an additional metric in your measurement system it can be prudent to ask the question, 'what decisions will this metric help inform that isn't supported today with our existing metrics?' If the new metric helps to drive better decision-making, it should be a candidate for inclusion; if not, then omit it. A further clarifying question is 'do we need to measure this right now?' This question recognizes that the metrics we use will change over time as the decisions we need to make change throughout the development process.

## 3. Understand the effect of metrics on behaviors

In a positive culture, knowledge workers are motivated to deliver winning solutions and work with purpose, mastery, and autonomy. However, when too much emphasis is placed on a specific numerical indicator, and when that indicator is directly tied to compensation or career growth opportunities, achieving that number becomes the goal instead of creating effective solutions.

Additionally, the pressures to succeed often lead to the misuse of metrics. For example, flow efficiency may be used to assign blame for a missed delivery date to a particular ART that has become a bottleneck rather than using this information to identify systemic problems that need addressing. Perhaps the root cause was a lack of resources or changing priorities outside the ART's control.

In each case, [SAFe's Core Values](#) of transparency and alignment must provide the proper foundation for an effective measurement system alongside creating an environment where *the facts are always friendly*.

## 4. Interpret metrics carefully

Just collecting specific measures is not enough. If interpreted without proper understanding, an indicator may be quite misleading. For example, when measuring flow time, the work items must be *actual, valuable* features (stories, and so on) that carry business benefits; otherwise, the train may be reporting improvements in the flow of work but struggling to get any real value out of the door.

---

## Learn More

[1] Hunt, Helena, ed. *First Mover: Jeff Bezos In His Own Words*. Agate Publishing, 2018.

- [2] Kersten, Mik. *Project to Product: How to Survive and Thrive in the Age of Digital Disruption with the Flow Framework*. IT Revolution Press, 2018.
- [3] Martin, Karen, and Mike Osterling. *Value Stream Mapping: How to Visualize Work and Align Leadership for Organizational Transformation*. McGraw-Hill Education, 2018.
- [4] Accelerate. *2023 State of DevOps Report*. Google. Retrieved October 10, 2023, from <https://cloud.google.com/devops/state-of-devops>

Last update: 10 October 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

**Scaled Agile, Inc**

## Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

## Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



## + Framework



“ Few are those who see with their own eyes and feel with their own hearts.

—Albert Einstein

# Iteration Review



**Note:** For more on SAFe Scrum, please read the additional Framework articles in the Scrum series, including [SAFe Scrum](#), [Scrum Master/Team Coach](#), [Iterations](#), [Iteration Planning](#), [Iteration Goals](#), and [Iteration Retrospective](#)

The Iteration Review is a regular SAFe Scrum event where the team inspects the iteration increment, assesses progress, and adjusts the team backlog.

## Details

The iteration review is the second to last event of the iteration. It provides a way to regularly gather immediate, contextual feedback from the team and its stakeholders. The iteration review offers several benefits:

- It brings closure to the iteration timebox
- It allows team members to demonstrate their contributions and to take some satisfaction and pride in their work
- It provides an opportunity for the team to receive feedback to improve the solution under development
- It shows the results of the latest system increment to help determine future work

An iteration review is where the team demos a working, tested increment. No slides are needed. Instead, the focus is on the solution instead of a presentation. The team and stakeholders review the accomplishments in the iteration—based on this information, attendees collaborate on what to do next. The [Team Backlog](#) may also be adjusted to meet new opportunities.

## Inputs and Outputs of the Iteration Review

*Inputs* to the iteration review include:

- Iteration goals and [PI Objectives](#)
- The team's increment deployed to a staging environment (or production environment where appropriate)
- A brief list of work to be demoed to prepare people for what they are about to see

A successful iteration review event delivers the following *outputs*:

- Feedback on the increment and progress toward the iteration goals and broader PI Objectives
- Adjusted Team Backlog based on feedback
- Identification of risks and impediments

## Preparation

The preparation for the iteration review begins during Iteration Planning, where teams start thinking about how they will demo the committed [Stories](#). ‘Beginning with the end in mind’ facilitates iteration planning and alignment, fostering a more thorough understanding of the functionality needed ahead of iteration execution.

## Process

The PO starts the iteration review by discussing the iteration goals and their status. It proceeds with a walk-through of all the committed stories. Teams demonstrate the significant new behavior and knowledge gained from the iteration’s completed stories, [Spikes](#), [Refactors](#), and [Nonfunctional Requirements \(NFRs\)](#). The demos should be part of a working, tested system—preferably in a staging environment closely resembling production. Spikes and NFRs can be demonstrated via a presentation of findings if the functionality lacks a user interface. Stakeholders provide feedback on the stories that the team demoed, which is the primary goal of the review.

The team reflects on stories *not* completed after the demo and why they could not finish them. This discussion usually results in discovering impediments or risks, false assumptions, changing priorities, estimating inaccuracies, over-commitment, or other problems with [Team Flow](#). These findings often lead to further study in the [Iteration Retrospective](#) and may result in improvements

to support better planning and execution going forward. Figure 1 shows an iteration review in action.

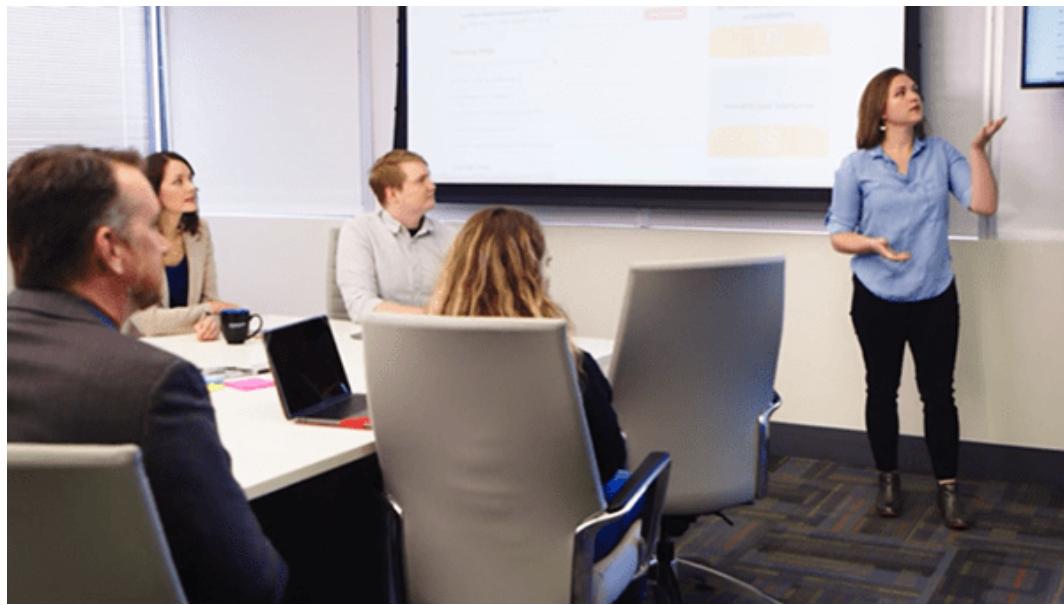


Figure 1. An Agile team demoing a working, tested increment

The team reflects on how well it did within the iteration and determines its progress toward its Team PI objectives. It finishes the event by refining the Team Backlog, based on the feedback received, before the next iteration planning event.

## Attendees

Attendees at the iteration review include:

- The [Product Owner \(PO\)](#)
- [Scrum Master/Team Coach](#)
- All team members and other stakeholders or subject matter experts
- Stakeholders, which may also include other teams or trains.

Although [Agile Release Train \(ART\)](#) stakeholders may attend, their interests and the level of detail they require are usually better aligned with the [System Demo](#).

## Agenda

The timebox for the event is a maximum of 90 minutes for a two-week iteration. Figure 2 shows an example agenda and a description of each item.



Figure 2. An example iteration review agenda

The Scrum Master/Team Coach or PO typically facilitates the iteration review for the team, ensuring they stay within the agreed event agenda and timebox. Following are descriptions of the example agenda:

1. **Review iteration goals** – Discuss the status of each iteration goal. Teams may also review PI objectives for a broader context.
2. **Demonstrate completed stories** – The iteration review proceeds with a walk-through and demonstration of each completed story (spikes, NFRs, and any other work finished by the team). Demos should show progress towards iteration goals, PI objectives, solution changes, test scenarios, or a prototype representing the user's environment. Spikes can be demoed as a presentation of findings or learning. The team and stakeholders should ask questions and provide constructive feedback.
3. **Reflect on any incomplete stories** – Next, the team should reflect on missed iteration goals and stories they did not complete to identify opportunities for future improvement. This discussion usually results in discovering impediments or risks, false assumptions, changing priorities, estimating inaccuracies, or over-commitment.
4. **Identify risks and impediments** – After the demo and reflecting on any incomplete stories, the team identifies new risks or dependencies that might impact achieving the PI objectives. Teams often use the ROAM (Resolved, Owned, Accepted, Mitigated) process to address the risks as needed.
5. **Refine the Team Backlog** – Based on stakeholder feedback, the team can refine their backlog to reflect any adjustments before the next [Iteration Planning](#) event.

## Guidelines

Below are some tips for running a successful iteration review event:

- Limit preparation to less than two hours
- Timebox the event to about 90 minutes
- Minimize the use of slides; the iteration review is intended to garner feedback on working, tested system components
- Verify that completed stories meet the definition of done (DoD)
- Demonstrate incomplete stories if enough functionality is available to get feedback
- Encourage providing constructive feedback and celebration of the team's accomplishments
- If a significant stakeholder cannot attend, the PO should follow up to report progress and get feedback

Note: Teams applying [Continuous Delivery](#) will generally review completed Stories or [Features](#) as soon as they are available rather than waiting until the end of the iteration.

---

## Learn More

[1] Leffingwell, Dean. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*. Addison-Wesley, 2011.

[2] Leffingwell, Dean. *Scaling Software Agility: Best Practices for Large Enterprises*. Addison-Wesley, 2007.

Last update: 23 November 2022

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## **Training**

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## **Content & Trademarks**

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## **Scaled Agile, Inc**

### **Contact Us**

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### **Business Hours**

Weekdays: 9am to 5pm

Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



[Home](#) » System Demo

## System Demo

“ Base milestones on objective evaluation of working systems.

—Lean-Agile Principle #5

**Definition:** The System Demo provides stakeholders an integrated view of new features for the most recent iteration delivered by all the teams on the ART. Each demo provides an objective measure of progress and the opportunity to give feedback.

## Summary

The System Demo is a key event for Agile Release Trains (ART), that showcases the integration of new features developed by the Agile Teams. The System Demo provides an objective measure of progress toward PI Objectives and fosters actionable feedback. It contrasts with team-specific Iteration Reviews by involving all ART members and stakeholders. The System Demo is led by the RTE.

## What is the System Demo?

The System Demo provides stakeholders with an integrated view of new features delivered by teams on the Agile Release Train (ART) in the most recent iteration. Each System Demo provides an objective measure of progress toward the goals of the PI with the opportunity to give feedback.

The System Demo plays a crucial role in the way an Agile Release Train operates and is attended by everyone on the ART. This contrasts with the specific Iteration Reviews for each team, which focus on the specific deliverables of a single Agile Team within an iteration. Generally, only the Agile Team and any stakeholders involved in that team's work attend Iteration Reviews.

During the System Demo, the Agile Teams on the ART showcase the features they've been working on, demonstrating them from a production-like environment. This provides an objective measure of progress in place of checking off tasks or generating status reports. Regularly demonstrating a working system also helps identify defects or design flaws early on, making them cheaper and easier to fix. It also generates new ideas to improve over time.

In addition to the teams on the ART, the System Demo is attended by business owners, key stakeholders, and often customers. In this way, it creates an opportunity to gain a shared understanding of the current state of the products and solutions on a regular cadence. System Demos create a safe space for early identification of issues and opportunities. It is dedicated time for stakeholders to clearly articulate the potential benefits of features as they are developed. The feedback provided during the System Demo informs future iteration work of the Agile Teams.

System Demos also benefit the Agile Teams on an ART. Demonstrating the integrated system helps teams understand and optimize how their work contributes to the overall solution. Each Agile Team gets visibility into the work of the other teams, fostering collaboration and knowledge sharing. Agile Teams get immediate feedback from stakeholders in the demo so they can make quick changes and focus on what is important. This reduces wasted effort.

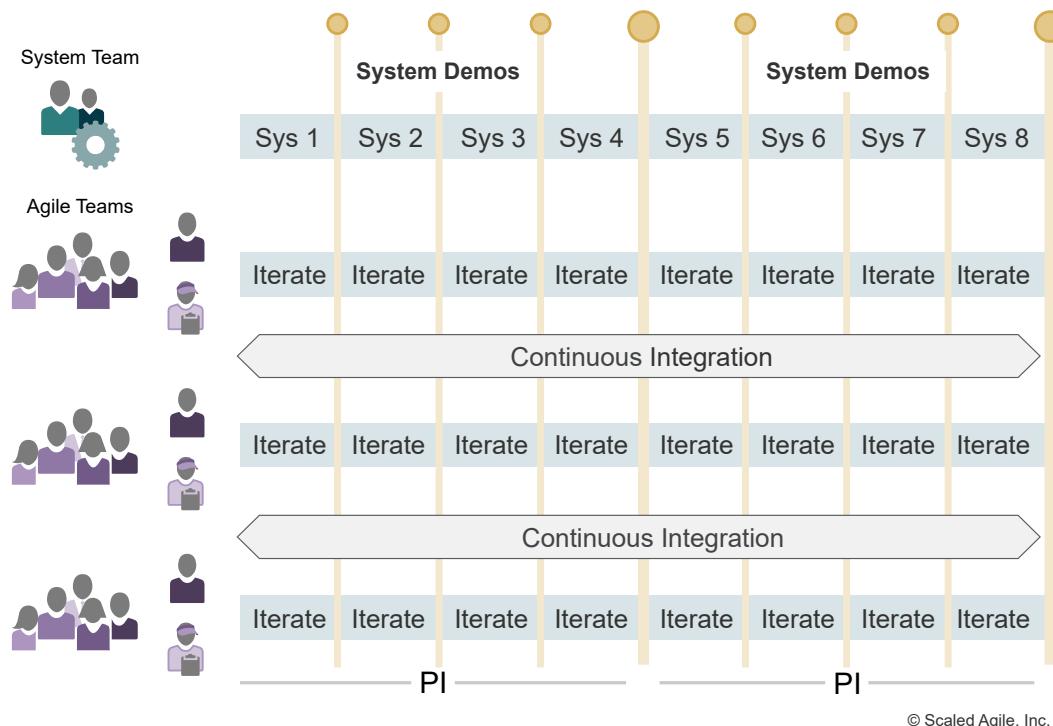


Figure 1. System Demos occur on a cadence at the end of each Iteration

For larger Solutions involving multiple ARTs, this System Demo becomes a part of the broader Solution Demo. Solution Demos have participation from all involved ARTs.

Read more about how multiple ARTs demo their connected solution:

## How often does the System Demo take place?

The System Demo occurs at the end of every Iteration. It takes place as close to the end of the iteration as possible—ideally, the next day. Delaying integration significantly inhibits learning and creates a false sense of progress. To achieve this, ARTs are encouraged to make necessary efforts to ensure demos are conducted regularly. Any delays in the System Demo could point to underlying issues like the need for better integration practices or additional resources for the System Team. Addressing these challenges promptly significantly improves the workflow and outcome.

Sometimes, timely integration and system demos might be challenging for organizations transitioning to Lean and Agile methods or those developing more complex cyber-physical solutions. That's normal and should not be an excuse to reduce the scope or extent of integration. Most of the challenges should disappear as the ART matures. When full integration at every iteration is not achievable, the Agile Teams should consider the following:

- Integrating a subset of capabilities, components, or subsystems to show a particular feature or nonfunctional Requirement (NFR).
- Integrating partially while using physical or digital prototypes in place of scarce or costly components.
- Creating test doubles to speed up integration and testing.
- Integrating less frequently (for example, every other iteration) until it's possible to do it more often.
- Dedicating a System Team to the ART to put in place the infrastructure and capabilities required to integrate frequently.

In addition to the System Demo for each iteration, the ART holds a final PI System Demo that shows all the features developed over the PI. This event is a critical part of the Inspect and Adapt (I&A) event.

Read more about the PI System Demo as part of the Inspect & Adapt event:

Inspect & Adapt

# Who attends the System Demo?

Every member of the ART attends the System Demo whenever possible. Additionally, the feedback from a diverse group of stakeholders plays a crucial role in effectively steering the ART.

Attendees typically include:

- Product Managers and Product Owners, who are usually responsible for running the demo
- One or more members of the System Team, who often help set up the demo in the staging environment
- Business Owners, executive sponsors, customers, and customer proxies
- System Architect, IT operations, and other development participants
- ART Agile team members attend whenever possible

Attendees not only observe the proceedings but actively contribute valuable insights.

Business Owners and Product Managers share how well the work meets customer and business needs. Agile Team members offer technical recommendations, highlight opportunities for further enhancements, and suggest ways to foster better inter-team collaboration to amplify customer value. The System Team members, System Architects, and IT operations personnel ensure a comprehensive view of the solution's alignment with immediate and strategic technical goals.

# How to facilitate the System Demo?

The RTE usually facilitates the System Demo, with active participation from others on the ART. Anyone on the ART can present the work. Team members from multiple teams often collaborate to demonstrate their integrated features together.

Although the System Demo may evolve over time, the following agenda items provide a good starting point:

**Opening:** Start with a brief introduction and provide the context for the System Demo.

**Live demo of features:** Highlight the connection to the PI Objectives and the expected business and customer benefits.

**Questions and feedback:** Create space for discussion.

**Identify risks and impediments:** Agree on the next steps for addressing the raised risks and impediments.

**Conclude:** Summarize the actions from the System Demo. Briefly review the plan for the remainder of the PI.

Keeping the System Demo to no more than an hour helps maintain engagement. It is enough time to show the benefits to the business and customer of the recent work across the ART. It also supports focused participation for higher-impact feedback.

Demonstrating real, tested functionality directly from a production-like environment, rather than relying on extensive preparations or presentations, makes the demo more authentic and focused. Incorporating demonstrations of the system's impact will further highlight the solution's value and effectiveness for attendees.

Here are some additional tips for a successful System Demo:

- Show real, working features rather than slides.
- Show how the system architecture is developing and how NFRs have been applied.
- Provide an overview of the items to be demoed to the ART and stakeholders before the event.

## How do we apply what we learned from the System Demo?

The goal of the System Demo is for all ART members to learn from the most recent development experience and adjust their course of action as needed. The System Demo provides fast feedback, which must then be appropriately actioned.

When a team or teams receive specific feedback in the System Demo, several events in SAFe can help them act on it. Team retrospectives offer moments to evaluate their performance, identify improvements by discussing the system demo feedback, and create action plans. During iteration planning, the feedback from the previous system demo helps shape the priorities of the upcoming iteration and may require new stories to be created.

ART Syncs are a great opportunity for the RTE, Scrum Masters/Team Coaches, Product Management, and Product Owner to discuss the feedback from the System Demo and align to the necessary actions. ART backlog refinement and PI Planning provide opportunities to create new features for feedback that affect multiple teams.

The feedback from the System Demo and the actions taken are critical steps in the ART's continuous improvement journey.

---

## References

[1] Leffingwell, Dean. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*. Addison-Wesley, 2011.

[2] Leffingwell, Dean. *Scaling Software Agility: Best Practices for Large Enterprises*. Addison-Wesley, 2007.

## In this article

**What is the System Demo?**

**How often does the System Demo take place?**

**Who attends the System Demo?**

**How to facilitate the System Demo?**

**How do we apply what we learned from the System Demo?**

## Key Takeaways

- The System Demo is an event where the Agile Teams on an ART showcase new features from the most recent iteration to stakeholders.
- The System Demo includes the active involvement of all ART members and stakeholders of the ART.
- The System Demo enables the attendees to provide fast feedback, allowing teams to make necessary adjustments to the solutions they deliver.

Last Update: 15 October 2024

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## **Framework**

[Download SAFe Posters & Graphics](#)

[Blog](#)

## **Training**

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## **Content & Trademarks**

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## **Scaled Agile, Inc**

### **Contact Us**

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### **Business Hours**

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



+ Framework

 ENGLISH (US)

**“** Inspection does not improve the quality, nor guarantee quality. Inspection is too late. The quality, good or bad, is already in the product. Quality cannot be inspected into a product or service; it must be built into it.

—W. Edwards Deming

## Ready to start learning?

Use our finder to explore current offerings or learn more about a specific course

Course

SAFe for Teams

[See available classes](#)

[Learn more about the course](#)

# Built-In Quality

Built-In Quality is a set of practices to help ensure that the outputs of Agile teams in business and technology domains meet appropriate quality standards throughout the process of creating customer value.

## Details

To support [Business Agility](#), enterprises must continually respond to market changes. The quality of the work products that drive business value directly determines how quickly the teams can deliver their solutions. Although work products vary by domain, they are likely to involve software, hardware designs, scripts, configurations, images, marketing materials, contracts, and other elements. Products built on stable foundations that follow standards are easier to change and adapt. Built-in quality is even more critical for large solutions, as the cumulative effect of even minor defects and wrong assumptions may create unacceptable consequences.

Building quality in requires ongoing training and commitment. But the benefits warrant the investment and include:

- Higher customer satisfaction
- Improved velocity and delivery predictability
- Better system performance
- Improved ability to innovate, scale, and meet compliance requirements

Built-in quality is linked to the fast flow of value described in [SAFe principle 6: Make value flow without interruptions](#). Accelerating problem discovery and taking corrective action occurs by shifting learning left on the timeline. Improved collaboration, workflow automation, more frequent delivery, and faster customer feedback support a quicker learning process.

SAFe applies Built-in Quality across five key domains. Each domain has a set of quality practices that vary from universally applicable generic practices to those specific to one or a few domains.

Figure 1 provides a consolidated view of Built-in Quality in SAFe.

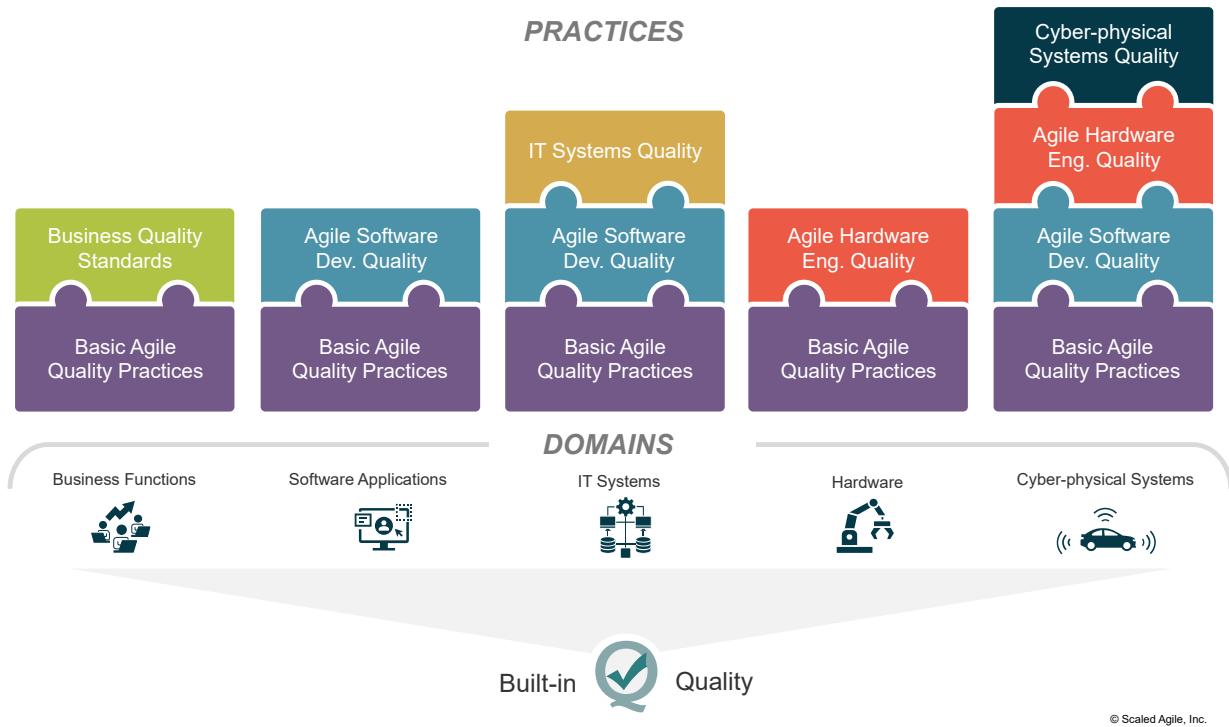


Figure 1. Key domains and practices of Built-In Quality in SAFe

© Scaled Agile, Inc.

The rest of this article describes the components of Figure 1 in deeper detail.

## Built-in Quality Domains



Built-in Quality practices vary based on the domains in which they are applied. Despite the same intent behind the Built-in Quality approach to creating customer value, the actual practices reflect the intricacies of their environment and context. The following are the Built-in Quality domains in SAFe:

### Business Functions

Business functions include marketing, sales, HR, finance, supply chain management, and other non-IT disciplines. Along with routine operations, each function also includes complex efforts requiring specific quality outputs for success. For example, creating a new marketing campaign or establishing new HR policies involve certain quality expectations.

### Software Applications

Software is an essential contributor to business agility, the ability to scale the business, and better compete in the digital age. But seizing such opportunities requires maintaining predictable quality when delivering solutions.

### IT Systems

IT infrastructure powers vast ecosystems of today's enterprise solutions landscape. The more complex the solutions, the more sophisticated the IT systems must be to sustain them. To support the reliable operation of the enterprise, IT systems require substantial quality standards and, therefore, proper quality practices.

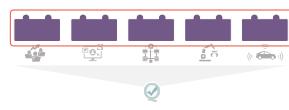
## Hardware

When used in computer technology, hardware typically refers to cables, monitors, integrated circuits, and other tangible elements of a computer system. But more generally, hardware refers to devices with concrete physical properties: mass, size, and matter. Examples include motors, gears, tools, chassis, cases, and simple or complex mechanisms. Due to their significantly higher cost of change, hardware systems require a unique approach to quality.

## Cyber-physical Systems

Cyber-physical systems are complex systems wherein multiple physical elements are controlled by software algorithms. Examples include robots, aircraft, and automobiles. These are some of the world's most complex systems and often include intricate electrical, mechanical, optical, fluidic, sensory, and other subsystems. Their complexity and the high impact of failure emphasize the critical importance of quality in such systems.

## Basic Agile Quality Practices



Basic Agile quality practices can be applied to work products in any domain. They have proven their worth and provide a common starting point for knowledge workers to understand and improve the quality attributes of the artifacts, work products, systems, and services that benefit themselves and their customers. A set of five SAFe Basic Agile Quality Practices are described in the sections below.

## Shift Learning Left

Every development effort involves numerous unknowns that surface as development progresses and teams learn new facts. If the learning happens late in the process, underlying issues will significantly impact the solution, and significant rework and delays will result. However, if learning takes place much earlier—or is *shifted left*—problems reveal themselves sooner, enabling corrective action with minimum impact (Figure 2).

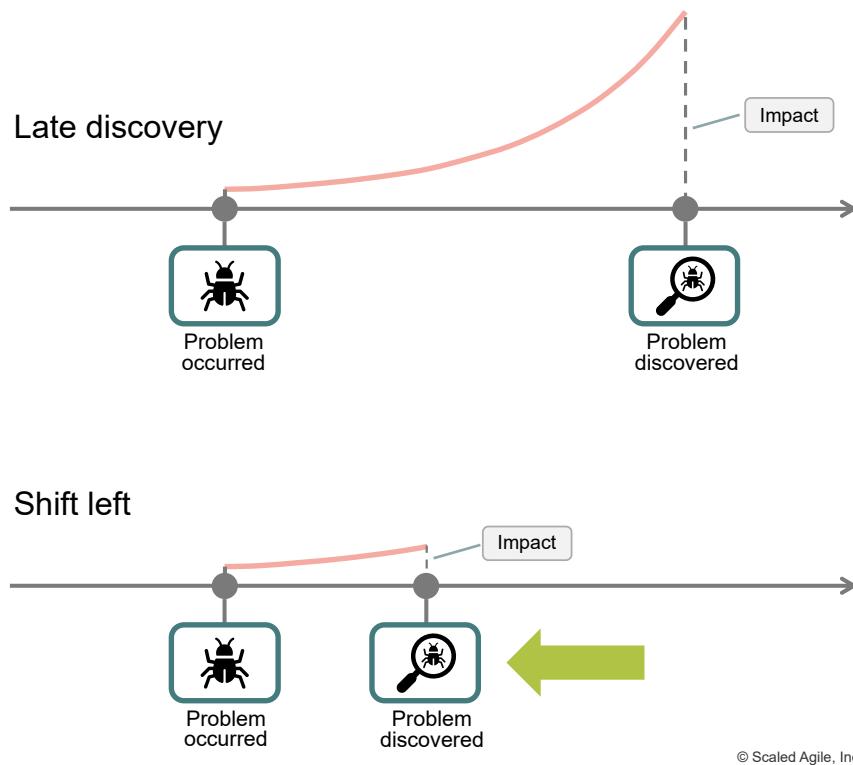


Figure 2. Shifting quality left reveals problems sooner

Shifting learning left does not simply mean that some actions take place earlier on the timeline but also that the structure of some of the basic processes is changed. For example, a test-first approach requires shifting away from conventional testing. Instead, tests are created whenever possible *before* the desired solution functions are implemented.

## Pairing and Peer Review

Pair work describes a practice wherein two knowledge workers collaborate over the same asset in real time. Often, one serves as the driver, directly advancing the work product, while the other acts as the navigator, providing real-time evaluation and feedback. Team members switch roles frequently. Because the work product will contain each member's shared knowledge, perspectives, and best practices, pairing creates and maintains higher quality. As teammates learn from each other, the skillsets of the entire team rise and broaden. Additionally, peer review helps spot quality issues as one team member examines the work products of the other. Many governance processes around software, for example, mandate peer review as a compliance activity.

## Collective Ownership and T-shaped skills

Collective Ownership is a quality practice where individual team members have the requisite skills and authority to update any relevant asset. This approach reduces dependencies between teams and ensures that any individual team member or team will not block the fast flow of value delivery. Any individual can add functionality, fix errors, improve designs, or refactor *because the work product is not owned by one team or individual*. Collective ownership is supported by quality standards that encourage consistency, enabling everyone to understand and maintain the quality of each component. Collective ownership is further enabled by 'T-shaped skills.' T-shaped

skills characterize individuals who possess deep experience in one area but also have broad skills in other areas. T-shaped skills also represent the ability to work well with others.

## Artifact Standards and definition of done

Assets created and maintained by the organization must adhere to standards that help ensure their value to the business. These standards may reflect how the artifacts are being built or what properties they must manifest. Standards are often unique to the specific organization and solution context, emerging gradually, validated frequently, and corrected by multiple feedback cycles. To productively maintain artifact standards, the teams must understand the motivations for their existence. Artifact design practices and the effective use of automation help facilitate standards. Enacting productive artifact standards involves applying a definition of done (DoD) – an essential way of ensuring that a work product is complete and correct. Each team, train, and enterprise should build a DoD that suits their needs.

## Workflow Automation

Workflows tend to have many manual steps. Handoffs from one worker to another, searching for an asset of interest, and manual inspection of an asset to a standard are just a few examples. The fact is, all these manual steps are error-prone and cause delays in the process. Many of these tasks can be automated if the teams take the time to invest in a more automated pipeline that supports the activities. Automation provides substantial gains due to reduced execution costs and intrinsic adherence to standards. Of course, this can be done incrementally, and it often starts by putting a Kanban system in place and then noting steps that can be automated. Sometimes the first step is simply setting up automated notifications when an item changes state. Even simpler, many such systems are designed as true pull systems where the worker simply checks the system to see what work is available to them based on its state. In this case, the handoff is automatic and doesn't require separate communications overhead just to know the state of a work product.

## Business Quality Standards



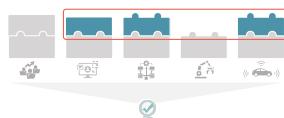
The above sections describe a set of five basic Agile quality practices that can be applied to every business domain. Virtually every aspect of business operations—accounting and finance, legal, sales, development, HR, marketing, operations, production, and more—is subject to internally or externally imposed quality standards, which are often linked to compliance requirements. Each business function produces specific outputs, which must satisfy quality standards relevant to that context.

No matter your business function, the steps to achieve quality with Agility include the following:

- Organize into Agile teams, get trained, and iterate.
- Define the standards and compliance policies for your function.
- Agree on the definition of done (DoD) for artifacts and activities for your workflow.

- Implement the basic Agile quality practices.
- Measure and learn. Specialize Agile quality practices further to your specific function.
- Improve relentlessly.

## Agile Software Development Quality Practices



Software may well be the richest and best-defined area for applying Built-in Quality. This was driven by necessity, as software is exceedingly complex and intangible. You can't touch it or see it, so traditional approaches to inspecting, measuring, and testing are inadequate. If quality isn't built in endemically, then it's unlikely to exist at all. To address this new challenge, many new quality practices like those above were inspired by Extreme Programming (XP), which has a zest for going fast with quality. They have proven their worth and have now started influencing quality practices in other domains. The practices below apply well to software development, and we will describe them in that context, but they can be applied to other domains as well.

### Continuous Integration

Building large-scale value requires knowledge workers to build the system in increments, resulting in frequent small changes. Each must be continually checked for conflicts and errors and integrated with the rest of the system to assure compatibility and forward progress. [Continuous Integration](#) (CI) provides developers with fast feedback (Figure 3). Each change is quickly built, integrated, and then tested at multiple levels. CI automates the process of testing and migrating changes through different environments, notifying developers when tests fail.

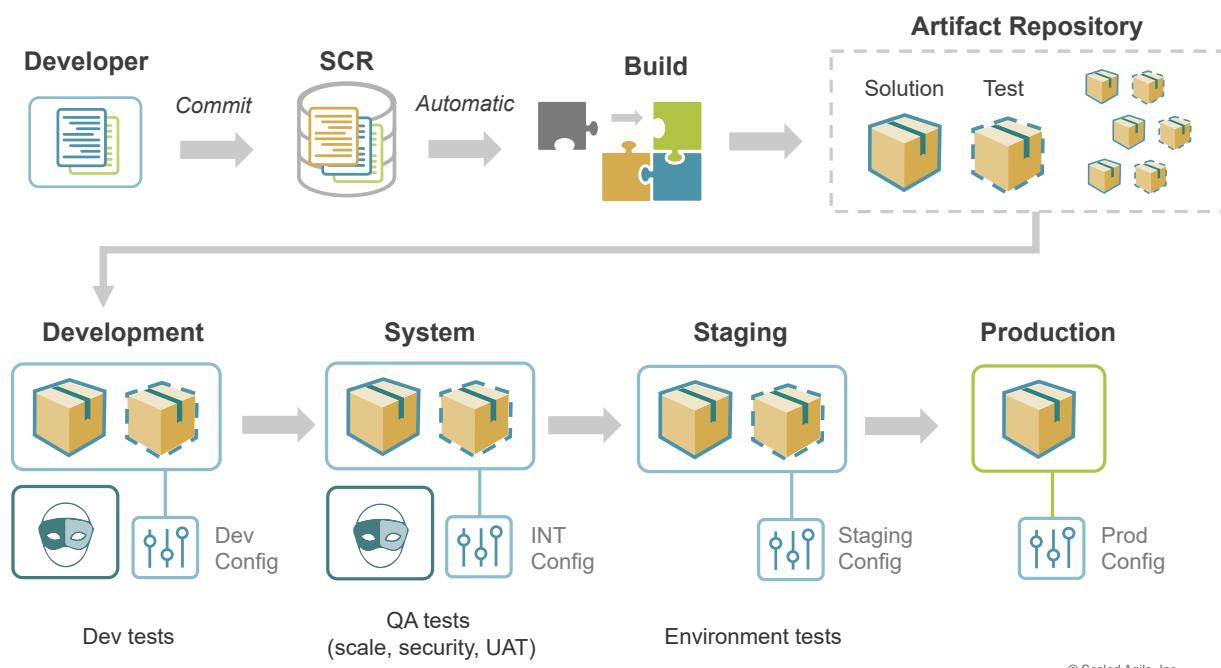


Figure 3. Continuous integration (CI) fosters system-wide quality

Continuous integration is vital within and across teams, allowing them to quickly identify and resolve issues in all parts of the codebase.

## Test-first Practices

Agile teams operate in a fast, flow-based system to develop and release high-quality business capabilities quickly. Instead of performing most of the testing at the end, Agile teams define and execute many tests early and often as a part of their integration process. Tests are defined for small units of code using **Test-Driven Development** (TDD), for **Story**, **Feature**, and **Capability** acceptance criteria using **Behavior-Driven Development** (BDD), and for the feature or capability benefit hypothesis using **Lean UX** (Figure 4). Building quality in ensures that Agile development's frequent changes do not introduce new errors while enabling fast, reliable execution.

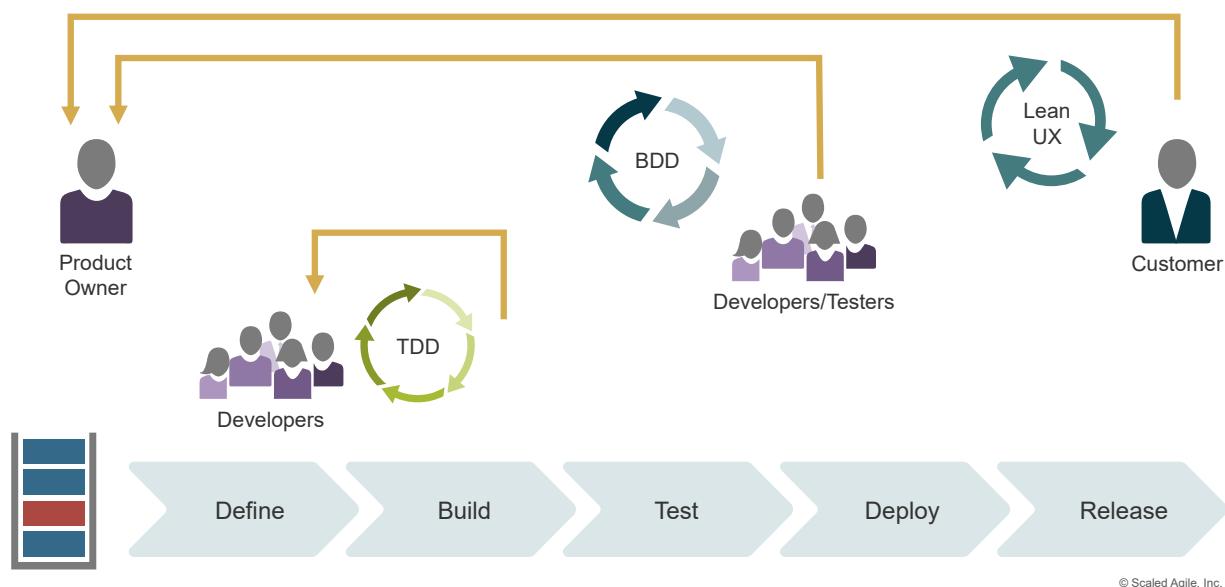


Figure 4. Test-first practices accelerate flow

## Refactoring

Constantly changing technology and evolving business objectives make it difficult to maintain and continually increase business value. However, two paths to the future exist:

- Keep adding new functionality to an existing code base toward an eventually unmaintainable 'throw-away' state
- Continuously refactor the system to build a foundation for efficiently delivering the current business value as well as future business value

Refactoring, which improves the internal structure or operation of an area of code without changing its external behavior, is better. With continuous refactoring, the useful life of an enterprise's investment in software assets can be extended substantially, allowing users to benefit from a flow of value for years to come. But refactoring takes time, and the return on investment is not immediate, so an allowance for time and effort must be part of capacity planning considerations. For more, see the extended guidance article on [Refactoring](#).

## Continuous Delivery

Continuous delivery provides the ability to release value to customers whenever they need it. This is accomplished by the [Continuous Delivery Pipeline](#) (CDP), which contains four aspects: *continuous exploration*, *continuous integration*, *continuous deployment*, and *release on demand*. The CDP enables organizations to map their current pipeline into a new structure and use relentless improvement to deliver value to customers. Feedback loops internally within and between the steps and externally between the customers and the enterprise fuel improvements. Internal feedback loops often center on *process* improvements; external loops often center on *solution* improvements. The improvements collectively create synergy, ensuring the enterprise is ‘building the right thing, the right way’ and frequently delivering value to the market. Additionally, SAFe [DevOps](#) features crucial practice domains for establishing fast and reliable value delivery mechanisms.

Continuous delivery helps SAFe teams release on demand. Releasing with quality, however, requires a specific, scalable definition of done that helps ensure that the requisite quality is built in. Figure 5 shows an example:

Team Increment	System Increment	Solution Increment	Release
<ul style="list-style-type: none"><li>• Stories satisfy acceptance criteria</li><li>• Acceptance tests passed (automated where practical)</li><li>• Unit and component tests coded, passed, and included in the build verification test</li><li>• Cumulative unit tests passed</li><li>• Assets are under version control</li><li>• Engineering standards followed</li><li>• NFRs met</li><li>• No must-fix defects</li><li>• Stories accepted by Product Owner</li></ul>	<ul style="list-style-type: none"><li>• Stories completed by all teams in the ART and integrated</li><li>• Completed features meet acceptance criteria</li><li>• NFRs met</li><li>• No must-fix defects</li><li>• Verification and validation of key scenarios</li><li>• Included in build definition and deployment process</li><li>• Increment demonstrated, feedback achieved</li><li>• Accepted by Product Management</li></ul>	<ul style="list-style-type: none"><li>• Capabilities completed by all trains and meet acceptance criteria</li><li>• Deployed/installed in the staging environment</li><li>• NFRs met</li><li>• Solution end-to-end integration, verification, and validation done</li><li>• No must-fix defects</li><li>• Included in build definition and deployment process</li><li>• Documentation updated</li><li>• Solution demonstrated; feedback achieved</li><li>• Accepted by Solution Management</li></ul>	<ul style="list-style-type: none"><li>• All capabilities done and meet acceptance criteria</li><li>• End-to-end integration and solution V&amp;V done</li><li>• Regression testing done</li><li>• NFRs met</li><li>• No must-fix defects</li><li>• Release documentation complete</li><li>• All standards met</li><li>• Approved by Solution Management</li><li>• Satisfies release governance requirements</li></ul>

© Scaled Agile, Inc.

Figure 5. An example of a scalable definition of done

To support security practices, teams generate a Software Bill of Materials (SBOM) for each release describing the commercial and open-source components and dependencies to ensure no vulnerabilities.

## Agile Architecture

[Agile Architecture](#) is a set of values, practices, and collaborations that support a system’s active, evolutionary design and architecture. It embraces the DevOps mindset, allowing a system’s architecture to evolve continuously while simultaneously supporting the needs of current users.

Agile architecture supports Agile development practices through collaboration, emergent design, intentional architecture, and design simplicity. It also enables designing for testability, deployability, and changeability. Rapid prototyping, set-based design, domain modeling, and decentralized innovation, in turn, support Agile architecture.

The essential concept of [Architectural Runway](#) allows Agile teams and trains to provide effective enablement for future business capabilities and features while progressively validating underlying architectural assumptions.

## IT Systems Quality Practices



Every modern enterprise depends on properly functioning IT systems for its business success. With more and more business workflows being powered by IT, ensuring the reliability, scalability, safety, and security of IT systems becomes increasingly important. It requires a robust approach to building quality into these systems. A sample of IT-specific quality practices is described below.

### Infrastructure as Code

One of the critical challenges in ensuring the quality of IT ecosystems comes from defining and sustaining configurations consistently. Often representing hundreds or even thousands of environment parameters, configurations grow out of sync and cause problems in different parts of the enterprise's solution landscape. 'Infrastructure as Code' is an approach to control those configurations programmatically and thus benefit fully from automation in defining, procuring, and maintaining configurations consistently and integrally. Containerization is an excellent enabler of Infrastructure as Code, as it permits applying programming interfaces to various aspects of the execution environment. Additionally, using 'immutable infrastructure'—an approach where IT components are rebuilt whenever needed, rather than modified in production—forces the organization to explicitly control all changes to the environment by formally redefining them and redeploying the component that changed.

### NFRs and SLAs

IT infrastructure must provide certain qualities to the execution environment to support the systems essential to business operation. These quality attributes include things such as security, reliability, performance, maintainability, and scalability (Nonfunctional Requirements or NFRs). Additionally, relevant Service-Level Agreements (SLAs), such as Mean Time Before Failure (MTBF) and Mean Time to Repair (MTTR), must be ensured. In SAFe, NFRs and SLAs are achieved incrementally by early and continuous testing and timely corrective action. Ensuring that systems meet their NFRs and SLAs requires instrumentation and the proactive build and use of the architectural runway.

### Telemetry and Monitoring

Responding to unanticipated loads, security attacks, hardware, software, and network failures, require a range of options, from downgrading or removing services to adding service capacity. Telemetry and logging capabilities allow organizations to understand and fine-tune their architecture and operating systems to meet intended loads and usage patterns. Effective monitoring requires that full-stack telemetry is active for all features deployed through the CDP. Monitoring ensures that issues with system performance can be anticipated or addressed rapidly in production.

## Cybersecurity Standards

IT environments must meet increasingly stringent quality standards to protect against unauthorized access, use, disclosure, or destruction. The spectrum of activities to achieve comprehensive cybersecurity includes:

- Technology enablement (data encryption, streamlined identity management, etc.)
- Frequent testing and validation (audits, penetration testing, etc.)
- Training and proper habits for the workforce
- Testing of all new assets for various vulnerabilities
- Frequently review new vulnerability alerts against existing solution's SBOM for affected components and provide patches or hotfixes

## Automated Governance

Recent advances in DevOps and related methods, practices, and tooling provide new opportunities for IT teams to automate governance. Automated governance replaces tedious, manual, and error-prone activities and specifically addresses security, compliance, and audit needs. For more on this topic, see the reference [2]: Investments Unlimited, A Novel about DevOps, Audit Compliance, and Thriving in the Digital Age.

Automation of configuration management, audit, security testing (during both build and deployment), and immutable infrastructure help reduce human error that can lead to system vulnerabilities.

## Agile Hardware Engineering Quality Practices



Ensuring quality in hardware systems and components is complicated because the cost of change increases with time, and the impact of quality issues with hardware is high. This can include catastrophic field failure, recalls of volumes of manufactured products, and expensive field replacement or repair. This risk pressures organizations to effectively apply Built-in Quality practices while developing engineered hardware systems and subsystems. There are several techniques organizations use to ensure Built-in-Quality in hardware systems, which are described below.

## Modeling and Simulation

In Agile, the goal is to build and learn as quickly as possible. Modeling and simulation in the virtual environment—and rapid modeling in the prototype environment—help shift learning left, as shown in Figure 6.

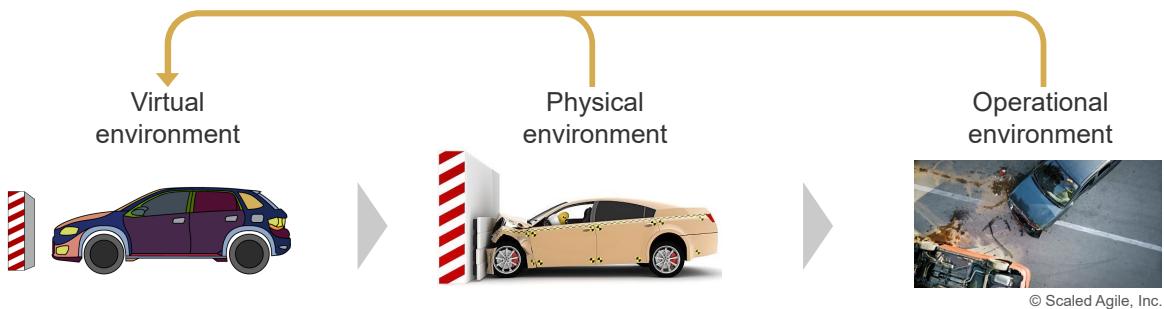


Figure 6. Shift learning left for hardware engineering

Analysis and simulation in digital models used in electrical and mechanical Computer-Aided Design (CAD) and MBSE (see below) can test changes quickly and economically. Digital twins combine multiple virtual models with data harvested from telemetry in the operational systems to improve the models and better predict how systems will behave in the future. The feedback loops in Figure 6 show how data from other environments validate and improve the digital environment. Some aerospace and automotive products even use model simulations for certification, substantially reducing the time and cost of changes.

## Rapid Prototyping

The virtual environment cannot reveal all issues. Physical prototypes are a lower-cost substitute for real, “bent metal” hardware. They provide higher-fidelity feedback, available only in a physical environment. Example prototype practices include:

- Wood and other low-fidelity mockups
- Breadboarding electrical components
- 3d-printed mechanical and electrical parts (PCBs, wiring harnesses)

Increasingly, additive manufacturing is used to lower the costs of rapid experimentation and prototyping. “Additive manufacturing uses data computer-aided-design (CAD) software or 3D object scanners to direct hardware to deposit material, layer upon layer, in precise geometric shapes. As its name implies, additive manufacturing adds material to create an object. By contrast, when you create an object by traditional means, it is often necessary to remove material through milling, machining, carving, shaping, or other means.”[3]

Many organizations with the equipment and knowledge to ‘print’ mechanical and electrical parts can produce and ship them in a single day. And parts made with additive manufacturing are now making their way into production.

# Cyber-physical Systems Quality Practices



Cyber-physical systems require an organization to deal effectively with hardware components and the software that governs its behavior. Additionally, because such systems operate directly in the real world, the impact of quality issues can be significant and often subject to regulatory compliance.

## Model-Based Systems Engineering

**Model-Based Systems Engineering** (MBSE) is the practice of developing a set of related digital models that help define, design, and document a system under development. These models provide an efficient way to explore, update, and communicate system aspects to stakeholders while significantly reducing or eliminating dependence on traditional documents. By testing and validating system characteristics early with the model, they facilitate timely learning of properties and behaviors, enabling fast feedback on requirements and design decisions.

## Frequent End-to-end Integration

In the software domain, continuous integration is the heartbeat of continuous delivery: It's the forcing function that verifies changes and validates assumptions across the entire system. Agile teams invest in automation and infrastructure that builds, integrates, and tests every developer change, providing immediate feedback on errors.

Large, cyber-physical systems are far more challenging to integrate continuously because:

- Long lead-time items may not be available
- Integration spans organizational boundaries
- Automation is rarely end-to-end
- The laws of physics dictate certain limitations

Instead, *frequent* end-to-end integration addresses the economic tradeoffs of the transaction cost of integrating versus delayed knowledge and feedback (Figure 7).

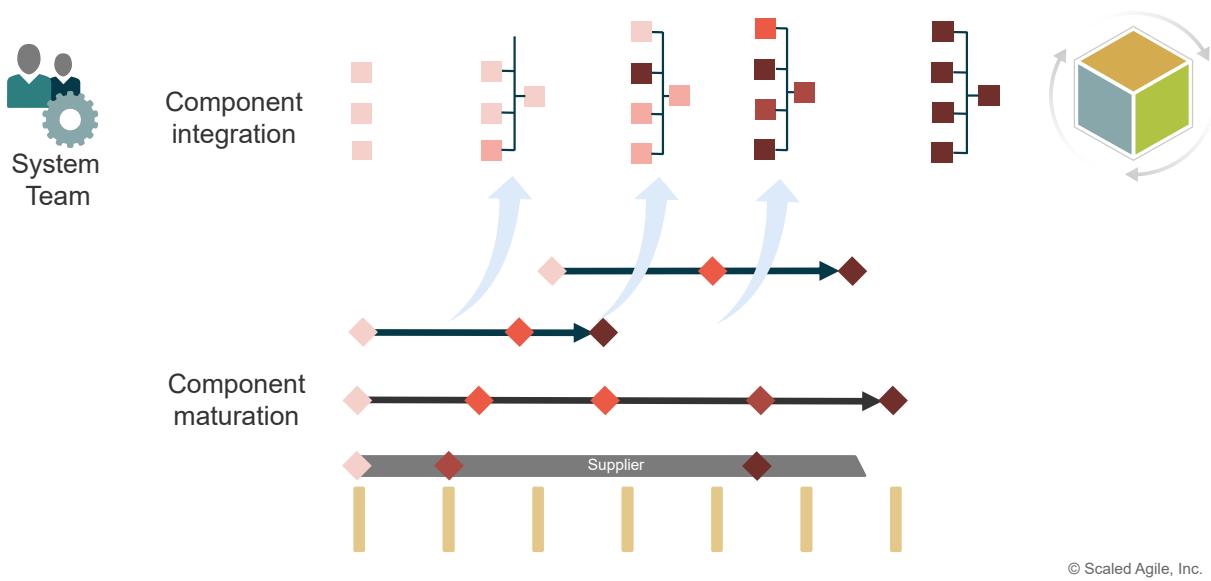


Figure 7. Frequent end-to-end integration

The goal is frequent partial integration with at least one complete solution integration for each PI.

---

## Learn More

[1] [https://en.wikipedia.org/wiki/Software\\_supply\\_chain](https://en.wikipedia.org/wiki/Software_supply_chain)

[2] Beal, Helen and Bill Bensing , Jason Cox , Michael Edenzon , John Willis. *Investments Unlimited: A Novel about Devops, Security, Audit Compliance, and Thriving in the Digital Age*, IT Revolution Press, 2022

[3] <https://www.ge.com/additive/additive-manufacturing>

Last updated: 18 January 2022

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## **Training**

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## **Content & Trademarks**

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## **Scaled Agile, Inc**

### **Contact Us**

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### **Business Hours**

Weekdays: 9am to 5pm

Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

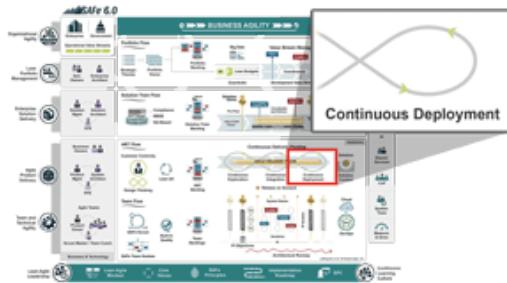
[Code of Conduct](#)

[Remote Training Policy](#)



+ Framework

ENGLISH (US) ▾



“In order for you to keep up with customer demand, you need to create a deployment pipeline. You need to get everything in version control. You need to automate the entire environment creation process. You need a deployment pipeline where you can create test and production environments, and then deploy code into them, entirely on demand.

—Erik to Grasshopper, *The Phoenix Project*

## Continuous Deployment

Continuous Deployment (CD) is an aspect of the Continuous Delivery Pipeline that automates the migration of new functionality from a staging environment to production, where it is made available for release.

CD is the third aspect in the four-part [Continuous Delivery Pipeline](#) (CDP) of [Continuous Exploration](#) (CE), [Continuous Integration](#) (CI), Continuous Deployment (CD), and [Release on Demand](#) (Figure 1).

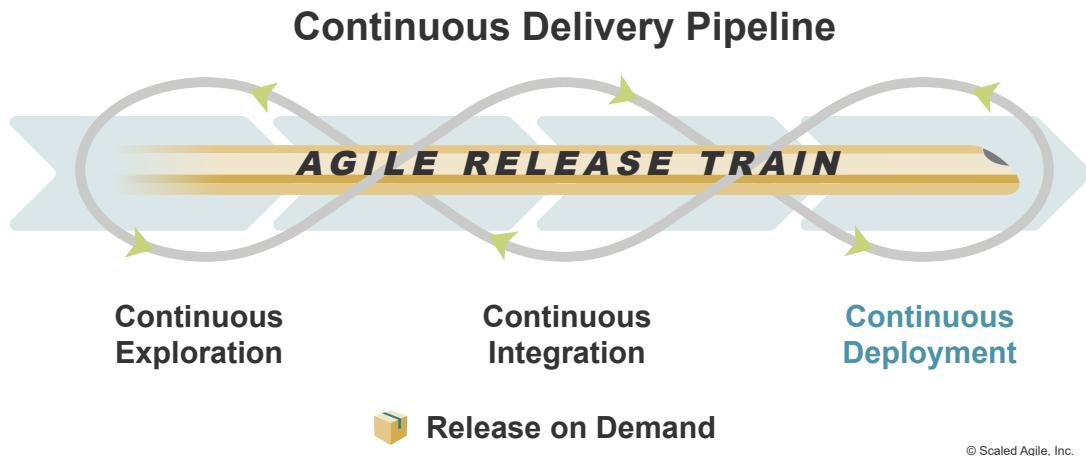


Figure 1. Continuous deployment in the context of the CDP

© Scaled Agile, Inc.

Features must be available and verified in production *before* the business needs them to support [Release on Demand](#). Therefore, it's optimal to separate the deployment process from releasing, enabling changes to move into production without affecting the behavior of the current system. Continuous deployment allows teams to deploy small, incremental changes to production continually.

The capability to continuously deploy is critical for releasing on demand. In turn, it allows Agile Release Train (ARTs) to respond to market opportunities with the highest possible value in the shortest sustainable lead time, permitting customers to consume new functionality when they are ready.

## Details

Traditional development practices treat deployment and release as the same activity. In this model, changes deployed to production are immediately available to users. Continuous deployment, however, separates the deployment and release processes. This practice fosters design thinking and fast value flow by:

- **Targeting functionality to specific customers** – Enables the organization to target customers with particular functionality, allowing the organization to assess the impact of changes before deploying functionality to all customers.
- **Promoting experimentation, such as A/B Testing** – Design thinking practices, such as A/B testing, require the ability to present different functionality to distinct target users, gathering feedback that helps create the optimal user experience.
- **Promoting small batches** – Automating the CDP (for example, tests, builds, deploys) makes deploying in small batches economically feasible.
- **Releasing on business needs** – ARTs tend to release less frequently when the deployment process is complex and error-prone. Organizations that invest in automation and relentless process improvement can release faster and with lower risk, substantially increasing

**Business Agility.** For example, a release can be deployed in production ahead of a marketing campaign, giving the organization more flexibility in maximizing all aspects of value delivery.

To enable these capabilities, ARTs focus on reducing the transaction cost and risk of moving changes to production by automating all aspects of continuous deployment. Ensuring the deployment process is a repeatable, predictable activity without significant incidents helps teams achieve continuous deployment. Moreover, improving deployment to make value flow without interruption [Principle #6](#)) is critical for achieving business agility. See the [ART Flow](#) article for more information.

## The Four Activities of Continuous Deployment

SAFe describes four activities of Continuous Deployment, as illustrated in Figure 2.

1. **Deploy** – the practices necessary to deploy a solution to a production environment
2. **Verify** – the practices needed to ensure solution changes operate in production as intended before releasing them to customers
3. **Monitor** – the practices to monitor and report on any issues that may arise in production
4. **Respond** – the practices to address any problems rapidly which may occur during deployment

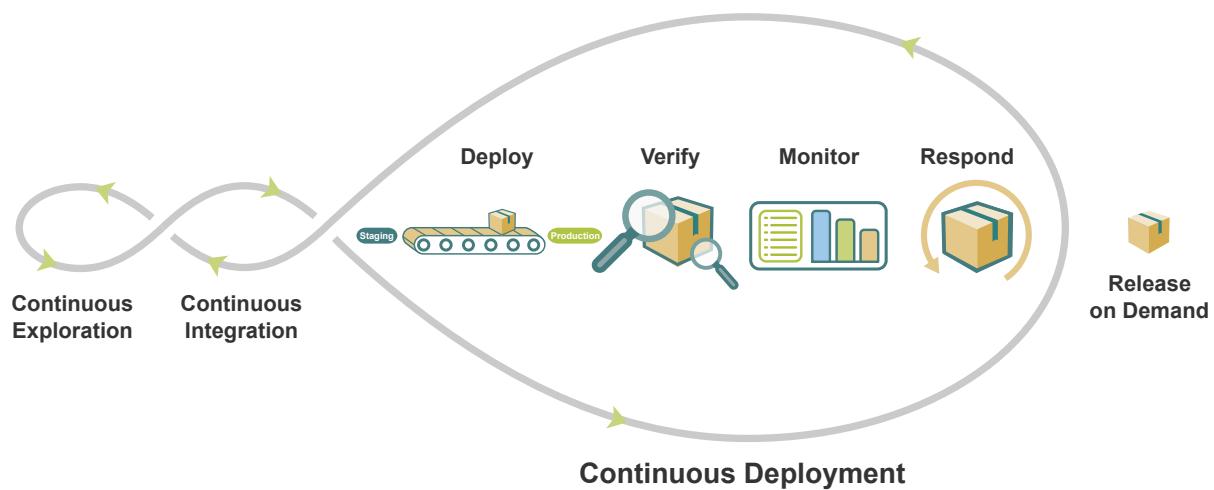


Figure 2. Four activities of continuous deployment

© Scaled Agile, Inc.

## Deploy

Deployment is the migration of changes into a production environment. In the CDP, deploying changes is done continuously. Partial functionality can be implemented incrementally into production.

Suppose a user story map has 27 stories to implement a new workflow. Traditional legacy practices would likely result in all 27 stories deployed in one large batch. Instead, individual stories can be deployed in ‘dark mode’ with continuous deployment and feature toggles. When the ART has deployed the full features set, the business can choose when to release them to users.

Ideally, the deployment pipeline triggers the deployment process automatically following a successful build, integration, and validation. This approach makes the workflow a fully-automated one-click process, from code-commit to production-deploy. Highly sophisticated enterprises can reliably deploy anytime, even during peak periods. This approach eliminates the need to work weekends, nights, or other off-hours to deploy.

Several practices contribute to the ability to deploy:

- **Dark launches** – the ability to deploy new functionality to a production environment without releasing it to end users
- **Feature toggles** – a technique to facilitate dark launches by implementing toggles in the code, which enables switching between old and new functionality
- **Deployment automation** – the ability to deploy a tested solution automatically from staging to production
- **Selective deployment** – the ability to deploy to specific production environments and not others based on criteria such as geography, market segment, and more
- **Self-service deployment** – when automated deployment is partially implemented, self-service allows a single command to move solutions from staging to production
- **Version control** – maintaining environments under version control enables fast deployment and recovery
- **Blue/green deployment** – a technique that permits on-demand switching between staging and production environments

## Verify

Deployments must be verified for functional integrity and robustness before releasing to end users. These two processes almost happen simultaneously when tightly coupled, making recovery decisions a primary concern. However, when they are separated, there’s room to test new functionality extensively in production *before* approving it for release. After migration to production, solutions undergo a final round of testing. Typically, this requires a smoke test, light user acceptance testing, and a stress and performance test, which must occur in a production setting. This verification provides the necessary sanity check that tests the behavior of the solution in its actual production [Solution Context](#).

[Continuous Integration](#) reasonably ensures that the solution will behave as expected in production; however, surprises do occur. When verification reveals critical defects, deployments must either be rolled back or fixed quickly to prevent them from harming the production environment or disrupting the business flow.

Four practices help drive verification after deployment:

- **Production testing** – testing solutions in production using feature toggles or ‘dark’ launches
- **Test automation** – the ability to automate tests and run them rapidly and repeatedly
- **Test data management** – managing test data in version control to ensure consistency in automated testing
- **Testing nonfunctional requirements (NFRs)** – teams also test system attributes such as security, reliability, performance, scalability, and usability to ensure NFRs meet quality standards

## Monitor

Verifying that deployed features didn’t break on their way into production is an essential pre-release quality check. However, teams must also ensure they can measure a feature’s performance and value over its lifespan. The insights that drive this critical feedback loop primarily come from robust monitoring capabilities, which must be in place before release.

Effective monitoring requires that full-stack telemetry is active for all features deployed through the CDP. This telemetry allows teams to verify system performance, end-user behavior, incidents, and business value rapidly and accurately in production. The data collected provides tracking and monitoring of each feature, increasing the fidelity of analysis of the business value delivered and increasing responsiveness to production issues.

While teams cannot collect some business-value metrics until release, they need to know how to obtain the measures before the release decision occurs. Some practices which help support this include:

- **Full-stack telemetry** – the ability to monitor for problems across the entire stack that a system covers
- **Visual displays** – tools that display automated measurements
- **Federated monitoring** – consolidated monitoring across applications in the solution that creates a holistic view of problems and performance

The [Measure & Grow](#) article provides some guidance on the types of metrics that monitoring requires.

## Respond

The ability to respond to and recover from unforeseen production issues is critical to supporting continuous deployment and streamlining the CDP. The reasons are obvious:

- Production issues directly affect customers and end users, so the value of deployed solutions can quickly erode when problems occur.
- Production issues causes rework—fixes, patches, redevelopment, retesting, redeployment, etc. That disrupts the normal flow of value through the pipeline.

Since production issues can harm delivery efficiency and lower value, teams need the capability to detect problems proactively and recover quickly. As measured by Mean Time to Restore (MTTR), fast recovery is among the most reliable leading indicators of high [DevOps](#) maturity [5]. Recovery is also one of the five elements of SAFe's [CALMR](#) approach to DevOps.

The goal of responding and recovering is to identify potential issues *before* they turn into incidents and to prevent them from affecting business operations. This capability requires detecting difficulties internally before end users discover them, quickly identifying root causes, and restoring services with well-rehearsed procedures. In contrast, making hasty, reactive changes directly to production systems—‘just to keep the lights on’—invites source code and configuration differences between environments, unverified changes, and long-term risk.

Several practices support the ability to respond and recover from production issues:

- **Proactive detection** – a technique for proactively creating faults in the solution to identify potential problems and situations before they occur. For example, Chaos Monkey [1], developed by Netflix, is an open-source tool that randomly terminates instances in production to ensure that engineers implement their services to be resilient to instance failures.
- **Cross-team collaboration** – a mindset of cooperation across the [Value Stream](#) to identify and solve problems as they arise
- **Session replay** – the ability to replay end-user sessions to research incidents and identify problems
- **Rollback and fix forward** – the ability to both rollback a solution quickly to a previous environment or to fix a problem quickly through the pipeline without the need to rollback
- **Immutable infrastructure** – This concept refers to never amending servers or Virtual Machines (VMs) after deployment. Instead, a new server is built from an image with the appropriate changes if something needs updating.
- **Version control** – environments should be maintained under version control to rollback quickly

After teams have demonstrated that features have been deployed successfully to production and have the necessary monitoring and recovery capabilities to track and manage ongoing value, they have completed the *continuous deployment* stage of the CDP. In turn, this gives the enterprise the ability to release whenever warranted.

## Enabling Continuous Deployment with DevOps

Continuous deployment involves critical operation activities frequently associated with the 'Ops' in DevOps. They focus on deploying solutions to production environments, verifying their functional integrity, and ensuring effective monitoring and post-release support.

Figure 3 illustrates that SAFe's CALMR approach to DevOps (center) and several practice domains (inner rings) enables continuous deployment. Each of the four activities (in green) is a collaborative effort that draws upon DevOps expertise from multiple disciplines to maximize delivery speed and quality.

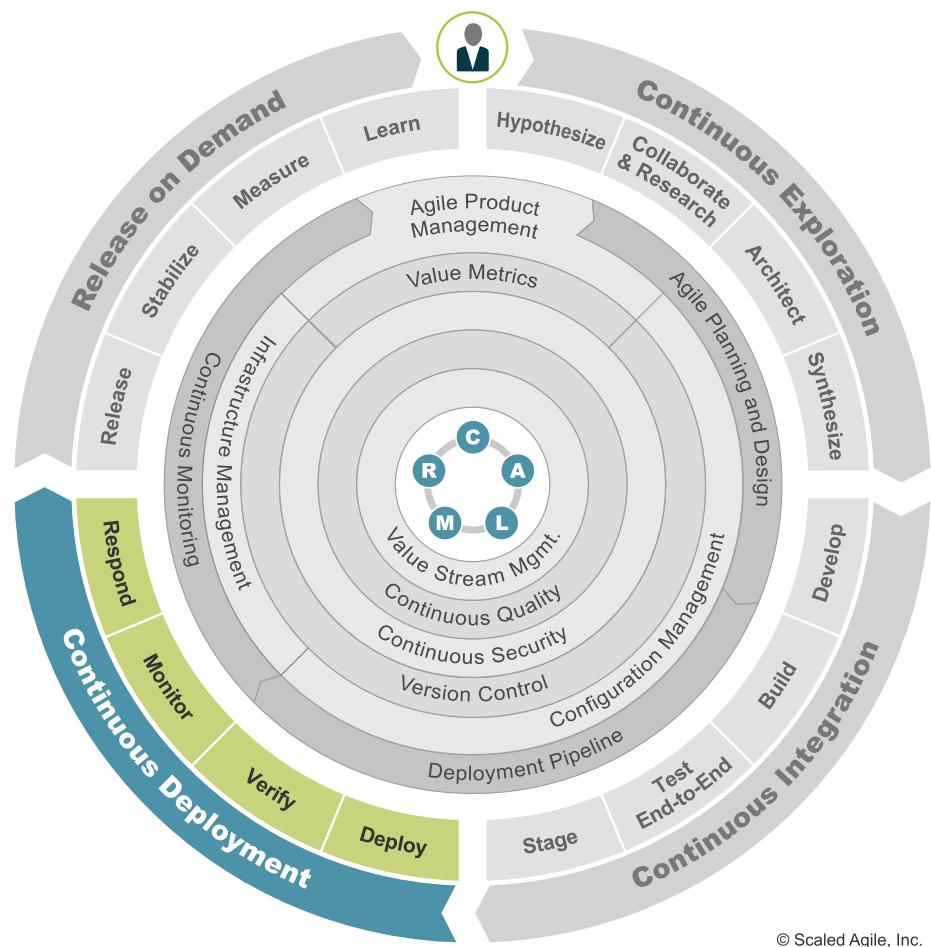


Figure 3. DevOps enables continuous deployment

For instance, *deploying* solutions in the CDP involves using tools that automate the provisioning of production infrastructure, deploy solution binaries to select targets, verify production functionality, capture runtime telemetry, and proactively alert on issues. DevOps practices and tools streamline these capabilities, allowing solutions to be deployed and fully prepared for on-demand release in minutes.

All four continuous deployment activities are enabled by DevOps, though with different combinations of technical practices and tooling. See the [DevOps](#) article series for more guidance on DevOps and how it facilitates the CDP.

---

## Learn More

- [1] <https://netflix.github.io/chaosmonkey/>
- [2] Kim, Gene, et al. *The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win*. IT Revolution Press, 2013.
- [3] Kim, Gene, Jez Humble, Patrick Debois, and John Willis. *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations*. IT Revolution Press,  
2016
- .
- [4] Humble, Jez, and David Farley. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Addison-Wesley, 2010.
- [5] Gregory, Janet, and Lisa Crispin. *More Agile Testing: Learning Journeys for the Whole Team*. Addison-Wesley Signature Series (Cohn). Pearson Education, 2014.
- [6] State of DevOps Report. <https://puppet.com/resources/whitepaper/state-of-devops-report>

Last update: 9 January 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm

Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

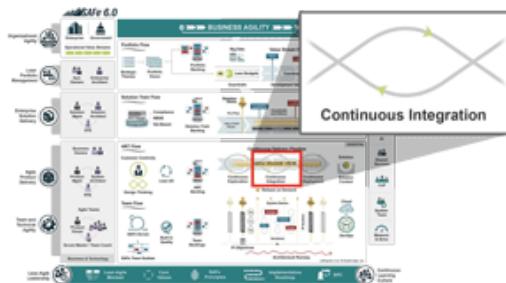
[Remote Training Policy](#)





+ Framework

ENGLISH (US) ▾



“ The epiphany of integration points is that they control product development. They are the leverage points to improve the system. When timing of integration points slip, the project is in trouble.

—Dantar Oosterwal, *The Lean Machine*

## Continuous Integration

Continuous Integration (CI) is an aspect of the Continuous Delivery Pipeline in which new functionality is developed, tested, integrated, and validated in preparation for deployment and release.

CI is the second aspect in the four-part [Continuous Delivery Pipeline](#) of Continuous Exploration (CE), Continuous Integration (CI), [Continuous Deployment](#) (CD), and [Release on Demand](#) (Figure 1).



Figure 1. Continuous integration in the context of the continuous delivery pipeline.

## Details

Continuous integration is a critical technical practice for each Agile Release Train (ART). It improves quality, reduces risk, and establishes a fast, reliable, and sustainable development pace.

With continuous integration, the system always runs, meaning it's potentially deployable, even during development. CI is most easily applied to software solutions where small, tested vertical threads can deliver value independently. In larger, multi-platform software systems, the challenge is harder. Each platform has technical constructs which need continuous integration to validate new functionality. CI is even more complicated when systems comprise software, hardware, components, and services provided by suppliers. But the fact remains that frequently integrating and testing features together is the only practical way to validate a solution fully.

As a result, teams need a balanced approach that allows them to build-in quality and gets fast feedback on their integrated work. For purely software-based solutions, continuous integration is relatively easy to achieve with modern tools. For more complex systems with hardware and software, a *continuous integration* approach is required (see the [Enterprise Solution Delivery](#) article) to balance the economic trade-offs between frequency, the scope of integration, and testing.

## The Four Activities of Continuous Integration

As illustrated in Figure 2, SAFe describes four activities associated with continuous integration:

1. **Develop** describes the practices necessary to implement stories and commit the code and components to version control
2. **Build** describes the techniques needed to create deployable binaries and merge development branches into the trunk
3. **Test end-to-end** describes the practices necessary to validate the solution

4. **Stage** describes the steps required to host and validate solutions in a staging environment before production

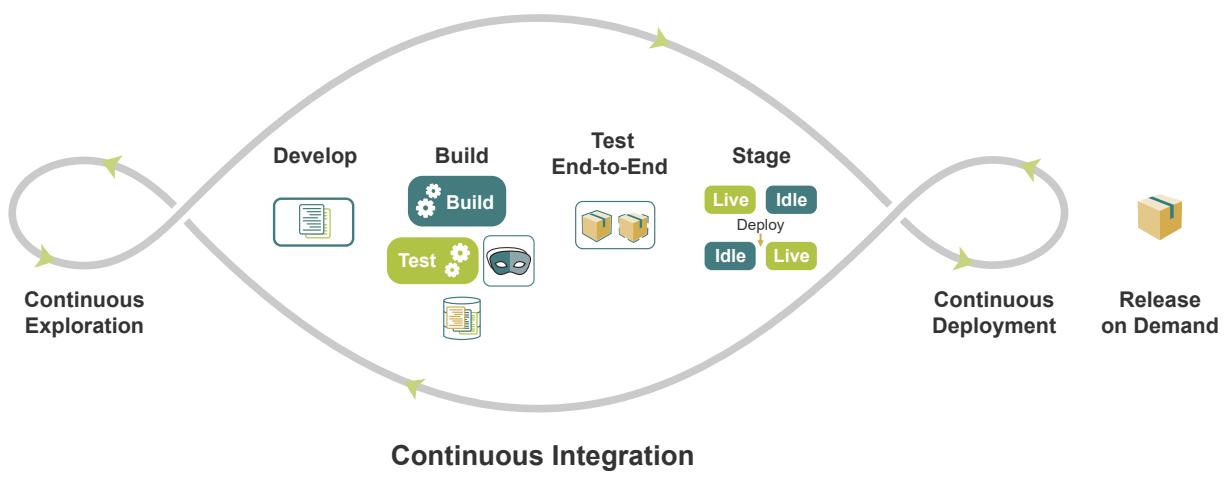


Figure 2. Four activities of continuous integration

© Scaled Agile, Inc.

## Develop

Developing the solution refers to implementing stories by refining features from the [ART Backlog](#) as may be needed and then coding, testing, and committing the work product into the source control system. Testing in this activity tends to focus on unit and story-level testing and often requires test doubles (see [Test-Driven Development](#)) to replicate other components or subsystems that are not readily available or easily tested.

Several practices are associated with developing the solution:

- **Break features into stories** – This enables continuous delivery via small batches and smooth integration, including creating user story maps to ensure that workflows meet customer needs.
- **Behavior-Driven Development (BDD)** – BDD is a process Product Owners and teams use to understand requirements better and improve quality by creating acceptance criteria and automating tests, often before the code is written. BDD works with TDD and is described [here](#).
- **Test-Driven Development (TDD)** – TDD involves writing the unit test first, then building the minimal code needed to pass the test. This testing technique leads to better design, higher quality, and increased productivity. TDD works with BDD and is described further [here](#).
- **Version control** – Effective version control allows teams to recover quickly from problems and improve quality, ensuring the integration of the right components. Aggregating assets under version control is a leading indicator of continuous integration maturity.

- **Built-in quality** – [Built-In Quality](#) prescribes practices around flow, architecture & design quality, code quality, system quality, and release quality.
- **Application telemetry** – Application telemetry is the primary mechanism that acquires and then uses application data to help determine the results of relevant hypotheses.
- **Threat modeling** – In addition to threat modeling done in continuous exploration (architect step), the system design should identify possible vulnerabilities that teams may introduce with new functionality.

## Build

During the build phase, teams continuously integrate new code. Automating the build and test tools to run upon code commit is one of the best ways to integrate. Passing versus not-yet-passing and broken automated tests are objective indicators of progress. Automating code building enables teams to fix problems quickly *before they affect more significant parts of the system*. Addressing a broken build should be the highest priority. A ‘gated commit’ ensures software has passed the gate (e.g., unit tested, performance-tested, free of known defects, and so on) before being checked into the main codebase or *trunk*. Code that passes the tests is automatically integrated, which removes the complications of managing multiple source code branches. *Trunk-based development* helps ensure code can be released on demand reliably without costly code freezes.

Five practices can help build a high-quality solution:

1. **Continuous code integration** – Code commit should automatically trigger the compilation and testing of changes. Ideally, this happens on each commit and should happen several times daily.
2. **Build and test automation** – The automated compilation process includes unit- and story-level tests to verify changes, often requiring *test doubles* to enable fast builds and replicate other systems.
3. **Trunk-based development** – Teams should integrate code quickly, at least once daily, or ideally upon commit, and all teams should work off a single trunk, avoiding long-lived branches.
4. **Gated commit** – Committing to the main trunk is risky, as broken changes can impact many teams. Therefore, only the modifications validated through the build and test process merge into this branch.
5. **Application security** – Code analysis tools inspect the code and third-party packages for known vulnerabilities.

## Test the solution end-to-end

While critical, automated local story and component testing aren’t enough. System-level integration and testing are required to test features thoroughly. Figure 3 illustrates how the

**System Team** helps integrate the work of all teams on the ART frequently, providing some objective evidence of progress.

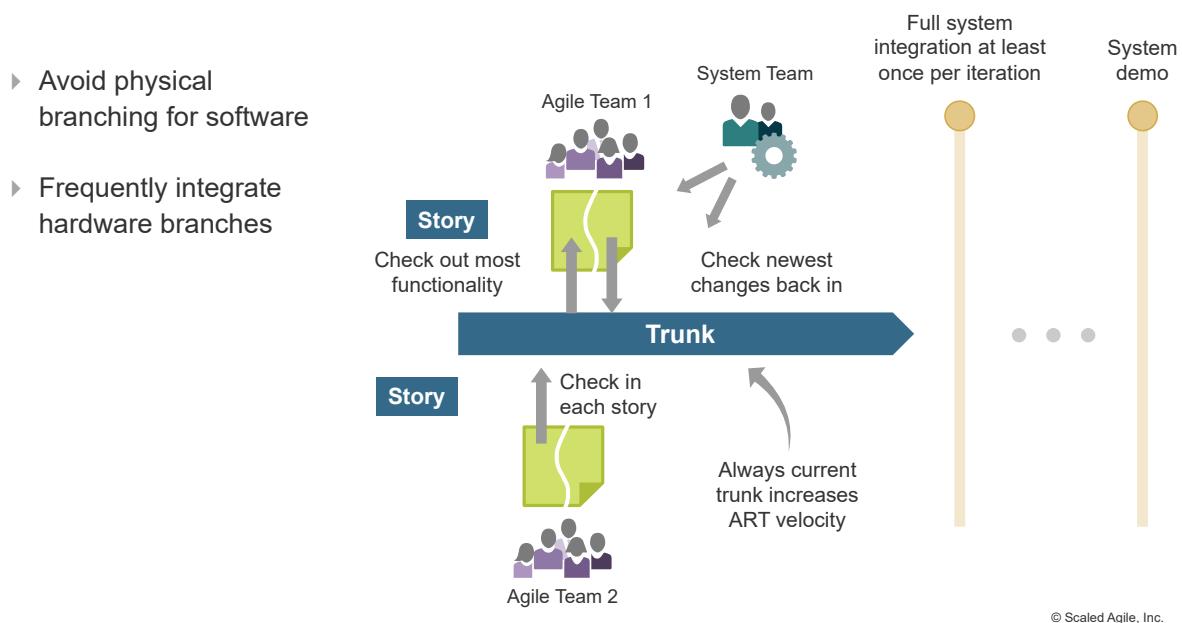


Figure 3. Integrating the work of all teams on the ART

System-level testing frequently happens during the iteration, ideally after every commit. However, whatever the circumstances, such full-system integration must be accomplished at least once per iteration. Otherwise, the late discovery of defects and issues from earlier iterations causes substantial rework and delays.

Six practices can help improve end-to-end system testing:

1. **Test and production environment congruity** – Environment congruity assures that testing exercises the solution as it would behave in front of live users and decreases the probability that defects will escape into production.
2. **Test automation** – Automated testing should include a variety of tests, such as functional, integration, regression, and more. The [Agile Testing](#) article details a testing matrix of what can and should be automated.
3. **Test data management** – To create stability, tests must be consistent and realistic, replicating production as much as possible and under source control.
4. **Service virtualization** – Different kinds of testing require different environments. Service virtualizations allow teams to simulate a production environment without the costs and effort associated with creating and managing physical infrastructure.
5. **Testing nonfunctional requirements (NFRs)** – System attributes, such as security, reliability, performance, scalability, and usability, are critical and require testing.
6. **Continuous integration with suppliers** – [Suppliers](#) bring unique contributions that reduce lead time and improve value delivery, requiring continuous integration. It helps to adopt a

shared integration cadence and establish objective evaluation milestones.

## Stage

Finally, the ART must validate the entire solution in staging based on the following practices:

- **Maintain a staging environment** – A staging environment closely resembling production provides the place for such validation.
- **Blue/Green deployment** – The blue/green deployment pattern involves two environments—live (production) and idle (staging). Changes flow continuously to idle, stage, and ready them for production deployment. When ready, a configuration change switches the two environments. Idle becomes live, while the old live becomes the new idle. This approach enables continuous delivery, zero-downtime deployment, and fast failure recovery.
- **System demo** – This is where stakeholders evaluate a solution's readiness for production deployment.

## Enabling a Culture of Continuous Integration

Continuously integrating large and complex systems is a time-consuming journey. The following section provides some suggestions for building a thriving CI culture and practice.

- **Integrate frequently** – The more often teams integrate, the quicker they find problems. And the harder it is to do, the more often they need to do it. This practice eliminates impediments and adds automation along the way, resulting in faster learning cycles and less rework.
- **Make integration results visible** – When the integration process breaks, everybody should know why it failed. When it's fixed, creating new tests should detect the problem earlier and prevent it from happening again.
- **Fixing failed integrations is a top priority** – Teams that continue working through integration failures fall short of the values and culture associated with building production-ready systems. Teams often use flashing lights or other notifications, drawing attention to a broken build and establishing visible indicators displaying the percentage of the time the system remains broken.
- **Establish a shared cadence** – Integration points are more accessible when all teams follow the same consistent rhythm. Suppose teams cannot do a complete integration within an iteration. In that case, they can make near-term trade-offs on what's possible while continuously improving their techniques and infrastructure toward this goal.
- **Develop and maintain proper infrastructure** – Effective continuous integration depends on the availability of test and staging environments. Infrastructure is, of course, an investment. But [Lean-Agile Leaders](#) take the long view and make the investments necessary today to increase velocity for the marathon ahead.

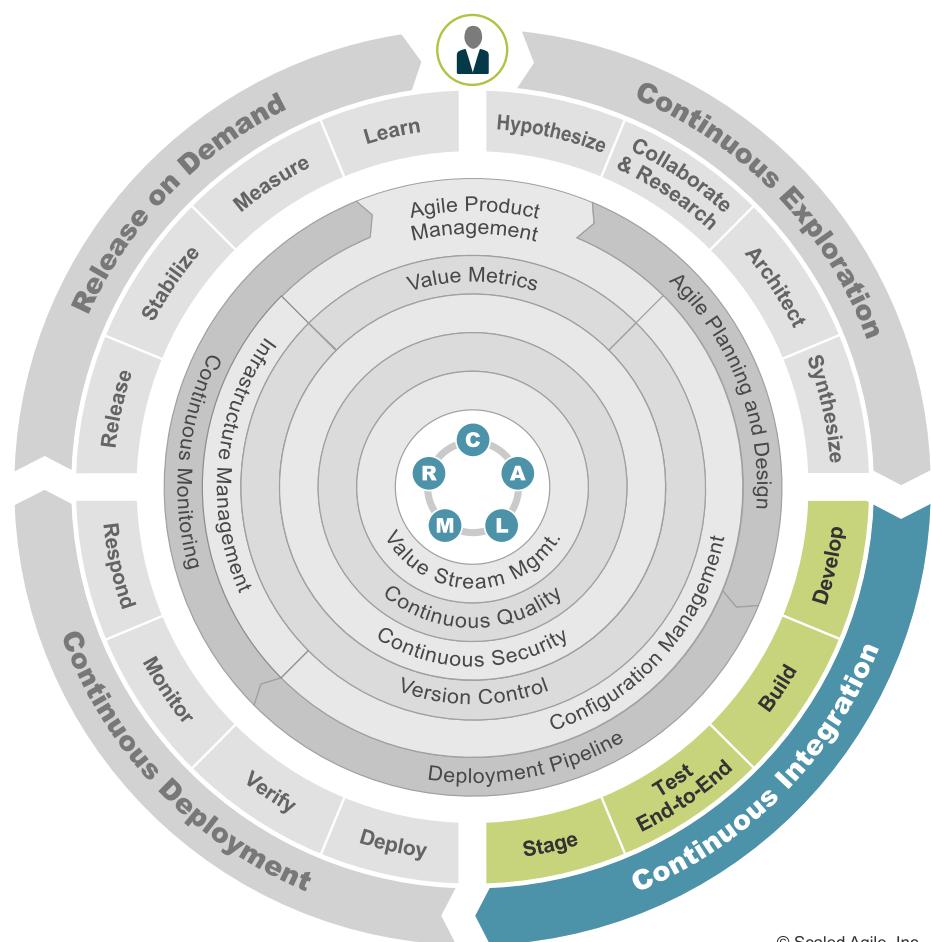
- **Apply supportive software engineering practices** – Continuous integration is more accessible when designing systems with those concerns in mind. **Test-first** development and planning for testability call for more modular solutions and separation of concerns, as well as using primary interfaces and physical test points.

Another important aspect of CI culture is ensuring a fast flow of value through the pipeline. See the [ART Flow](#) article for more information on making value flow without interruption ([Principle #6](#)).

## Enabling Continuous Integration with DevOps

Continuous integration involves crucial ‘development’ activities that originally inspired the ‘Dev’ in DevOps. These activities focus on solution development and pipeline flow through pre-production environments. Applying DevOps thinking, practices, and tooling in this segment of the value stream enables rapid development, frequent code integration, and built-in quality and compliance.

As illustrated in Figure 4, SAFe’s CALMR approach to DevOps (center) enables continuous integration and several practice domains (inner rings). Each of the four activities (in green) is a collaborative effort that draws upon DevOps expertise from multiple disciplines to maximize delivery speed and quality.



#### Figure 4. DevOps enables continuous integration

For example, *building* solutions in the continuous delivery pipeline crosses multiple DevOps domains. Checking code into version control triggers the deployment pipeline to invoke automated merge, quality, and security checks, then apply configurations stored as code to build shippable, full-stack binaries. Using DevOps, this process typically turns source code into tested, deployable solutions quickly.

All four continuous integration activities are enabled by DevOps, though with different combinations of technical practices and tooling. See the DevOps article series for more detailed guidance on DevOps and how it allows the continuous delivery pipeline.

---

## Learn More

[1] Oosterwal, Dantar P. *The Lean Machine: How Harley-Davidson Drove Top-Line Growth and Profitability with Revolutionary Lean Product Development*. AMACOM, 2010.

[2] Leffingwell, Dean. *Scaling Software Agility: Best Practices for Large Enterprises (Agile Software Development Series)*. Pearson Education, 2007.

[3] Kim, Gene, Jez Humble, Patrick Debois, and John Willis. *The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations*. IT Revolution Press, 2016

Last update: 06 January 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## **Training**

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## **Content & Trademarks**

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## **Scaled Agile, Inc**

### **Contact Us**

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### **Business Hours**

Weekdays: 9am to 5pm

Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

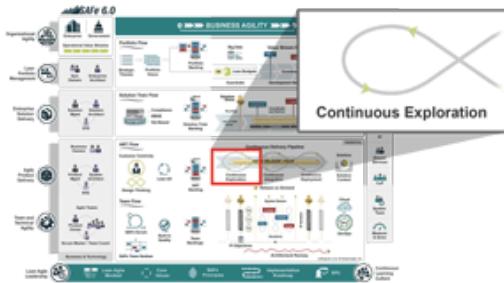
[Remote Training Policy](#)





## + Framework

ENGLISH (US)



“ Specifically, you can take the time to develop and bring to the table an outside-in, market-centric perspective that is so compelling and so well informed that it can counterbalance the inside-out company-centric orientation of last year’s operating plan.

—Geoffrey Moore, *Escape Velocity*

## Continuous Exploration

Continuous Exploration (CE) is an aspect of the Continuous Delivery Pipeline that drives innovation and fosters alignment on what should be built by continually exploring the market and customer needs, defining a vision, roadmap, and set of features for a solution.

*Continuous Exploration (CE)* is the first aspect of the four-part [Continuous Delivery Pipeline \(CDP\)](#), which also includes [Continuous Integration \(CI\)](#), [Continuous Deployment](#), and [Release on Demand](#) (Figure 1).

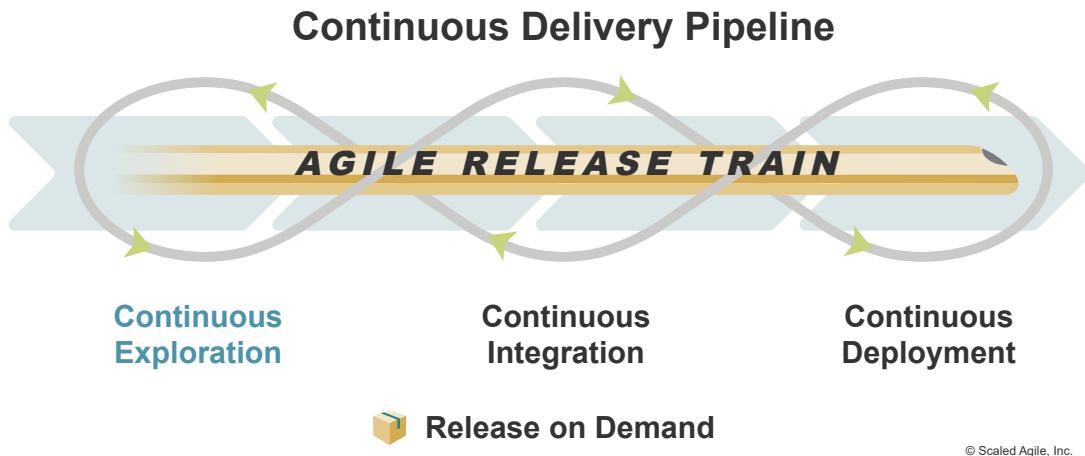


Figure 1. Continuous exploration in the context of the CDP

During CE, new ideas are raised, refined, and prepared as a list of prioritized [Features](#) in the [ART Backlog](#). Agile Teams implement the features prioritized by [Product Management](#) during [PI Planning](#), which kicks off the CI process. After that, the CD cycle pulls them into production, where they are validated and prepared for release.

## Details

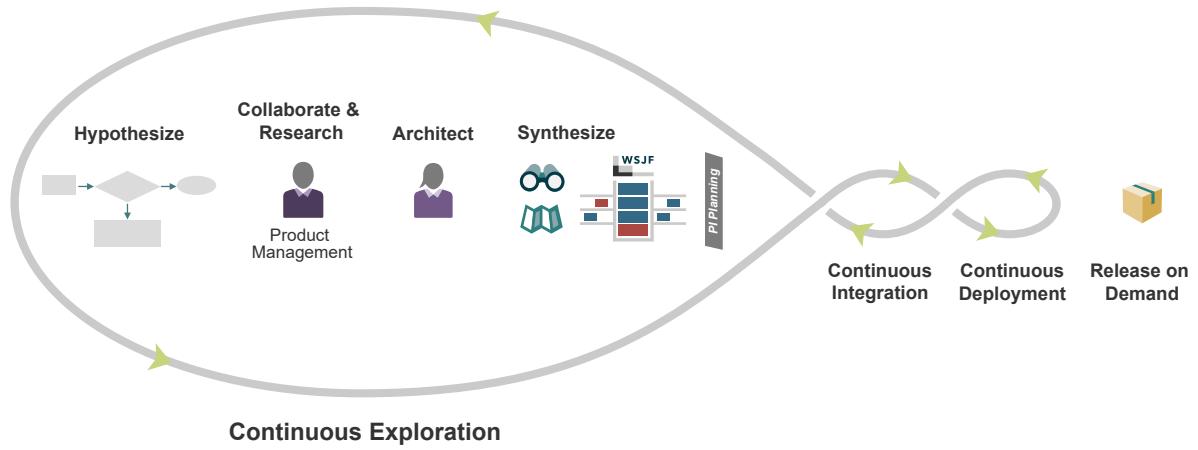
[Agile Product Delivery](#) is one of the seven core competencies of SAFe. It allows the enterprise to deliver increasingly valuable solutions to end users with optimal frequency. CE is integral to that process and focuses on applying [Customer Centricity](#) and [Design Thinking](#) to understand and create alignment on new development opportunities while recognizing that all such ideas are hypotheses that need to be validated.

CE replaces traditional waterfall approaches of up-front, rigid requirement definitions with a process that generates a consistent flow of [Features](#) ready for implementation in the ART Backlog. Decomposing features into small batches of [Stories](#) enables work to move quickly through the remaining aspects of the CDP to the [Customer](#). Getting fast feedback is built into the process, allowing the teams to adjust to market needs. See the [ART Flow](#) article for more information on making value flow without interruption ([Principle #6 – Make value flow without interruptions](#)).

Customers, [Suppliers](#), partners, [Business Owners](#), [Agile Teams](#), [Product Owners](#), and [Lean Portfolio Management](#) are among the internal and external stakeholders involved in this process. Their involvement may be indirect, such as through secondary research on market needs. Or it can be direct, as when Agile Teams participate in an [Innovation and Planning Iteration](#). CE activities enable the organization to align to a shared [Vision](#), a set of features in the backlog defined for implementation, and a forecasted [Roadmap](#).

## The Four Activities of Continuous Exploration

Figure 2 illustrates the four steps of continuous exploration, described in the following sections.



**Figure 2. Four activities of continuous exploration**

© Scaled Agile, Inc.

## Hypothesize

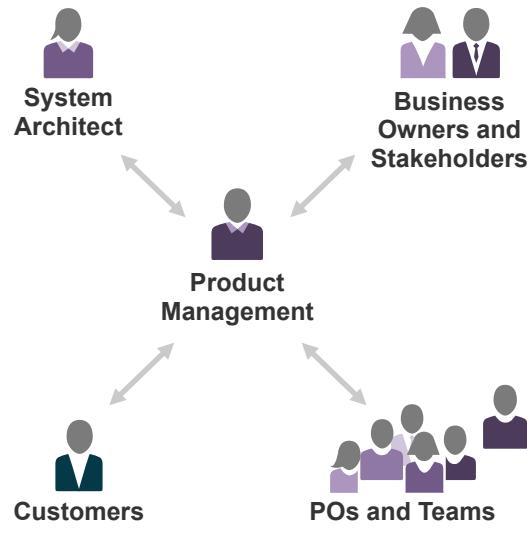
Hypothesize describes the practices for generating ideas and the measurements needed to validate them with [Customers](#). Its primary purpose is to define a Solution hypothesis that teams will validate through the CDP.

Product Management has notions of Customer needs based on their understanding of the marketplace, [Strategic Themes](#), [Portfolio Vision](#), and Roadmap. However, these ideas should not be considered facts. Instead, teams should consider them a hypothesis that needs to be tested and proven. Accordingly, the practices associated with hypothesis-driven development include:

- **Lean startup thinking** – Defining *Minimum Marketable Features (MMFs)* and *Minimum Viable Products (MVPs)* [1] helps evaluate hypotheses quickly with minimal investment. MMFs and MVPs represent the smallest amount of usable functionality for early customers, who can provide feedback for future product development.
- **Innovation accounting** – Evaluating hypotheses for a new product or feature requires a different approach than measuring existing solutions. It requires us to consider two questions: 1) Are we progressing toward our outcome hypothesis? 2) How do we know? [Innovation accounting](#) uses actionable metrics (leading indicators) for determining early results and is a good predictor of future business outcomes. Leading indicators answer these two questions and improve economic decisions during initial solution development and evaluation of the MMF or MVP.

## Collaborate and Research

Creating a compelling and differentiated vision requires Product Management to facilitate a continuous and *collaborative* process, soliciting input from diverse stakeholders, as illustrated in Figure 3.



© Scaled Agile, Inc.

Figure 3. Product Management collaborates with multiple stakeholders to refine requirements

- **System Architects** – [System Architects](#) have in-depth technical knowledge of solutions. They are responsible for understanding them at the system level and their use cases and [Nonfunctional Requirements \(NFRs\)](#). Although it's natural to view these roles as technically and internally inclined, architects should also have significant and ongoing customer engagement that enables them to identify new ways to solve unmet needs.
- **Customers** – Customers judge value by voting with their wallets or feet. Accordingly, they're the primary source of feedback on the solution and how well it meets their needs. *But a note of caution:* customer motivations are often heavily bound to their current solution context, so they are often motivated only to improve things incrementally. In other words, customer *feedback alone does not constitute a product strategy*. But failing to meet current and evolving customer needs is a sure path to failure.
- **Business Owners and stakeholders** – Business Owners have the business and market knowledge needed to set the mission and vision. A solution that doesn't meet their expectations likely has no value.
- **POs and teams** – Product Owners and Agile Teams create domain expertise through their work creating the solution. In many cases, they are closest to both technical and user concerns. Their input is integral to the ongoing evolution of the solution.

Collaboration and research are grounded in specific practices:

- **Primary market research** – Product Management develops additional insights through primary market research, including surveys, focus groups, questionnaires, and competitive analysis for customer understanding.
- **Customer visits and Gemba walks** – A Gemba walk [2] or customer visit is a process where the product team observes how stakeholders execute the specific activities in their operational value streams to identify opportunities for relentless improvement. There's no substitute for first-person observation of the daily activities of the people doing the work.

Whether structured or informal, Product Managers and Product Owners need to understand how people use systems in their work environments. They can't do that at their desk, so there is no substitute for "getting outside the building," visiting customers, and observing users in their specific [Solution Context](#).

- **Secondary market research** – To broaden their thinking, Product Management uses various secondary market research techniques to understand the customers and markets they're serving comprehensively. Staying abreast of market/industry trends is a critical outcome of secondary market research.
- **Lean UX thinking** – [Lean UX](#) [3] is a collaborative process of working with stakeholders to define Minimum Marketable Features (MMFs) and validate them quickly with customers.

Collaborative research enables the organization to refine its processes further and create artifacts that clearly express its emerging understanding of the problem space. These include:

- **Developing personas to focus design** – Informed by research, personas help the organization understand their target customers
- **Building empathy for the user** – Empathy maps ensure that the team considers the user's needs and how they may evolve through successive releases
- **Designing the customer experience** – Customer journey maps provide the design link between the operational value stream and the Customer's user experience

While these artifacts tend to be relatively stable over successive releases, the entire enterprise must find ways to avoid making strategic decisions on stale insights.

## Architect

With a clear understanding of the problem, CE moves into the solution space, defining the minimum amount of architecture that will support the Solution and enable continuous delivery.

Architects serve the business and the Customer by ensuring the [Architectural Runway](#) is sufficient to deliver the required functionality and is designed to enable the [Continuous Delivery Pipeline \(CDP\)](#). System Architects support the CDP through five practices:

1. **Architecting for releasability** – Different parts of the solution require distinct release strategies. Therefore, design solutions to enable various incremental release strategies and evolve them over time based on business demand.
2. **Architecting for testability** – Systems designed and architected modularly enable continuous testing.
3. **Separating deployment and release** – The capability to deploy continuously requires architectural enablers that allow functionality to be moved into production but hidden from Customers.

4. **Architecting for operations** – Build telemetry and logging capabilities into every application and solution to meet operational support needs. Allow services to be downgraded or even removed during high loads or in response to incidents. Build capabilities for fast recovery and fix-forward.
5. **Threat modeling** – Information security considerations should start early, identifying threats to proposed architecture, infrastructure, and applications. Capture essential security requirements as Nonfunctional Requirements to influence backlogs.

## Synthesize

Synthesize distills the knowledge gained into a new future state for the solution. The vision, roadmap, and prioritized backlog of features align the ART's teams to a shared direction. Focus synthesis on ensuring these assets are ready for PI planning. The following practices are needed to accomplish this:

- **Creating the solution vision** – The vision provides the reasons or purpose for developing new features.
- **Maintaining the solution roadmap** – The ART roadmap provides a view into the near future, helping Product Management prioritize the work, enabling System Architects to prioritize the architecture, and providing visibility for Business Owners.
- **Defining a backlog with clearly written items** – Defining features that fit in a PI is critical for ARTs to align on what is needed and for teams to plan. The backlog also reflects essential security requirements.
- **Behavior-driven development (BDD)** fosters collaboration between Product Management, Product Owners, and Agile Teams, which clarifies requirements by adding acceptance criteria.
- **Economic prioritization** – Prioritized features enable effective development. The budget guardrails of capacity allocation, investment horizons, and continuous Business Owners engagement are critical in prioritization.
- **PI Planning** – The exploration work done by the ART is an essential input to the following PI planning event and helps with alignment.

Where there's alignment on what needs to be built, features smoothly flow to the CI segment of the CDP. However, this does not mean that exploration is over. Feedback is continually flowing back from deployed and released features. This feedback informs new decisions about what the ART should work on next and is integral to the CE process.

## Enabling Continuous Exploration with DevOps

Activities in continuous exploration set the pace for the entire CDP. Execution is slow when they involve large batches, rigid specifications, and commitments to fixed plans. Thus, for ARTs to achieve continuous delivery, these 'upstream' activities should be driven by a bias for speed and

validated learning. Applying [DevOps](#) thinking, practices, and tooling early in the value stream reinforces *all* SAFe principles, aligns the entire ART to a DevOps mindset, and primes the CDP.

Many DevOps-related concepts apply at this level. Figure 4 illustrates SAFe's [CALMR](#) approach to DevOps (center) and practice domains (inner rings) support CE. Each of the four activities (in green) is a collaborative effort that draws upon DevOps expertise from multiple disciplines to maximize delivery speed and quality.

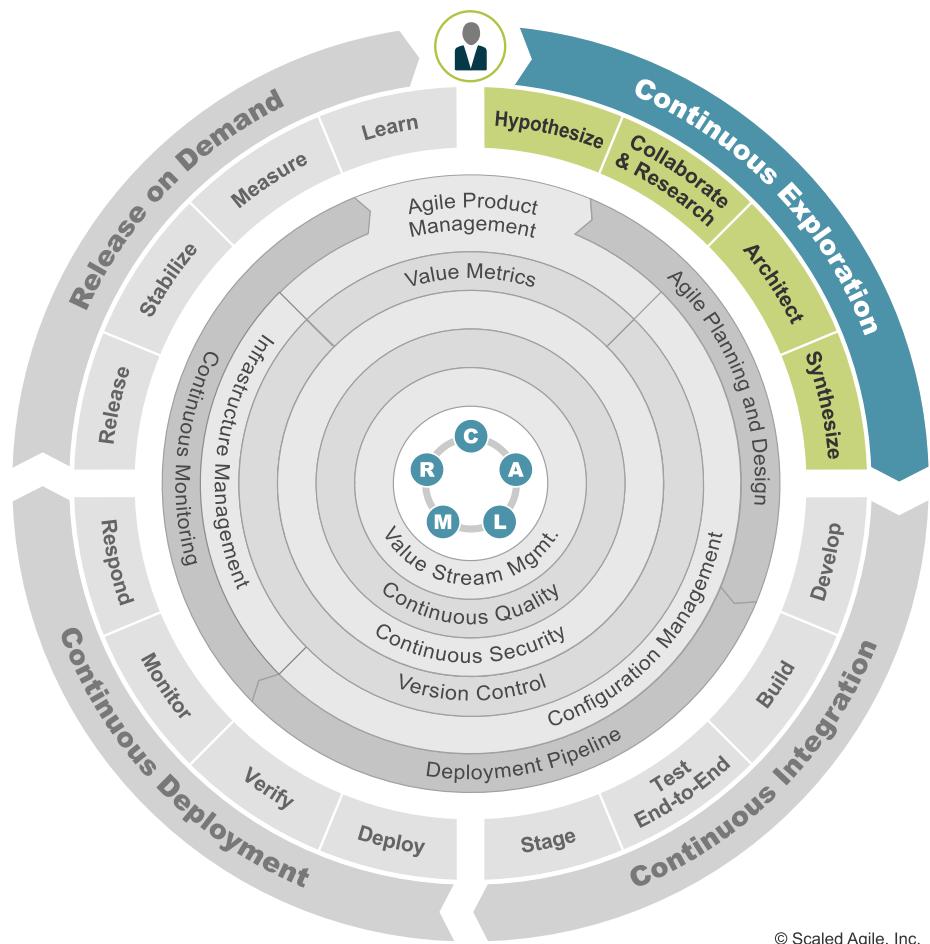


Figure 4. DevOps enables continuous exploration

For example, *architecting* for continuous delivery is not a one-dimensional activity. It crosses several disciplines, as Figure 4 suggests. Agile architecture must account for desired quality and security levels, align to value stream performance objectives, produce tangible configurations under version control, and generate backlog items and NFRs that support Agile planning and emergent design. Moreover, the CALMR mindset should guide all architectural decisions and actions to maximize delivery speed and solution value.

All four CE activities are enabled by DevOps, though with different combinations of technical practices and tooling.

## Learn More

[1] Ries, Eric. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Random House, Inc, 2011.

[2] Womack, Jim. *Gemba Walks Expanded 2nd Edition*. Lean Enterprise Institute, Inc, 2019.

[3] Gothelf, Jeff, and Josh Seiden. *Lean UX: Designing Great Products with Agile Teams*. O'Reilly Media, 2016.

Leffingwell, Dean. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise (Agile Software Development Series)*. Pearson Education, 2011.

Last update: 06 January 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

# Scaled Agile, Inc

## Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

## Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



## + Framework



“ Stay committed to your decisions but stay flexible in your approach.

—Tony Robbins

# Iteration Planning



**Note:** For more on SAFe Scrum, please read the additional Framework articles in the Scrum series, including [SAFe Scrum](#), [SAFe Scrum Master/Team Coach](#), [Iterations](#), [Iteration Goals](#), [Iteration Review](#), and [Iteration Retrospective](#)

Iteration planning is a SAFe Scrum event where all team members determine how much of the Team Backlog they can commit to delivering during an upcoming Iteration. The team summarizes this work as a set of committed iteration goals.

## Details

Iteration planning is the first event of the [Iteration](#). During planning, the team defines, organizes, and commits to the work for the next iteration. The iteration planning meeting is timeboxed to approximately 90 minutes for a two-week iteration. The team's backlog has been partially identified and planned during [PI Planning](#). In addition, the teams have feedback—not only from their prior iterations but also from the [System Demo](#), stakeholders, and others. All this context feeds into the iteration planning event to inform the plan for the upcoming iteration.

## Inputs and Outputs of Iteration Planning

Inputs to iteration planning include:

- A refined [Team Backlog](#), including [Stories](#) from the team's PI plan, which were identified during [PI Planning](#) and tentatively assigned to iterations, new stories that have been identified, and other items from the team's local context, including defects, refactors, maintenance, and technical debt
- The Team and ART [PI Objectives](#) created during PI planning
- Feedback from the [System Demos](#) and prior iterations, including any stories that did not meet the definition of done (DoD)

A successful iteration planning event delivers the following outputs:

- Stories planned for the upcoming iteration, including Enablers. Each has defined acceptance criteria and an estimate and is recorded in the iteration backlog.
- Committed iteration goals.
- Dependencies with other teams are understood and planned.

## Preparation

The teams prepare for the event as follows:

- **Backlog refinement.** Teams usually approach iteration planning with a pre-elaborated team backlog from the backlog refinement sessions held during the previous iteration.
- **Close out the previous iteration.** The team confirms the stories in the last iteration were completed and accepted. If any remain, they are moved to the team backlog and reprioritized.
- **Initial iteration goals.** The Product Owner (PO) may prepare some initial iteration goals based on the team's progress in the PI.

## Process

The PO typically starts the event by presenting high-priority stories from the team backlog and the initial iteration goals (if applicable). Many of these stories originated from PI planning, while others are from the team's local context. The acceptance criteria are then elaborated through conversation and collaboration with the Product Owner and other stakeholders. Next, they estimate the effort to complete each item using relative story points. Based on the estimates and business value, the Product Owner may change the ordering of the stories.

During planning, the team discusses implementation options, technical issues, [Nonfunctional Requirements \(NFRs\)](#), and dependencies. Next, the PO and team select the candidate stories based on their available capacity for the iteration. Some teams decompose stories into tasks (optional) and forecast them in hours to confirm they have the capacity and skills to complete them. Once planning is complete, the team synthesizes the work into iteration goals, commits to

the plan, and records the iteration backlog in a visible place, such as a story or Kanban board and Agile project management tooling.

## Attendees

Attendees of the iteration planning event include:

- The [Product Owner](#)
- [Scrum Master/Team Coach](#)
- All team members and any subject matter experts, as needed
- Any other stakeholders required, including representatives from other Agile Teams or trains

Scrum Masters/Team Coaches or POs typically facilitate iteration planning for the team, ensuring the participants stay within the agreed agenda and event timebox.

## Agenda

An example agenda for iteration planning follows (Figure 1), including a description of each item.



Figure 1. Example iteration planning agenda

During planning, the PO defines the 'what,' the team determines the 'how,' and 'how much,' as follows:

1. **Establish capacity** – The team calculates its capacity for the upcoming iteration using adjusted historical velocity.

2. **Story analysis and estimation** – In conversation with the PO, the team selects and estimates the most valuable stories to meet their PI Objectives, addressing local concerns and dependencies, where applicable.
3. **Tasking stories (optional)** – Some teams task stories to understand their capacity and capabilities better. They determine the best person to accomplish the work, estimate the effort (typically in hours), and identify dependencies with other tasks or stories. Planning stops once the team runs out of capacity.
4. **Develop iteration Goals** – This process repeats until when the team is out of capacity. Next, the teams summarize the plan as a set of iteration goals. (Note: Some teams work the other way around; they start with iteration goals and then work on capacity, story analysis, and estimating to support those goals.)
5. **Commit to iteration goals** – At the end of planning, the Product Owner and team agree on the final list of stories that will be selected, and they revisit and restate the iteration goals. Everyone commits to the iteration goals, and the scope of the work remains largely fixed for the duration of the iteration.

## Commitment to Iteration Goals

Iteration goals provide clarity, commitment, and information. The commitment to the iteration goals is reciprocal (Figure 2) and serves the following purposes:

- Aligns team members to a set of shared objectives for the iteration
- Focuses teams on meeting their PI objectives and managing dependencies with other teams
- Provides transparency and management information as needed

### Commitment to iteration goals are reciprocal:

- Teams agree to do everything they can to meet their commitment and if that's no longer feasible, they raise the concern immediately
- Business stakeholders commit to leaving priorities unchanged during the iteration

#### Commitment

Too much holding to a commitment can lead to burnout, inflexibility, and quality problems.



#### Adaptability

Too little commitment can lead to unpredictability and lack of focus on results.

Figure 2. Guidelines for team commitments

# Estimating Stories and Forecasting Team Capacity

## Relative Estimation of Stories for Planning

Agile Teams use story points to estimate stories relative to each other [2, 3]. The size (effort) for each item is compared to other stories. For example, an eight-point story should be four times the effort of a two-point story. (Note: Refer to the [Story](#) article to learn how to write and estimate stories, including practices for whole team estimation and how to split large stories so they can be completed within an iteration.)

## Forecasting Team Capacity

The team forecasts its *capacity* for an upcoming iteration by using its historical velocity as a starting point. The team's average velocity (completed story points per iteration) becomes more reliable and predictable as they work together. Predictable velocity helps with planning and limits Work in Process (WIP).

## Starting Capacity for New Teams

When the team is new and the average velocity is *unknown*, one method for initially forecasting the team's capacity is as follows:

1. Give the team 8 points for every full-time developer (including test, etc)
2. Subtract one point for every team member's vacation day, holiday, or other non-working days in the iteration.

Figure 3 provides an example of forecasting capacity for a new seven-person team.

### Starting Baseline for Capacity



A team has three developers, two testers, a Scrum Master and a Product Owner, with no scheduled time-off for this iteration.

Exclude Scrum Masters (or Team Coaches) and Product Owners from the calculation.

**Estimated starting capacity =**  
 **$5 * 8 \text{ pts} = 40 \text{ pts/iteration}$**

© Scaled Agile, Inc.

Figure 3. Example starting capacity for a new team

# Creating a Shared Basis for Story Point Estimation

Story points can be aligned (approximately normalized) by teams in the ART, providing a *shared basis* for capacity forecasting and economic decision-making.

One approach is as follows:

1. Each team finds a small story that would take a day to develop, test, and validate. Call it a 'one.'
2. Estimate every other story relative to that 'one.'

Moreover, aligning story points enables reasonable estimate costs for features and epics requiring multiple teams' support. While many teams will increase their velocity over time—and that's a good thing—in reality, the number tends to remain stable.

---

## Learn More

[1] Knaster, Richard, and Dean Leffingwell. *SAFe 5.0 Distilled: Achieving Business Agility with the Scaled Agile Framework*. Addison-Wesley, 2020.

[2] Leffingwell, Dean. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*. Addison-Wesley, 2011.

[3] Cohn, Mike. *Agile Estimating and Planning*. Robert C. Martin Series. Prentice-Hall, 2005.

Last Update: 14 March 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## **Training**

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## **Content & Trademarks**

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## **Scaled Agile, Inc**

### **Contact Us**

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### **Business Hours**

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)





+ Framework

 ENGLISH (US)

[Home](#) » PI Planning

## PI Planning

“ Future product development tasks can't be predetermined. Distribute planning and control to those who can understand and react to the end results.

—Michael Kennedy, Product Development for the Lean Enterprise<sup>1</sup>

**Definition:** PI Planning is a cadence-based event for the entire ART that aligns teams and stakeholders to a shared mission and vision.

## Summary

PI Planning is an event that ensures all the teams on an ART, stakeholders, and leaders are aligned to a shared mission and vision. It is typically a 2-day event for the entire ART that takes place every 8-12 weeks. During this event, all the teams on the Agile Release Train work collaboratively to create a plan to deliver the highest value work in the upcoming PI and commit to a set of PI Objectives. PI Planning is led by the Release Train Engineer (RTE). The event takes place in the Innovation and Planning Iteration, which provides time and space for planning without impacting delivery.

## What is PI Planning?

PI Planning is a cadenced event used for an Agile Release Train (ART) to align Agile Teams and ART leadership to a shared mission, vision, and committed plan. PI planning is essential in SAFe.

PI planning delivers many business benefits, including:

- Establishing face-to-face communication among all Agile Team members and stakeholders

- Aligning development to business goals with the business context, vision, and Team and ART PI objectives
- Identifying dependencies and fostering cross-team and cross-ART collaboration
- Providing the opportunity for just the right amount of architecture and Lean User Experience (UX) guidance
- Matching demand to capacity and eliminating excess Work in Process (WIP)
- Making fast decisions
- Creating a wholistic, transparent view of where and when value will be delivered

Where possible, everyone is physically together. It may not always be practical for the entire Agile Release Train (ART) to collocate. While physical face-to-face planning has benefits, the focus is that the people who do the work plan the work. Real-time, virtual, face-to-face planning has also proven effective when physical presence is not possible as long as all members of the ART are participating.

## How does an ART prepare for PI Planning?

PI planning is a significant event that requires preparation, coordination, and communication. It is facilitated by the RTE. Event attendees include Business Owners, Product Management, all Agile Teams, System and Solution Architects, and other stakeholders. The RTE must schedule PI planning events far enough in advance to ensure attendance. The active participation of Business Owners in this event provides an essential guardrail on budgetary spending.

Inputs to PI planning include:

- Business context
- Roadmap and vision
- Highest priority **Features** of the ART Backlog

A successful PI planning event delivers two primary outputs:

- **Committed PI objectives** – Each team creates a set of PI objectives with a business value assigned by the Business Owners.
- **ART planning board** – Identifying new feature delivery dates, feature dependencies among teams, and relevant milestones

For the event to be successful, preparation is required in three major areas:

- **Organizational readiness**
- **Content readiness**
- **Logistics readiness**

The following sections describe these three areas.

## Organizational readiness

Before PI planning, there must be strategy alignment among participants, stakeholders, and Business Owners. Critical roles are assigned. To address this in advance, however, event organizers must consider the following:

- **Planning scope and context** – Is the planning process's scope (product, system, technology domain) understood? Do we know which teams need to plan together?
- **Business alignment** – Is there reasonable agreement on priorities among the Business Owners?
- **Agile teams** – Do we have Agile teams? Are there dedicated team members and an identified Scrum Master/Team Coach and Product Owner for each team?

## Content readiness

It's equally important to have a clear vision and context so that the right stakeholders can participate. Therefore, the PI planning must include the following:

- **Executive briefing** – A briefing that defines the current business context
- **Product vision briefing(s)** – Briefings prepared by Product Management, including the top 10 features in the ART Backlog

- **Architecture vision briefing** – A presentation made to communicate new enablers, features, current and future state architecture, and Nonfunctional Requirements (NFRs) related to the coming PI

## Logistics readiness

Preparing an event to support a large number of attendees isn't trivial. This prep can include securing and preparing the space for physically collocated planning. For remote attendees or a fully distributed PI Planning, this also includes investment in the necessary technical infrastructure. Considerations include:

- **Locations** – Each location where planning takes place needs preparation in advance.
- **Technology and tooling** – Real-time access to information and tooling to support distributed planning or remote attendees
- **Communication channels** – Primary and secondary audio, video, and presentation channels must be available

This article focuses on the planning activities of a single ART. However, large value streams may contain multiple ARTs and suppliers. In this case, multiple PI Planning events may need coordination. Further information on how to coordinate at this scale is found in Pre-Planning and Coordinate and Deliver guidance articles.

Read more about the ART Backlog and Vision:

[ART Backlog](#)

[Vision](#)

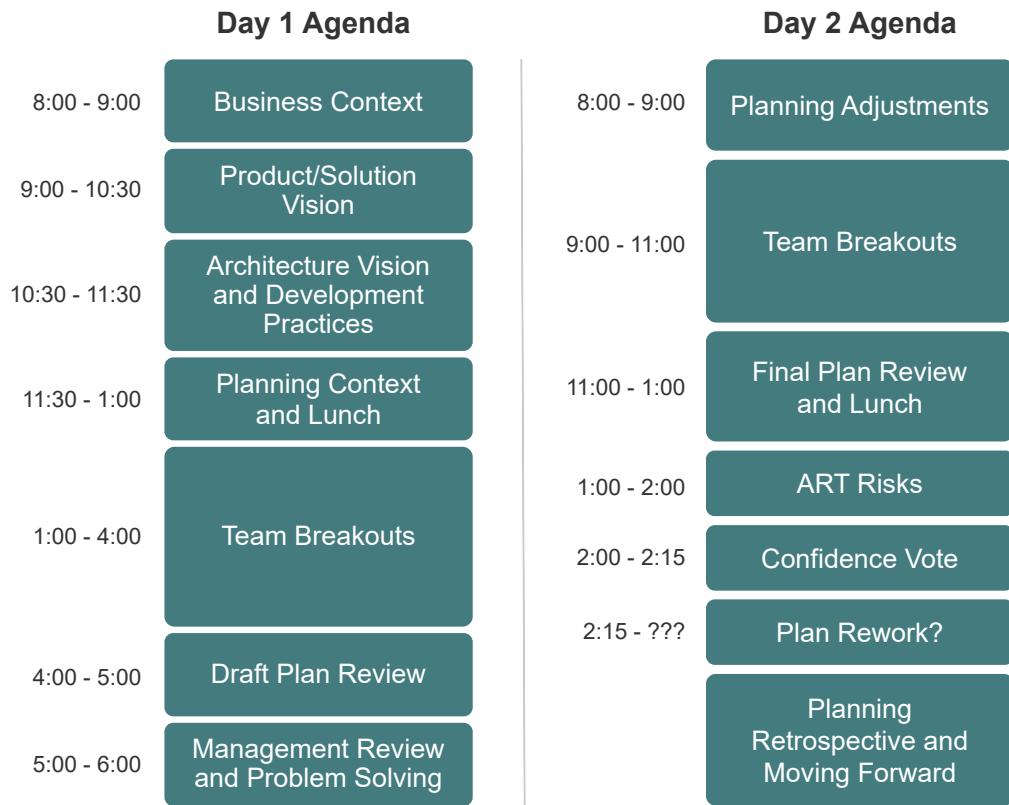
## How to run a PI Planning event?

PI Planning has a standard agenda that begins with a presentation of the business context and vision. This is followed by team breakouts—where the teams create high-level iteration plans and committed objectives for the upcoming PI. Facilitated by the Release Train Engineer (RTE), this event includes all members of the ART and occurs within the Innovation and Planning (IP) Iteration.

PI Planning occurs in a special iteration timebox called the IP Iteration. This avoids affecting the capacity of other iterations the Agile Teams are delivering within. PI Planning takes two days,

although the ART can extend this timebox to accommodate planning across multiple time zones.

The event follows an agenda similar to Figure 1. Descriptions of each item follow.



© Scaled Agile, Inc.

Figure 1. Standard two-day PI planning agenda

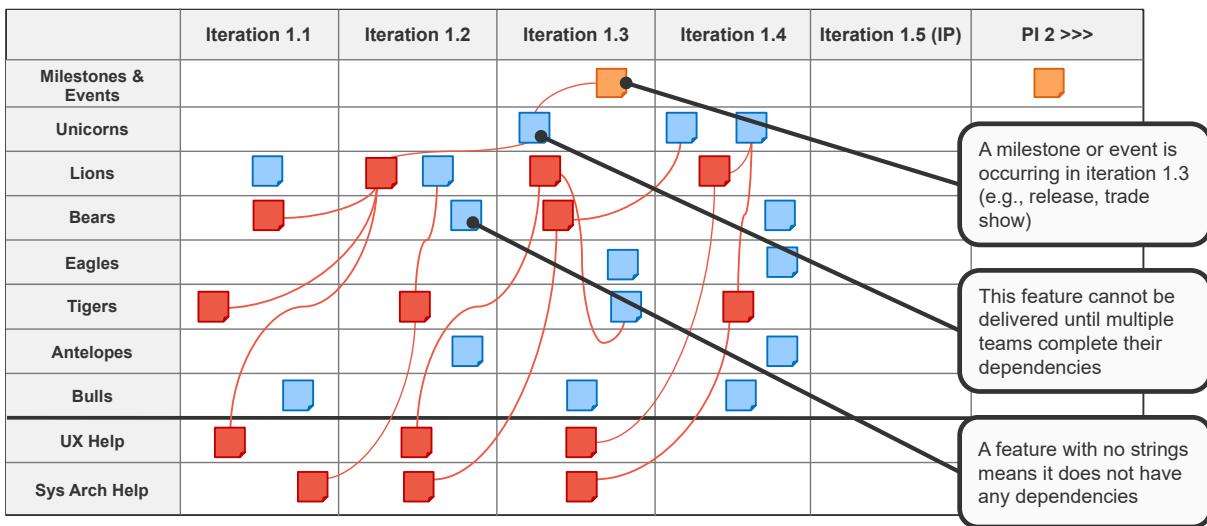
## Day 1 Agenda

- **Business context** – A Business Owner or senior executive describes the current state of the business, shares the portfolio vision, and presents a perspective on how effectively existing solutions address current customer needs.
- **Product/solution vision** – Product Management presents the current vision (typically represented by the top ten or so upcoming features). They highlight changes from the previous PI planning event and any relevant milestones.
- **Architecture vision and development practices** – The System Architect presents the architecture vision. Also, a senior development manager may introduce Agile-supportive changes to development practices, such as test automation, DevOps, Continuous Integration, and Continuous Deployment, which the teams will adopt in the upcoming PI.
- **Planning context and lunch** – The RTE presents the planning process and expected outcomes.
- **Team breakouts #1** – In the breakout, teams estimate their capacity for each Iteration and identify the backlog items they will likely need to realize the features (Figure 2). Each team creates draft plans, visible to all, iteration by iteration.



Figure 2. An Agile Team breakout

During this process, teams identify risks and dependencies and draft their initial team PI objectives. The PI objectives typically include 'uncommitted objectives,' which are goals built into the plan (for example, stories that have been defined and included for these objectives) but are not committed to by the team because of too many unknowns or risks. Uncommitted objectives are not extra things to do in case there is time. Instead, they increase the reliability of the plan and give management an early warning of any objectives that the ART may not be able to deliver. The teams also add the features and associated dependencies to the ART Planning Board, as shown in Figure 3.



ART Planning Board Legend:



Red strings (or lines for digital boards) are used to connect a feature or milestone to one or more dependencies. Sometimes a dependency has its own dependencies (see Lions in iteration 1.2)

© Scaled Agile, Inc.

Figure 3. ART planning board showing features and dependencies

- **Draft plan review** – During the tightly timeboxed draft plan review, teams present key planning outputs, which include capacity and load, draft PI objectives, potential risks, and dependencies. Business Owners, Product Management, and other teams and stakeholders review and provide input.
- **Management review and problem-solving** – Draft plans likely present challenges like scope, people and resource constraints, and dependencies. During the problem-solving meeting, management may negotiate scope changes and resolve other problems by agreeing to various planning adjustments. The RTE facilitates and keeps the primary stakeholders together for as long as necessary to make the decisions needed to reach achievable objectives.

Note: Solution Trains often hold an additional management review and problem-solving workshop after the first day of planning to address cross-ART issues. Alternatively, the RTEs of the involved trains may talk with each other to discuss the problems for the ART's specific management review and problem-solving meeting. The Solution Train Engineer (STE) helps facilitate and resolve issues across the ARTs.

## Day 2 Agenda

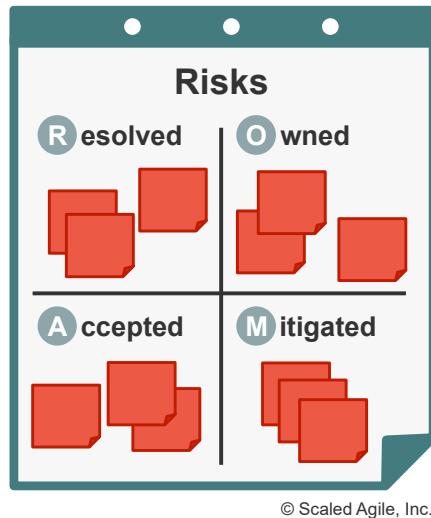
- **Planning adjustments** – The next day, the event begins with management presenting changes to the planning scope, people, and resources.
- **Team breakouts #2** – Teams continue planning and making the appropriate adjustments. They finalize their objectives for the PI, to which the Business Owners assign business value, as shown in Figure 4.

Objectives for PI 1	BV	AV
1. Show routing calculations between the five most frequent destinations	<b>10</b>	—
2. Navigate autonomously from distribution center to the most frequent destination	<b>4</b>	—
3. Parallel park for a delivery	<b>7</b>	—
4. Return to the distribution center after delivery	<b>10</b>	—
5. Include traffic data in route planning	<b>7</b>	—
6. Recall a delivery that is already in progress	<b>8</b>	—
<b>Uncommitted Objectives</b>		
7. Spike: Reduce GPS signal loss by 25%	<b>6</b>	—
8. Demonstrate real-time rerouting to avoid delays (e.g., accident, construction)	<b>5</b>	—

© Scaled Agile, Inc.

Figure 4. A team's PI objectives sheet with assigned business value

- **Final plan review and lunch** – All teams present their plans to the group during this session. At the end of each team's time slot, the team states its risks and impediments and provides the risks to the RTE for use later in the ROAMing exercise. Each Agile Team then asks the Business Owners if the plan is acceptable. If the plan is accepted, the team brings their team PI objective sheet to the front of the room so everyone can see the aggregate objectives unfold in real-time. If the Business Owners have concerns, teams can adjust the plan to address the identified issues. The team then presents its revised plan.
- **ART PI Risks** – During planning, teams have identified risks and impediments that could impact their ability to meet their objectives. These are resolved in a broader management context before the whole train. One by one, the risks are discussed and addressed with honesty and transparency and then grouped into one of the following categories:
  - **Resolved** – The teams agree that the risk is no longer a concern
  - **Owned** – Someone on the ART owns the risk since it cannot be addressed during PI planning
  - **Accepted** – Some items are simply facts or potential problems that must be understood and accepted
  - **Mitigated** – Teams identify a plan to reduce the impact of the risk



© Scaled Agile, Inc.

Figure 5. Example ROAM Board

- **A confidence vote in two parts** – To gauge readiness for the final plan review, each team conducts a voting mechanism called ‘fist of five’ physically or in a digital tool. This first vote often occurs during the last team breakout. Additionally, once ART PI risks have been addressed, the entire ART conducts a ‘fist of five’ physically or in a digital tool. If the average is three or above, then management should accept the commitment. If it’s less than three, the ART may need to rework its plan. Anyone voting two or fewer should be allowed to voice their concerns. These concerns might add to the risk list, require replanning, or provide information.

The Agile Teams and the ART complete a confidence vote that indicates the following commitment:

1. Teams agree to do everything in their power to meet the agreed-to objectives.
2. In the event that objectives are not achievable, teams agree to escalate immediately so that corrective action can be taken.

- **Plan rework** – If necessary, teams adjust their objectives until the ART has high confidence. This additional planning is one occasion where alignment and commitment are valued more highly than adhering to a timebox.
- **Planning retrospective and moving forward** – Finally, the RTE leads a brief retrospective for the PI planning event to capture what went well, what didn’t, and what to do better next time.

Read more for guidance on adapting this agenda to support planning across multiple time zones:

# What happens after PI Planning?

**Next steps** – Typically, a discussion about the next steps, along with final instructions to the teams, follows, including:

- Cleaning up the rooms used for planning (if applicable)
- Entering the team PI objectives and stories in Agile lifecycle management (ALM) tooling
- Reviewing team and ART events calendars
- Determining Iteration Planning and Team Sync locations and timing

After the planning event, the RTE and other ART stakeholders summarize the individual team PI objectives into a set of ART PI objectives (Figure 6) and use this to communicate externally and track progress toward the goals.

Product Management refines the roadmap using the ART PI objectives, improving the forecast for the following PIs.

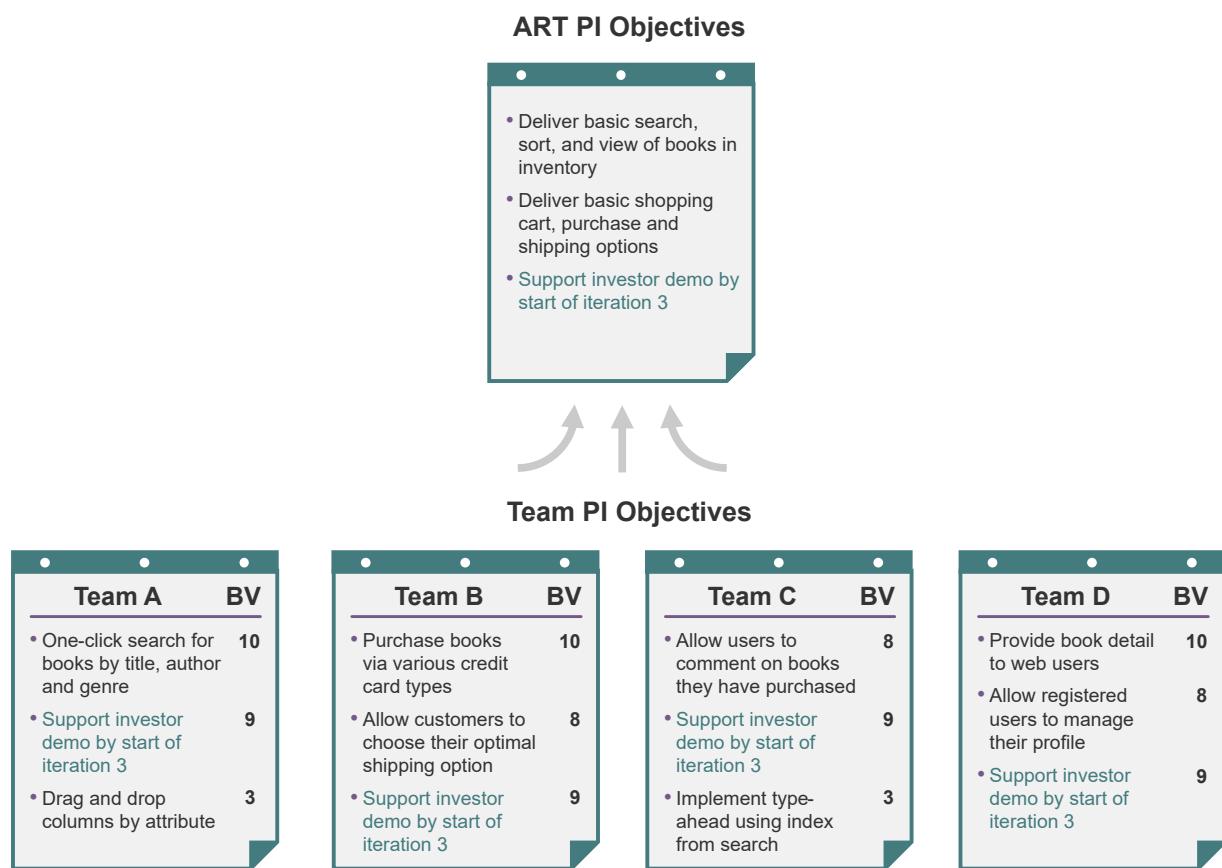


Figure 6. ART PI objectives

Teams leave the PI planning event with a prepopulated backlog for the upcoming PI. They take their team's PI objectives, plans, and risks to their regular work area or update their digital tool to accurately reflect the plan. ART risks remain with the RTE, which ensures that the people responsible for owning or mitigating a risk have captured the information and are actively managing the risk. The ART Planning Board is maintained physically or digitally throughout the PI to enable dependency and milestone management.

Most importantly, the ART executes the PI, tracking progress and adjusting as necessary as new knowledge emerges. Execution of the PI begins with all the teams conducting planning for the first iteration, using their PI plans as a starting point. It offers fresh input for the iteration planning processes that follow. Since the plans created during PI Planning did not consider detailed story-level acceptance criteria, the team will likely adjust the first and subsequent iteration plans.

Read more about creating and communicating with PI Objectives:

[PI Objectives](#)

## References

[1] Kennedy, Michael. *Product Development for the Lean Enterprise*. Oaklea Press, 2003.

### In this article

[What is PI Planning?](#)

[How does an ART prepare for PI Planning?](#)

[How to run a PI Planning event?](#)

[What happens after PI Planning?](#)

### Key Takeaways

- PI Planning is a critical event in SAFe where Agile Teams and stakeholders align as an Agile Release Train (ART) around a shared mission and plan.

- PI Planning provides the opportunity to align development objectives with business goals, identify dependencies, and plan work based on available capacity.
- Preparation involves ensuring organizational, content, and logistics readiness.
- PI Planning outputs include committed PI objectives from the Agile Teams and an ART planning board that reflects feature delivery dates and dependencies.

Last update: 15 October 2024

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## **Training**

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## **Content & Trademarks**

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## **Scaled Agile, Inc**

### **Contact Us**

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### **Business Hours**

Weekdays: 9am to 5pm

Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



+ Framework

ENGLISH (US) ▾



“ While building trust gives teams the ability to reconfigure and “do the right thing,” it is also necessary to make sure that team members know what the right thing is. Team members must all work toward the same goal, and in volatile, complex environments that goal is changeable.

—General Stanley McChrystal,  
*Team of Teams*

## Team Backlog

The Team Backlog is a Kanban system that is used to capture and manage the user stories and enablers intended to enhance the solution.

This includes stories originating from Features in the ART backlog as well as those arising from the team's local context.

## Details

The team backlog holds all the possible work that a team might do to enhance the solution. For example, it contains [User Stories](#), [Enablers](#), and other work items such as improvement stories for corrective actions from the team's [Retrospectives](#) or the [ART's Inspect and Adapt](#).

While it's not conceptually complex, some essential aspects make the team backlog critical for Agile development. For example:

- It contains all the work for an **Agile Team** to advance the solution and aligns all team members to a common goal.
- It's a list of wants, not commitments. Items can be estimated (preferable) or not, but it is just an ordered list, and there is no specific time commitment for completion. In other words, the backlog is time-independent, giving the team general flexibility regarding what gets implemented and when.
- All team members can enter stories into the backlog.
- It has an owner—the Product Owner (PO)—who helps the team manage the challenge of multiple stakeholders who may have divergent views of what's essential.

The PO, with input from the team and other stakeholders, is primarily responsible for creating and maintaining the team backlog. However, any team member can enter an item into the backlog for consideration. The PO prioritizes the backlog, balancing the needs of stakeholders. There are three primary inputs to the team backlog, as Figure 1 illustrates.

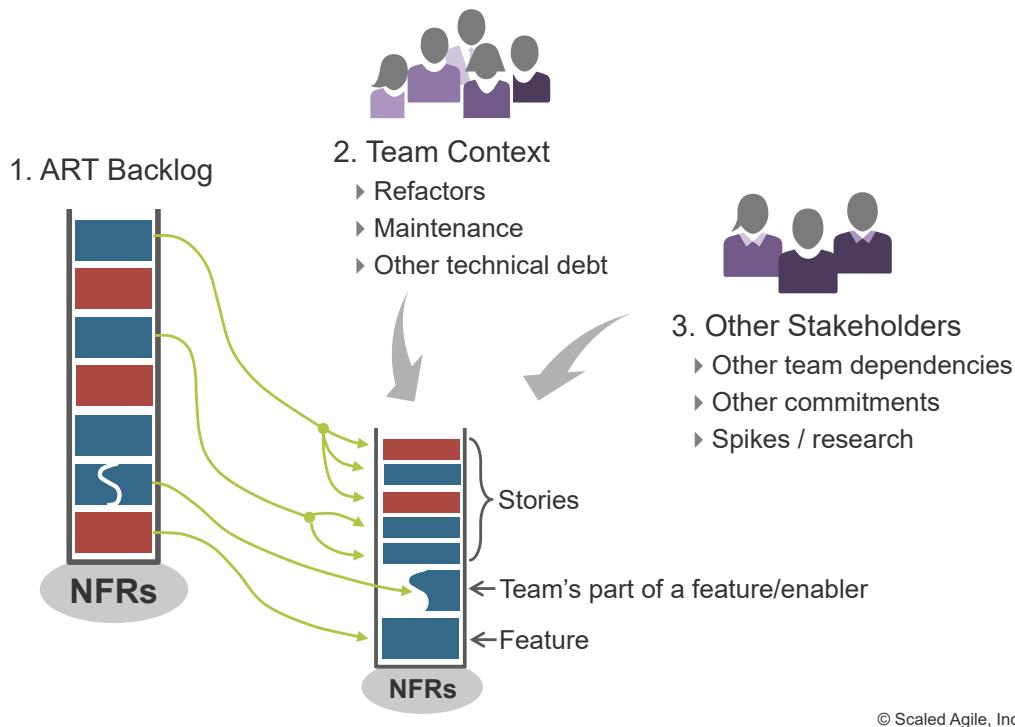


Figure 1. Input sources for a team backlog

- **ART Backlog** – The **ART Backlog** consists of upcoming features planned to be delivered by a train. During **PI Planning**, teams split the candidate features into stories and tentatively place them into upcoming **Iterations**. These new stories are maintained in the team backlog.
- **Team's local context** – The team's local concerns (other new functionality, defects, refactors, tech debt, and maintenance) are also in the backlog. Since PI planning is high-level, adjustments will likely occur during the PI. Teams using Scrum will probably make

adjustments during [Iteration Planning](#), while teams applying Kanban will likely do the same during backlog replenishment.

- **Other stakeholders** – Agile Teams on the ART are not islands, and their backlogs will contain some stories that support other teams' dependencies and other commitments, including the ART's [PI Objectives](#). These stories may include spikes for research required to estimate [Features](#), [Capabilities](#), and even [Epics](#).

Moreover, teams get feedback from previous increments, the [System Demo](#), and other groups that may affect the backlog.

[Nonfunctional Requirements \(NFRs\)](#) are persistent qualities that may affect the solution's design, performance, or quality. Since they serve as constraints (or restrictions) for all the team's items, the Big Picture illustrates them at the bottom of the backlog in Figure 1. Due to their importance, teams often automate acceptance tests for NFRs and include them in their definition of done (DoD).

## Building and Refining the Backlog

Agile Teams take a continuous, flow-based approach to maintain backlog readiness, so it always contains some stories ready for implementation without significant risk or surprise. Like a neglected garden that grows wild when left unattended for too long, the team backlog becomes unmanageable if not given care and attention. Refining the team backlog includes the following activities:

- Refining stories and establishing acceptance criteria
- The PO regularly prioritizes the team backlog in collaboration with the team and stakeholders
- New stories, including enablers, are discovered and described, and existing ones are changed or removed
- High-priority items are readied by defining acceptance criteria and sizing them to fit within small timeboxes
- Stories that have been around too long, or perhaps are no longer relevant, are removed

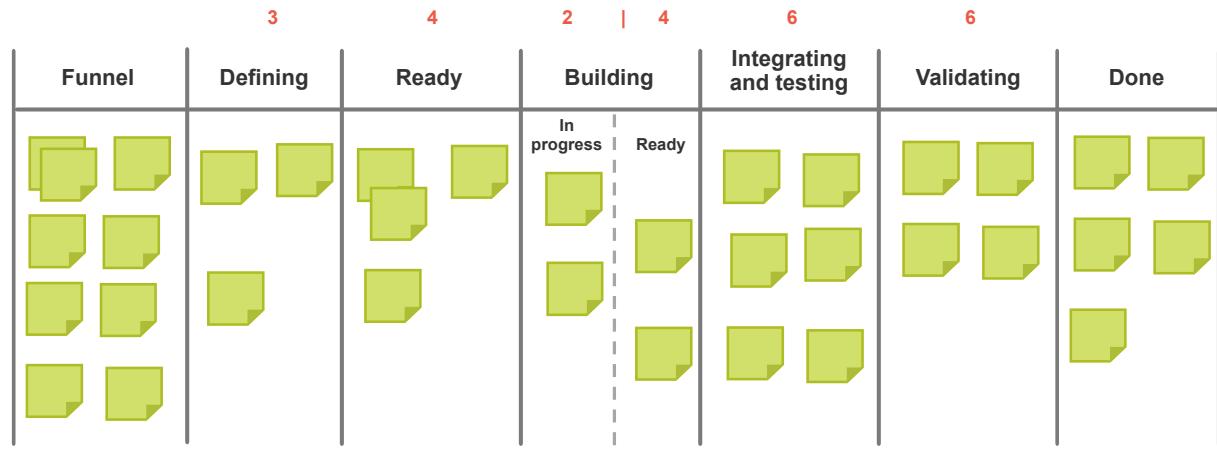
Although the PO manages the team backlog, refinement is a collaborative process. It creates a dialogue between the team, [Customers](#), and other stakeholders. This refinement breaks down barriers between the business and the development team, eliminating waste, handoffs, and delays. Developing story acceptance criteria increases the clarity of the requirements, leverages the team's collective knowledge and creativity, and creates buy-in and joint ownership.

There is no prescriptive meeting pattern for refining the backlog. Some teams like to do a bit of backlog refinement after their Team Sync. Others prefer weekly refinement sessions or requirements specification workshops, applying [Behavior-Driven Development](#) (BDD) techniques to help clarify stories. Since multiple teams often collaborate on feature development, new issues,

dependencies, and stories will likely arise. Backlog refinement also helps surface problems with the current plan, which may require discussion at the team, PO, or coach syncs.

## Managing the Backlog with Kanban

In SAFe, Agile Teams manage their backlog using a Kanban system. The backlog Kanban system facilitates alignment, visibility, and dependency management. Figure 2 illustrates an example of one team's initial Kanban system.



© Scaled Agile, Inc.

Figure 2. One Agile Team's initial Kanban board

This Kanban visualizes all active and pending work, workflow states, and work-in-process (WIP) limits. The system is WIP limited; a work item can be pulled into the next step only when the number of items is lower than the WIP limit. A few activities in the Kanban (typically beginning and end) may not be WIP-limited. The team defines and adjusts WIP limits, allowing it to adapt quickly to the flow of complex system development variations.

See the [Applying Kanban in SAFe](#) and [SAFe Team Kanban](#) articles for more information on establishing the team Kanban system.

## Balancing Value Delivery and System Health with Capacity Allocation

Like the ART, every [Agile Team](#) faces the problem of balancing internal work—maintenance, refactors, and technical debt—with the new user stories that deliver more immediate business value. While focusing solely on business functionality may work for a while, this approach will be short-lived as technical debt increases, ultimately slowing development velocity. Avoiding this risk requires continuous investment in evolving the solution's [Architectural Runway](#) while making customers happy with enhancements, new functionality, and bug fixes. Getting this balance right extends the system's life, deferring technical obsolescence.

But prioritizing different types of work can be challenging as the PO tries to compare the value of unlike things: defects, refactors, redesigns, technology upgrades, and new user stories. And there is no upper limit to the demand for any of these things.

In collaboration with the team, the PO applies *capacity allocation* (Figure 3) for each item type. Then the PO, team, and [System Architect](#) select the highest-priority backlog items for each capacity allocation slice during planning. Since many stories originate from features, PI planning commitments may predetermine some priorities. However, the PO can prioritize work from the team's local context by comparing value, size, and logical sequencing. Also, the PO can adjust the allocation percentage for each work item type to address long-term system health and value delivery. Teams should adapt the capacity allocation categories as needed. However, these categories should be consistent across teams in the ART.

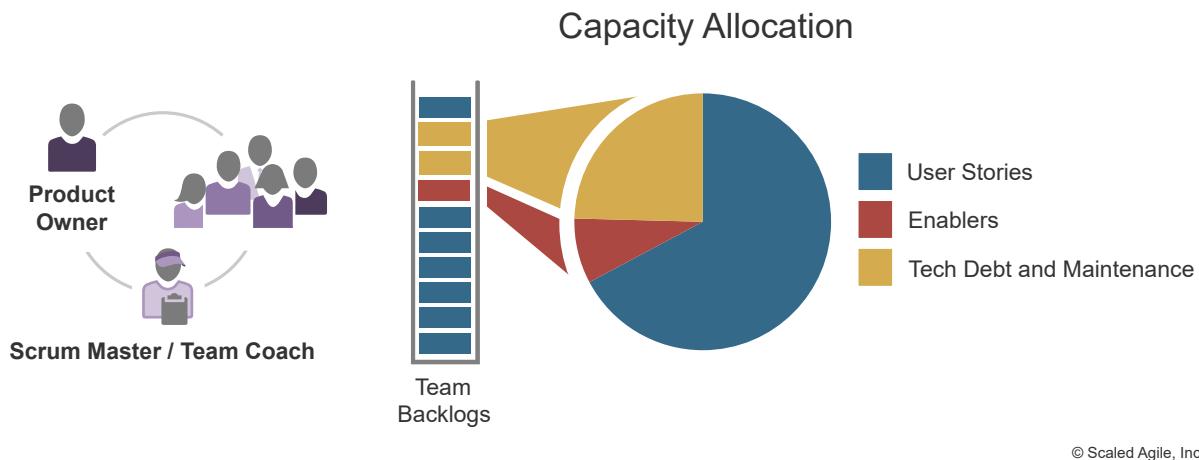


Figure 3. Typical examples of capacity allocation categories (user stories, enablers, and maintenance in this case)

## Learn More

[1] Knaster, Richard, and Dean Leffingwell. *SAFe 5.0 Distilled, Achieving Business Agility with the Scaled Agile Framework*. Addison-Wesley, 2020.

[2] Leffingwell, Dean. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*. Addison-Wesley, 2011.

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)





“ Stories act as a ‘pidgin language,’ where both sides (users and developers) can agree enough to work together effectively.

—Bill Wake, co-inventor of Extreme Programming

## Story

Stories are short descriptions of a small piece of desired functionality written from the user's perspective.

Agile Teams implement stories as small, vertical slices of system functionality that can be completed in a few days or less.

Stories are the primary artifact used to define system behavior in Agile. They are short, simple descriptions of functionality told from the user's perspective and written in their language. Each implements a small, vertical slice of system behavior.

Stories provide just enough information for business and technical people to understand the intent. Details are deferred until the story is ready to be implemented. Through acceptance criteria and acceptance tests, stories get more specific, helping to ensure system quality.

User stories deliver functionality directly to the end user. **Enabler** stories bring visibility to the work items needed to support exploration, architecture, infrastructure, and compliance.

## Details

SAFe describes a four-tier hierarchy of artifacts that outline functional system behavior: [Epic](#), [Capability](#), [Feature](#), and Story. Collectively, these artifacts are used to describe the solution's intended behavior. The detailed implementation work is expressed through stories, which comprise the [Team Backlog](#). Some stories emerge from business and enabler features in the [ART Backlog](#), while others come from the team's local context.

Each story is a small, independent behavior that can be implemented incrementally and provides some value to the user or the [Solution](#). It's a vertical slice of functionality to ensure that every [Iteration](#) delivers new value. Stories are small and must be completed in a single iteration (see the splitting stories section).

Often, stories are first written on an index card or sticky note. The physical nature of the card creates a tangible relationship between the team, the story, and the user: it helps engage the entire team in story writing. Sticky notes also offer other benefits: they help visualize work and can be readily placed on a wall or table, rearranged in sequence, and even passed off when necessary. Stories allow an improved understanding of the scope and progress:

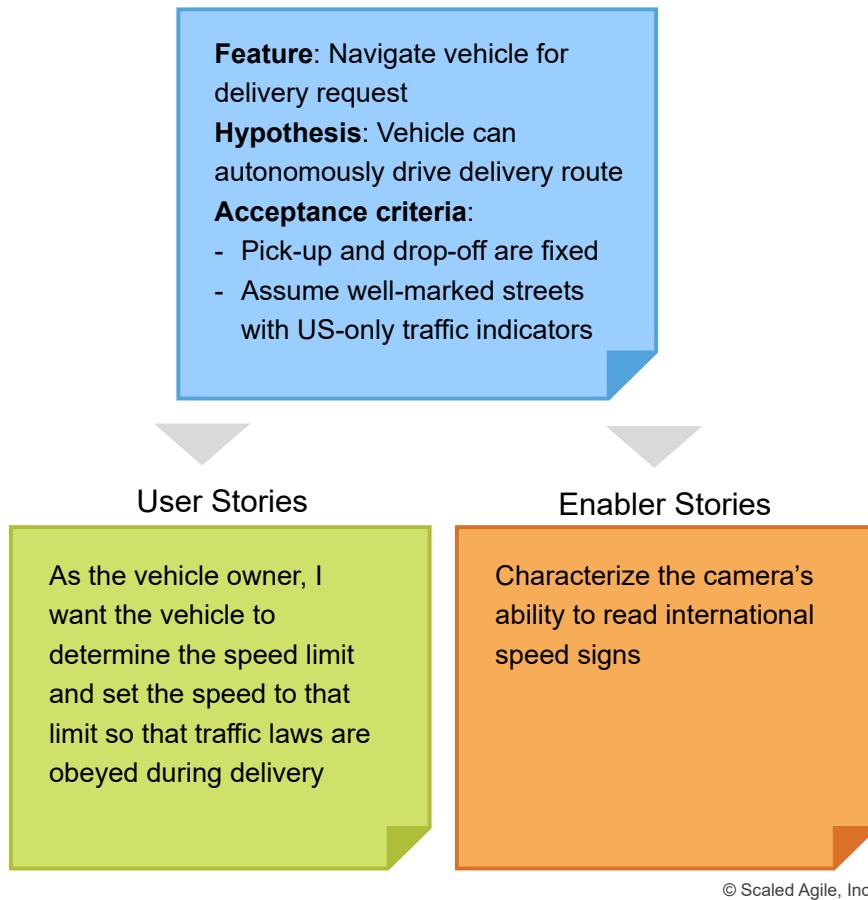
- "Wow, look at all these stories we are about to sign up for" (scope)
- "Look at all the stories we accomplished in this iteration" (progress)

While anyone can write stories, approving them into the team backlog and accepting them into the system baseline are the Product Owner's responsibility. Of course, stickies don't scale well across the [Enterprise](#), so stories often move quickly into Agile Lifecycle Management (ALM) tooling.

There are two types of stories in SAFe, user stories and enabler stories, as described below.

## Sources of Stories

Stories are typically driven by splitting business and enabler features, as Figure 1 illustrates.



© Scaled Agile, Inc.

Figure 1. Example of a business feature split into stories

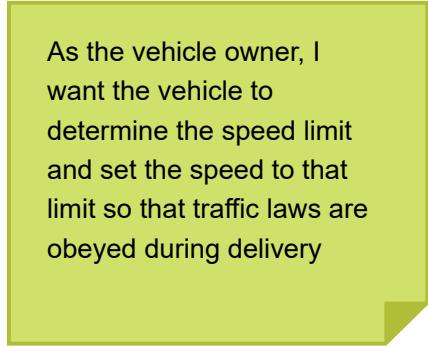
## User Stories

User stories are the primary means of expressing needed functionality. They essentially replace the traditional requirements specification. In some cases, however, they serve as a means to explain and develop system behavior later recorded in specifications supporting compliance, suppliers, traceability, or other needs.

Because they focus on the user as the subject of interest and not the system, user stories are value and customer-centric. To support this, the recommended form of expression is the 'user-voice form,' as follows:

As a (user role), I want to (activity) so that (business value)

By using this format, the teams are guided to understand who is using the system, what they are doing with it, and why they are doing it. Applying the 'user voice' format routinely tends to increase the team's domain competence; they come to better understand the real business needs of their user. Figure 2 provides an example.

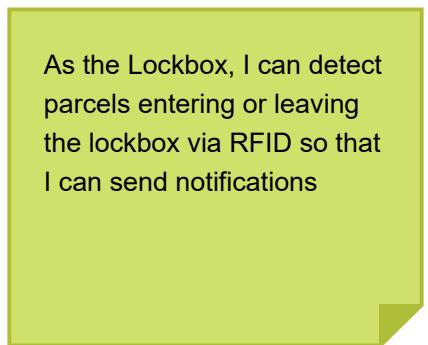


© Scaled Agile, Inc.

Figure 2. Example user story in user voice form

As described in [Design Thinking](#), personas describe specific characteristics of representative users that help teams better understand their end user. Example personas for the rider in Figure 2 could be a thrill-seeker ‘Jane’ and a timid rider ‘Bob.’ Stories descriptions can then reference these personas (As Jane I want...).

While the user story voice is typical, not every system interacts with an end user. Sometimes the ‘user’ is a device (for example, printer) or a system (for example, transaction server). In these cases, the story can take on the form illustrated in Figure 3.

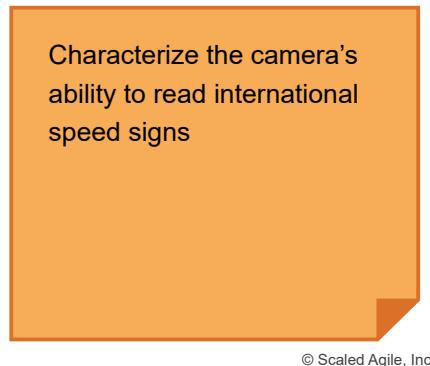


© Scaled Agile, Inc.

Figure 3. Example of a user story with a ‘system’ as a user

## Enabler Stories

Teams also develop the new architecture and infrastructure needed to implement new user stories. In this case, the story may not directly touch any end user. Teams use ‘enabler stories’ to support exploration, architecture, or infrastructure. Enabler stories can be expressed in technical rather than user-centric language, as Figure 4 illustrates.



© Scaled Agile, Inc.

Figure 4. Example enabler story

There are many other types of Enabler stories, including:

- [Refactoring](#) and [Spikes](#) (as traditionally defined in XP)
- Building or improving development/deployment infrastructure
- Running jobs that require human interaction (for example, indexing 1 million web pages)
- Creating the required product or component configurations for different purposes
- Verification of system qualities (for example, performance and vulnerability testing)

Enabler stories are demonstrated just like user stories, typically by showing the knowledge gained, artifacts produced, or the user interface, stub, or mock-up.

## Writing Good Stories

Good stories require multiple perspectives. In Agile, the entire team creates a shared understanding of what to build to reduce rework and increase throughput. Teams collaborate using [Behavior-Driven Development](#) (BDD) to define detailed acceptance tests that definitively describe each story.

Collaborative story writing ensures all perspectives are addressed, and everyone agrees on the story's behavior with the results represented in the story's description, acceptance criteria, and acceptance tests. The acceptance tests are written using the system's domain language using BDD. BDD tests are then automated and run continuously to maintain [Built-In Quality](#). The BDD tests are written against system requirements (stories) and, therefore, can be used as the definitive statement for the system's behavior, replacing document-based specifications.

## The 3Cs: Card, Conversation, Confirmation

Ron Jeffries, one of the inventors of XP, is credited with describing the 3Cs of a story:

- **Card** – Captures the user story's statement of intent using an index card, sticky note, or tool. Index cards provide a physical relationship between the team and the story. The card

size physically limits story length and premature suggestions for the specificity of system behavior. Cards also help the team ‘feel’ upcoming scope, as there is something materially different about holding ten cards in one’s hand versus looking at ten lines on a spreadsheet.

- **Conversation** – Represents a “promise for a conversation” about the story between the team, customer (or the customer’s proxy), the PO (who may be representing the customer), and other stakeholders. The discussion is necessary to determine the more detailed behavior required to implement the intent. The conversation may spawn additional specificity in the form of acceptance criteria (the confirmation below) or attachments to the user story. The conversation spans all steps in the story’s life cycle:
  - Backlog refinement
  - Planning
  - Implementation
  - Demo

These just-in-time discussions create a shared understanding of the scope that formal documentation cannot provide. Specification by example replaces detailed documentation. Conversations also help uncover gaps in user scenarios and NFRs.

- **Confirmation** – The acceptance criteria provide the information needed to ensure that the story is implemented correctly and covers the relevant functional and NFRs. Figure 5 provides an example. Some teams often use the confirmation section of the story card to write down what they will demo.

<p><b>Given</b> vehicle driving at the current speed limit</p> <p><b>When</b> a new speed limit is lower</p> <p><b>Then</b> the current speed becomes the new speed before the new limit applies</p>	<p><b>Given</b> driving at the current speed limit</p> <p><b>When</b> a slower vehicle is detected in front</p> <p><b>Then</b> the current speed is reduced to maintain the minimum safe distance</p>
--	---

© Scaled Agile, Inc.

Figure 5. Story acceptance criteria with BDD

[Agile Teams](#) automate acceptance tests wherever possible, often in business-readable, domain-specific language. Automation creates an executable specification to validate and verify the solution. Automation also provides the ability to quickly regression-test the system, enhancing [Continuous Integration](#), refactoring, and maintenance.

## Investing in Good Stories

Agile teams spend significant time discovering, elaborating, and understanding user stories and writing acceptance tests. This is as it should be, because it represents the fact that:

*Writing the code for an understood objective is not necessarily the most challenging part of software development.*

Instead, it is understanding the real objective of the code. Therefore, investing in good user stories, albeit at the last responsible moment, is a worthy effort for the team. Bill Wake coined the acronym INVEST [1] to describe the attributes of a good user story.

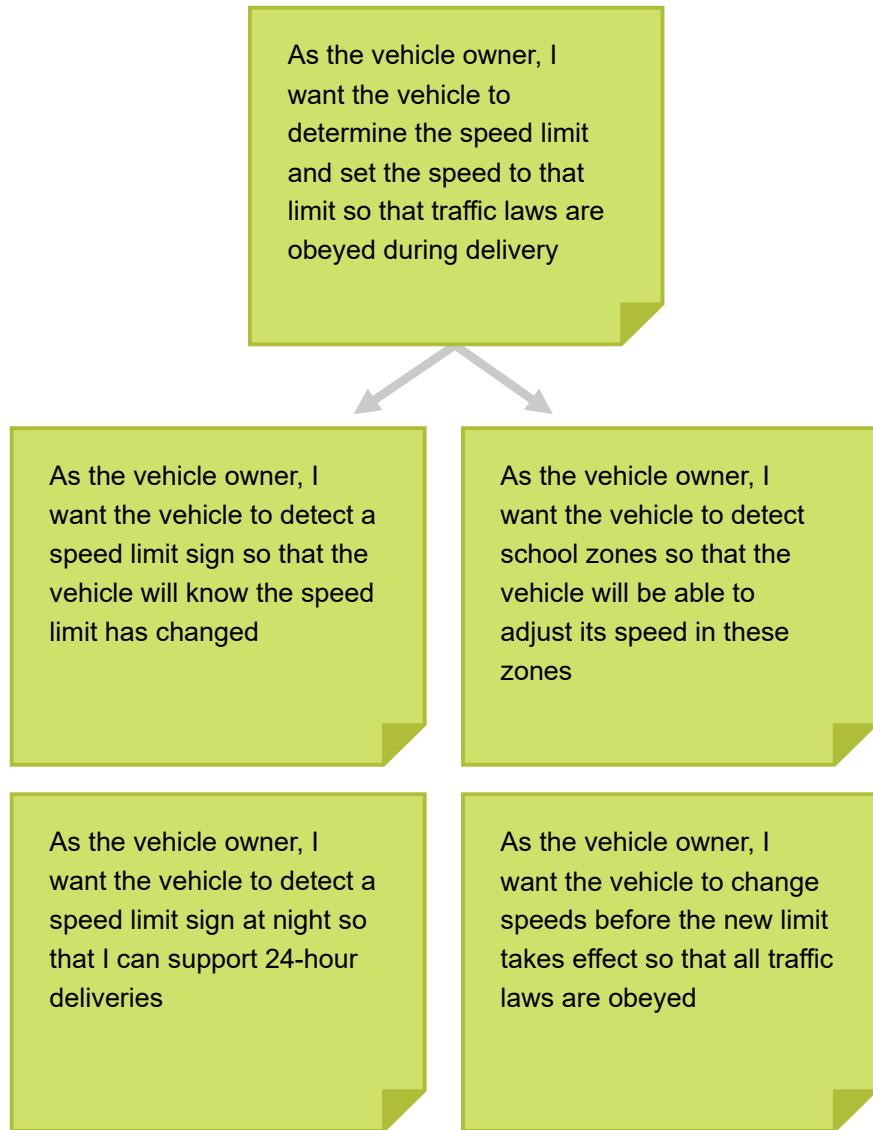
- **I** – Independent (among other stories)
- **N** – Negotiable (a flexible statement of intent, not a contract)
- **V** – Valuable (providing a valuable vertical slice to the customer)
- **E** – Estimable (small and negotiable)
- **S** – Small (fits within an iteration)
- **T** – Testable (understood enough to know how to test it)

## Splitting Good Stories

Smaller stories allow faster, more reliable implementation since small items flow through any system faster, with less variability and reduced risk. Therefore, splitting bigger stories into smaller ones is a mandatory skill for every Agile team. It's both the art and the science of incremental development. *Agile Software Requirements* describes ten ways to split stories [1]. A summary of these techniques follows:

- Workflow steps
- Business rule variations
- Major effort
- Simple/complex
- Variations in data
- Data entry methods
- Deferred system qualities
- Operations (ex., Create, Read, Update, Delete [CRUD])
- Use-case scenarios
- Break-out spike

Figure 6 illustrates an example of splitting by use-case scenarios.



© Scaled Agile, Inc.

Figure 6. An example of splitting a big Story into smaller stories

## Estimating Stories

Agile teams use story points and 'estimating poker' to value their work [1, 2]. A story point is a singular number that represents a combination of qualities:

- **Volume** – How much is there?
- **Complexity** – How hard is it?
- **Knowledge** – What's known?
- **Uncertainty** – What's unknown?

Story points are relative, without a connection to any specific unit of measure. Each story's size (effort) is estimated relative to the smallest story, which is assigned a size of 'one.' A modified

Fibonacci sequence (1, 2, 3, 5, 8, 13, 20, 40, 100) [2] is applied that reflects the inherent uncertainty in estimating, especially large numbers (for example, 20, 40, 100).

## Estimating Poker

Agile teams often use '*estimating poker*,' which combines expert opinion, analogy, and disaggregation to create quick but reliable estimates. Disaggregation refers to splitting a story or feature into smaller, easier-to-estimate pieces.

(Note that there are several other methods used as well.) The rules of estimating poker are:

- Participants include all team members
- Each estimator is given a deck of cards containing the modified Fibonacci sequence
- The PO participates but does not estimate
- The [Scrum Master/Team Coach](#) participates but does not estimate unless they are doing actual development work
- For each backlog item to be estimated, the PO reads the story's description
- Questions are asked and answered
- Each estimator privately selects an estimating card representing their estimate
- All cards are turned over at the same time to avoid bias and to make all estimates visible
- High and low estimators explain their estimates
- After a discussion, each estimator re-estimates by selecting a card
- The estimates will likely converge; if not, the process is repeated

Some amount of preliminary design discussion is appropriate. However, spending too much time on design discussions is often a wasted effort. The real value of estimating poker is agreeing on a story's scope. It's also fun!

## Velocity

The team's velocity for an iteration is equal to the sum of the points for all the completed stories that met their definition of done (DoD). As the team works together over time, their average velocity (completed story points per iteration) becomes reliable and predictable. Predictable velocity assists with planning and helps limit Work in Process (WIP), as teams don't take on more stories than their historical velocity would allow. This measure also estimates how long it takes to deliver epics, features, capabilities, and enablers, which are also forecasted using story points.

## Capacity

Capacity is the portion of the team's velocity that is available for any given iteration. Vacations, training, and other events can make team members unavailable to contribute to an iteration's

goals for some portion of the iteration. This decreases the maximum potential velocity for that team for that iteration. For example, a team that averages 40 points delivered per iteration would adjust their maximum velocity down to 36 if a team member is on vacation for one week. Knowing this in advance, the team only commits to a maximum of 36 story points during iteration planning. This also helps during PI Planning to forecast the actual available capacity for each iteration in the PI, so the team doesn't over-commit when building their PI Objectives.

## Starting Baseline for Estimation

In standard Scrum, each team's story point estimating—and the resulting velocity—is a local and independent concern. At scale, it becomes difficult to predict the story point size for larger epics and features when team velocities vary wildly. To overcome this, SAFe teams initially calibrate a starting story point baseline where one story point is defined roughly the same across all teams. There is no need to recalibrate team estimation or velocity. Calibration is performed one time when launching new [Agile Release Trains](#).

*Normalized* story points provide a method for getting to an agreed starting baseline for stories and velocity as follows:

- Give every developer-tester on the team eight points for a two-week iteration (one point for each ideal workday, subtracting two days for general overhead).
- Subtract one point for every team member's vacation day and holiday.
- Find a small story that would take about a half-day to code and a half-day to test and validate. Call it a 'one.'
- Estimate every other story relative to that 'one.'

**Example:** Assuming a six-person team composed of three developers, two testers, and one PO, with no vacations or holidays, then the estimated initial velocity =  $5 \times 8$  points = 40 points/iteration. (Note: Adjusting slightly lower may be necessary if one of the developers and testers is also the Scrum Master/Team Coach.)

In this way, story points are somewhat comparable across teams. Management can better understand the cost for a story point and more accurately determine the cost of an upcoming feature or epic.

While teams will tend to increase their velocity over time—and that's a good thing—in reality, the number tends to remain stable. A team's velocity is far more affected by changing team size and technical context than by productivity variations.



**Note:** SAFe Team Kanban teams typically spend less time estimating stories than scrum teams do. In the Kanban flow-based model, work items or stories are typically split and sized so that the team can generally deliver a story within a few days. In the

context of SAFe where teams need to participate in iteration planning and assign stories to future iterations, some notion of sizing is required.

SAFe Kanban teams may initially use estimating poker or a similar mechanism to size their stories. More likely, however, they develop a sense of breaking work into stories that are similar in size, as that assists flow in general and assures that no large story blocks other stories that also need to make their way through the Kanban system. As they understand their velocity, they are able to understand how many stories they can deliver in a unit of time, allowing them to place stories in iterations during PI Planning and to be able to make commitments to other teams as to when specific stories would be available.

For teams doing regular maintenance and support activities, estimating their normal backlog items often has less value. In many cases, these teams do not estimate this type of response work. However, all teams have retro items, potential improvements to their CD pipeline, and other significant tasks that require attention, scheduling, and estimating.

---

## Learn More

[1] Leffingwell, Dean. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*. Addison-Wesley, 2011.

[2] Cohn, Mike. *User Stories Applied: For Agile Software Development*. Addison-Wesley, 2004.

Last update: 7 December 2022

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



## + Framework

ENGLISH (US)



“ While we must acknowledge emergence in design and system development, a little planning can avoid much waste.

—James Coplien, *Lean Architecture*

## Architectural Runway

The Architectural Runway consists of the existing code, components, and technical infrastructure needed to implement near-term features with minimal redesign and delay.

Architectural Runway enables a continuous flow of value through the [Continuous Delivery Pipeline](#), providing the technology required to quickly define, build, validate, and release [Features and Capabilities](#). It also supports the practice of [Agile Architecture](#) by allowing an organization's technology landscape to evolve in response to changing business needs.

## Details

Agile development avoids big design up-front (BDUF) with a simple belief that “the best architectures, requirements, and designs emerge from self-organizing teams [1].” This yields the practice of *emergent design*—defining and extending the architecture only as necessary to deliver the next increment of functionality.

However, emergent design alone cannot handle the complexity of large-scale [Solution](#) development. At scale, relying solely on emergent design leads to the following problems:

- Lack of standards increases delivery costs and delays

- One-off solutions become difficult to change and maintain
- Systems become vulnerable to security and stability issues
- Quality becomes dependent on tribal knowledge
- Solution components have poor interoperability and reusability

These problems can result in poor solution performance, unfavorable economic outcomes, and delayed time-to-market. Organizations overcome these problems by balancing emergent design with *intentional architecture*, which requires some centralized planning and cross-team coordination. Both are implemented with **Enablers** and, together, 'pave' the architectural runway (Figure 1).

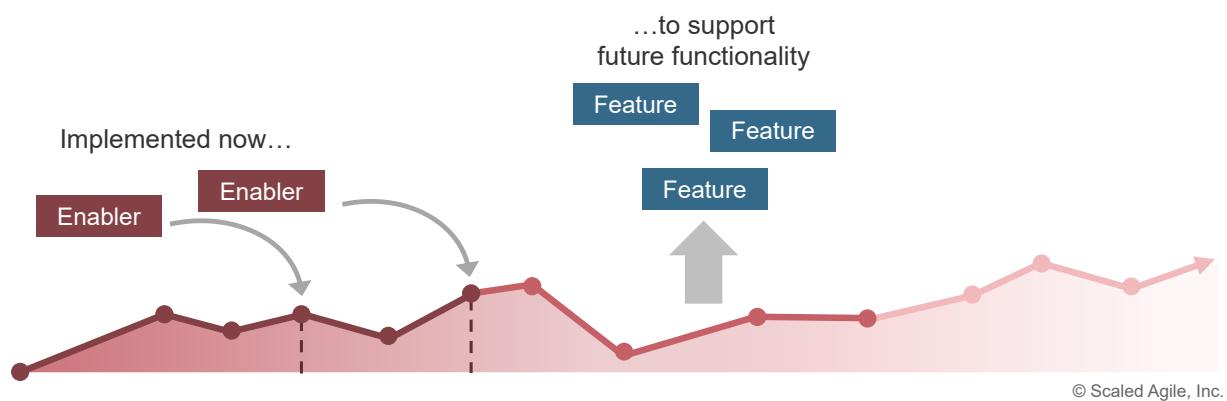


Figure 1. Architectural runway evolves in support of dynamic business needs

## Intentional Architecture Supports System Evolution

**Agile Teams** apply emergent design through the development of enabler **Stories** that incrementally evolve their parts of the solution. However, they typically lack the breadth of knowledge of the end-to-end solution that would allow them to design components and infrastructure across all domains effectively. Many of these design factors are simply beyond their scope of responsibility.

Intentional architecture accounts for these systemic design considerations and provides purposeful, planned architectural guidelines that enable the independent work of Agile teams to be integrated into a cohesive, sustainable solution. These cross-cutting guidelines are typically defined by **Enterprise Architects**, **Solution Architects**, and **System Architects** in close collaboration with **Product Management** and **Solution Management**. These roles are well suited to the task because of their broad knowledge of the technology landscape and deep knowledge of the **Solution Context**.

Intentional architecture is codified as enabler epics, capabilities, and features in the Portfolio Backlog, Solution Backlog, and ART Backlog. Architects then help steer the enablers through the appropriate Kanban systems, ensuring they produce the intended architectural runway.

# Building the Architectural Runway

Several roles may be involved in defining architectural runway, but its implementation is the responsibility of Agile teams. These teams are often responsible for delivering end-user-focused products or services. Therefore, building the architectural runway should not overly constrain them.

Just the right amount of architectural runway is required at any given time. Too much, and the architecture bottlenecks the teams and over-engineers the current solution increment. Too little, and the organization will not have the runway it needs to meet near-term business commitments. Capacity allocation is applied to the [Team Backlog](#) to help ensure that the ratio of enabler work to customer-facing work is always in balance.

A dedicated Agile team often oversees the initial implementation when significant investment in architectural runway is called for—such as to enable a new product launch, Horizon 3 portfolio initiative, or legacy environment. (Figure 2).

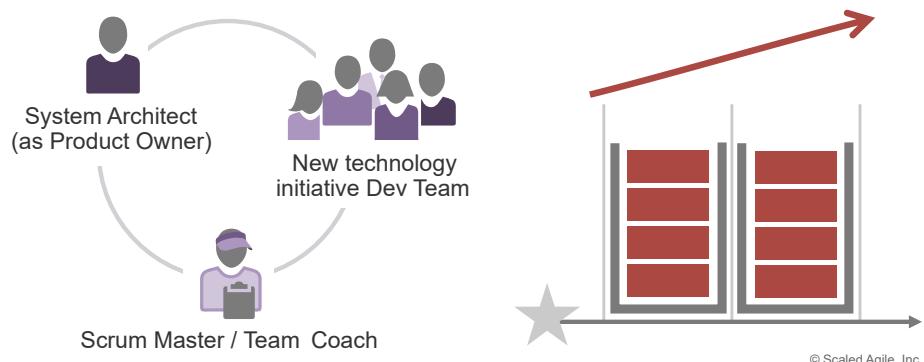
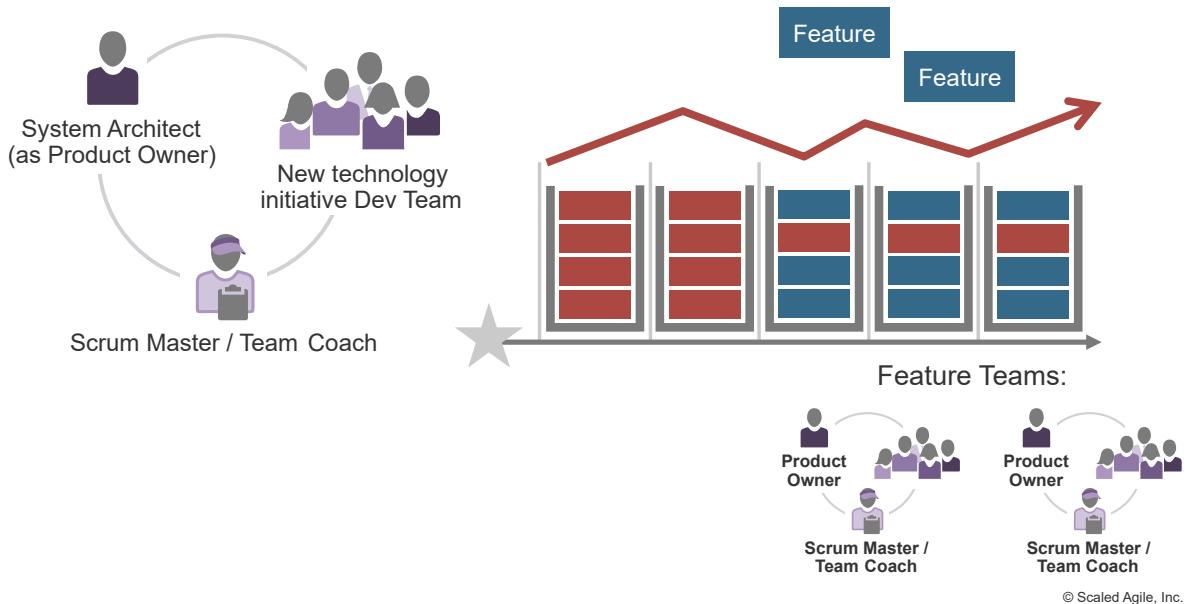


Figure 2. A dedicated Agile team establishing initial architectural runway

In this scenario, an architect often assumes the role of [Product Owner](#), acting as the voice of the customer and content authority over a backlog composed primarily of enablers. A specialized group of team members handles development, and a Scrum Master/Team Coach guides execution.

This team continues until the volume of runway work required by the ART is no longer sufficient to warrant a dedicated team. At this point, the responsibility for implementing additional architectural runway as needed becomes shared among the ART's persistent Agile teams, as illustrated in Figure 3.



**Figure 3. Agile teams share responsibility for extending the architectural runway**

Regardless of who performs the work, the rules for building the runway are both simple and Agile:

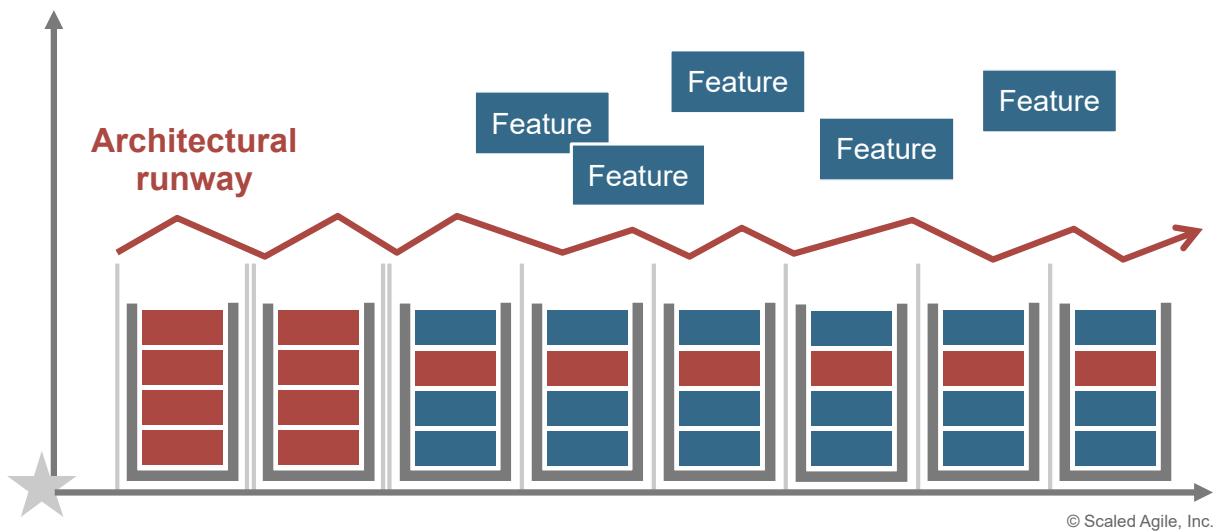
- Teams that build the runway iterate like every other Agile team on the ART
- The value is in working systems, not models or specifications
- Enabler features and stories should take no more than a PI or iteration, respectively, to deliver
- Agile teams are stakeholders of the architectural runway, leveraging it to deliver on customer commitments and providing feedback on its effectiveness

## Continually Invest in the Architectural Runway

Architectural runway is dynamic. It is 'consumed' by delivering near-term features and must be extended to support future features. The following are examples of forces that consume architectural runway:

- **Fast-moving Agile teams** – They quickly use the newest runway to deliver near-term features
- **Customer preferences** – After investing in architectural runway, stakeholders often shift backlog priorities to the features that directly benefit customers
- **Technology innovation** – Technology changes rapidly, rendering existing runway obsolete
- **Changing business needs** – The needs of the enterprise shift in response to emerging opportunities and threats

Architectural runway is consumed to deliver planned features. As business needs change and new features are requested, additional runway is required (Figure 4).



**Figure 4. Architectural runway is produced and consumed as business needs evolve**

Because the development of new features and capabilities consumes the architectural runway, a continual investment must be made to extend it. Teams commit to extending the runway as needed in each iteration to support quick, sustainable delivery velocity. This could include adding automation to the CDP, enhancing [DevOps](#) practices, increasing server capacity, or any other activity that accelerates delivery velocity.

## The Architectural Runway in Large Solutions

When building really big systems, the architectural runway takes on an even more critical role as multiple ARTs contribute to the same Solution as part of a Solution Train. The [Enterprise Solution Delivery](#) article describes ten Lean-Agile practices that guide large solution delivery, several of which relate directly to architectural runway:

- **Specify the solution incrementally** – The [Solution Intent](#) defines many constraints as Non-Functional Requirements (NFRs). Many NFRs are cross-cutting concerns that can be addressed and simplified by building architectural runway that supports integration and testing. Architectural runway also allows elements of the solution intent to smoothly and incrementally evolve from variable to fixed.
- **Apply multiple planning horizons** – Large solutions generally require a longer architectural runway, implemented with enabler epics over several PIs or even years. Efficient delivery of these long stretches of runway is orchestrated through connected iteration plans, PI roadmaps, and solution roadmaps.
- **Design for change** – Achieving these goals in large solutions requires intentional architecture to implement an effective Continuous Delivery Pipeline and ensures simplified operations with strong feedback via application telemetry.
- **Continually address compliance concerns** – A Lean approach to [Compliance](#) automates the collection and testing of compliance data to provide more effective and predictable outcomes. This goal requires early engagement with auditors and stakeholders to agree on

an acceptable approach. Creating capabilities through the architectural runway ensures consistency and removes significant compliance risks.

- **Evolve deployed systems** – A fast, economical Continuous Delivery Pipeline means solutions—and changes to those solutions—can be released on demand. Architectural runway delivers a CDP whereby deployed solutions can evolve incrementally with minimal customer disruption.

It is critical to implement long runway incrementally. Enabler epics are split into enabler features and/or capabilities, then implemented by ARTs. Each enabler feature must be completed within a PI, and enabler stories within an iteration. This allows significant investments in architectural runway to evolve with the right balance of intentional architecture and emergent design.

## The Backstory of the Architectural Runway

The term ‘architectural runway’ started as an analogy while observing PI-level burn-down charts. Often, when there isn’t enough architectural runway when teams start a PI, any features dependent on the new architecture are high risk.

ARTs can’t always ‘land those PIs’ (bring the burn-down to zero at the end of the PI). In that case, they don’t meet the PI objectives. A PI, like an airplane, needs enough runway to land safely.

In SAFe, the architectural runway line goes up and down over time because it is built up, then used, then built up some more, then used, and so on. There has to be just the right amount at any point to support near-term business objectives.

To extend the runway metaphor, the bigger the aircraft (system) and the faster the flying speed (velocity), the more runway is needed to land the PI safely.

---

## Learn More

[1] Manifesto for Agile Software Development. <http://www.agilemanifesto.org>

Last update: 9 January 2023

## **Framework**

[Download SAFe Posters & Graphics](#)

[Blog](#)

## **Training**

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## **Content & Trademarks**

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## **Scaled Agile, Inc**

### **Contact Us**

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### **Business Hours**

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



## + Framework

ENGLISH (US) ▾



“ Prediction is very difficult, especially if it is about the future.

—Widely attributed to Niels Bohr,  
Danish physicist

## Roadmap

The Roadmap is a schedule of events and milestones that forecasts and communicates planned solution deliverables over a time horizon.

Roadmaps are a visual tool that assists in the development and communication of planned deliverables, milestones, and investments over time and help distinguish different types of work. Roadmaps are the glue that links strategy to execution and offer the ability to develop, evolve and adjust planned activities. They also provide stakeholders with a view of the current, near-term, and longer-term deliverables that realize some portion of the [Portfolio Vision](#) and [Strategic Themes](#).

## Details

“Responding to change over following a plan” is one of the four values of the Agile Manifesto [1]. While this value emphasizes responding to change, it assumes that there is a plan to follow. Indeed, planning and road mapping are fundamental to Agile.

There are three types of roadmaps defined in SAFe:

1. **PI roadmap** – Illustrates commitments for an [Agile Release Train \(ART\)](#) or [Solution Train](#) for the planned, upcoming [PI](#). The forecast may provide the deliverables and milestones for the following two PIs.

2. **Solution roadmap** – Provides a *longer-term*—often multiyear view—showing the key milestones and deliverables needed to achieve the solution [Vision](#) over time.
3. **Portfolio roadmap** – Shows an *aggregated multiyear* view of how LPM will achieve the portfolio vision across all its [Value Streams](#).

While forecasts for innovation and invention are inherently uncertain, organizations still need roadmaps for many reasons:

- **Steering significant initiatives** – Some initiatives may take years to develop, requiring a longer horizon for planning and coordinating activities and deliverables to achieve the business objectives.
- **Preparing for releases** – [Customers](#), [Suppliers](#), and partners need to understand how [Solutions](#) might be implemented and evolve and how they will achieve the vision. Customers also need time to plan for changes and how to test and implement the solution in their environment.
- **Addressing strategic concerns** – [Government](#) agencies and others periodically publish new regulations. Ensuring [Compliance](#) is a strategic concern; road mapping can help visualize and track the necessary deliverables and milestones.
- **Aligning stakeholders** – Internal stakeholders such as finance, sales, and marketing need time to align with the development organization to establish financial forecasts, build and execute sales and marketing campaigns, and communicate with partners and [Customers](#).

*Milestones* are an essential element of roadmaps as they mark specific progress points on the development timeline and are critical for understanding and monitoring product evolution and risk. SAFe defines the following types of milestones:

- **Pls** are time-based milestones that appear on most roadmaps as they provide cadence-based, objective measures of progress
- **Fixed-date** milestones include events, release dates, contractual obligations, and scheduling of deliverables that must occur on or before a specific date.
- **Learning milestones** help validate technical and business opportunities and hypotheses.

Roadmaps, including milestones, provide stakeholders with a means to understand, collaboratively shape, and plan for future solutions, helping the enterprise, [Customers](#), and [Suppliers](#) achieve the desired outcomes.

## Applying Planning Horizons

Creating effective roadmaps requires an understanding of the appropriate time horizon. If the planning horizon is too short, the enterprise may jeopardize alignment and the ability to communicate new future [Features and Capabilities](#). Too long, the enterprise is basing assumptions

and implying commitments on an uncertain and distant future. Multiple planning horizons provide a proper balance (Figure 1).

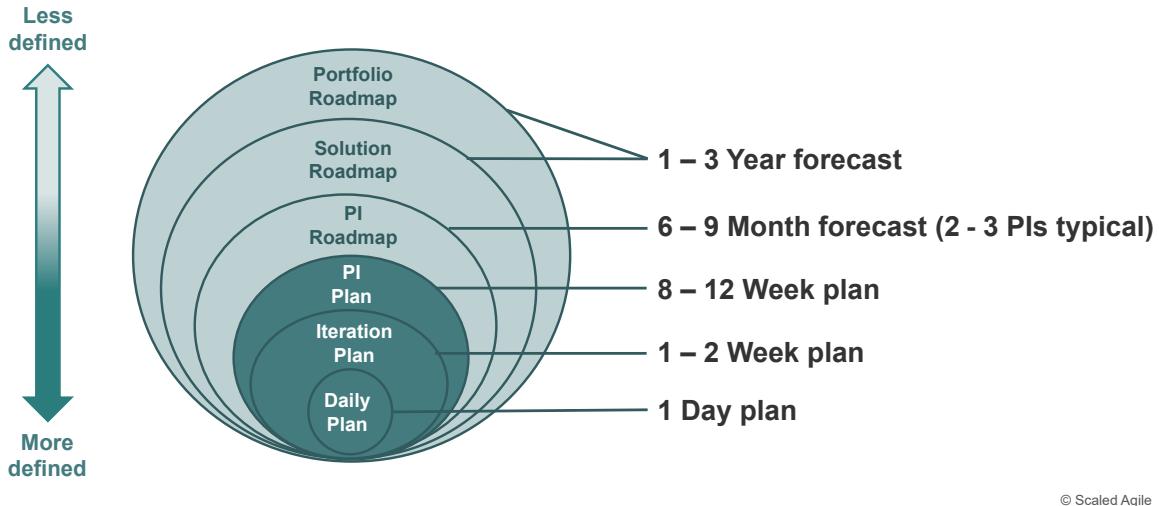


Figure 1. SAFe planning horizons

The outer levels of the planning horizon are longer-term and describe less defined and less committed behavior. In contrast, the inner levels are nearer-term, defining better understood and more committed solution behavior. The following sections describe each planning horizon.

## Daily Plan

The daily plan (*team sync*) is a short meeting (usually 15 minutes or less), typically held daily, to coordinate the team's work, inspect progress toward the iteration goal, communicate, and adjust upcoming planned work.

## Iteration Plan

In SAFe Scrum, [Iteration Planning](#) is a structured event that kicks off each iteration. Teams collaborate to determine how much of the team backlog they can deliver during an upcoming iteration and summarize those [Stories](#) into a set of [Iteration Goals](#) and the iteration backlog. That is their 'plan' and commitment to the ART business.

SAFe Kanban Teams often follow a similar pattern as Scrum, participating in each ART iteration, contributing to System Demos, implementing stories to advance toward their PI Objectives, and collaborating with other teams. However, Kanban teams do not have to plan at iteration boundaries, maintain an iteration backlog, or establish iteration goals. They complete their work within a timebox required to meet business and technical needs and class of service (e.g., fixed date, expedite). Kanban teams ensure they have a sufficient amount of prioritized backlog items to keep them productive and avoid delays. That is their 'plan' and commitment to the ART business.

## PI Plan

The PI plan is the output of the ART's most recent [PI Planning](#) event. During PI planning, Agile Teams get presented with new [Features](#) and plan the stories they need to deliver alongside work from their local context. Teams summarize this work as a set of team PI Objectives. The PI plan also includes the identification of risks and the ART planning board, which illustrates how the features will be delivered over time and highlight any dependencies.

## PI Roadmap

The *PI roadmap* typically consists of a summary of the PI Plan plus 1-2 *forecasted* PIs. The current PI is where the ART committed to the [PI Objectives](#). Since the business and technical context may change, planning subsequent PIs is less precise and defined. ARTs can generally predict with relatively good confidence since they use their historical average velocity.

Product Management generally leaves room for new [Features](#) in the following PIs, scheduling the amount of work to be less than available capacity. Each element on the roadmap is a feature, [Capability](#) (or ART [Epic](#)), intended for a particular PI. The PI roadmap may also reflect fixed-date and learning milestones during that period.

Figure 2 illustrates a roadmap that consists of a summarized *PI Plan* and two *forecasted* PIs. This timeframe is typically sufficient to communicate intent with stakeholders, the business, and partners. It's also short enough to keep commitments from interfering with adjustments to changing business priorities.

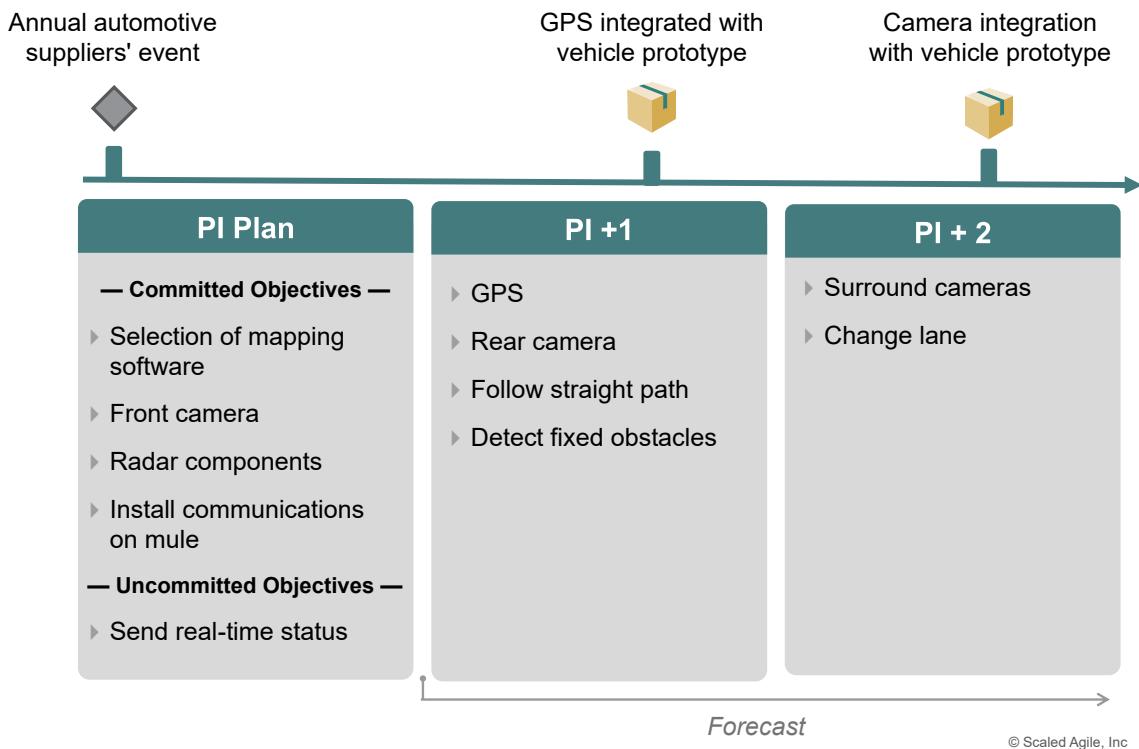


Figure 2. An example of a PI roadmap for an autonomous vehicle

Figure 2 also illustrates that some organizations prefer to create PI roadmaps with less than their available capacity, creating a more opportunistic plan for change. Others plan future PIs with a more complete but riskier schedule. Regardless of how PIs are forecast, items don't become committed until teams plan them during PI planning.

## The Solution Roadmap

The *Solution roadmap* in Figure 3 provides a multiyear view of planned epics and capabilities over time for a specific *solution* and timeframe within the portfolio.

## Solution Roadmap

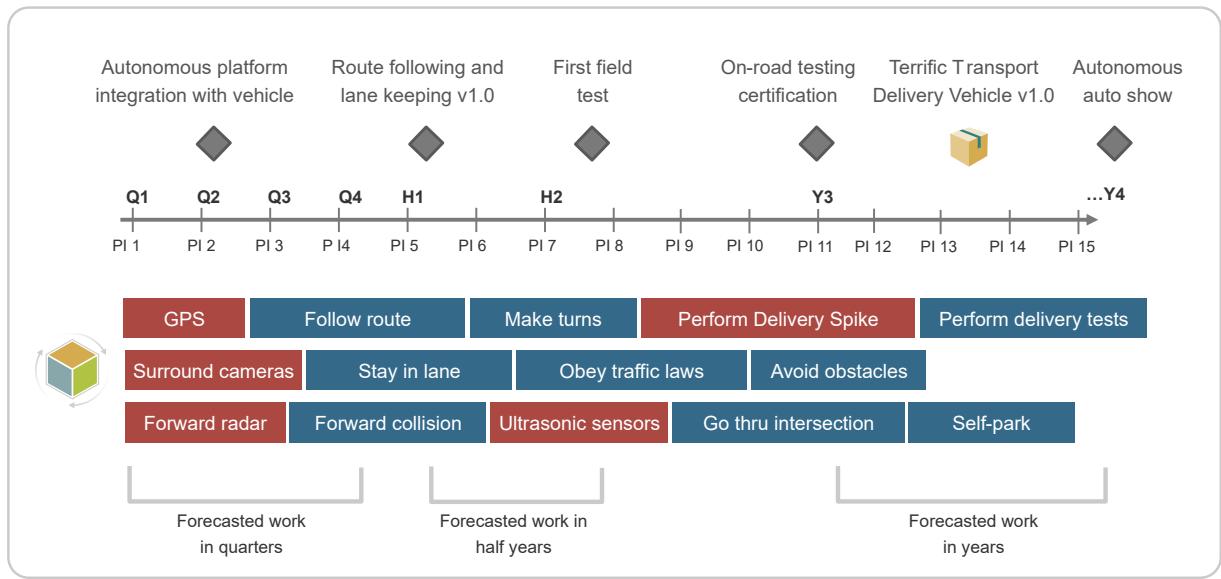


Figure 3. The solution roadmap for an autonomous delivery vehicle

In this example, the roadmap depicts milestones as diamonds, while a solution release is shown with a small box. As the time horizon extends, items are less defined and more uncertain. For example, the first year is planned in quarters (Q1, Q2, Q3, and Q4), which may or may not align with PI boundaries. The second year is shown in six-month increments (H1 and H2). Anything beyond that is scheduled in years (Y3 and Y4) to reflect the significantly higher uncertainty.

## The Portfolio Roadmap

The *Portfolio roadmap* (Figure 4) illustrates the plan of intent for achieving the portfolio vision, primarily with epics, for an extended time.

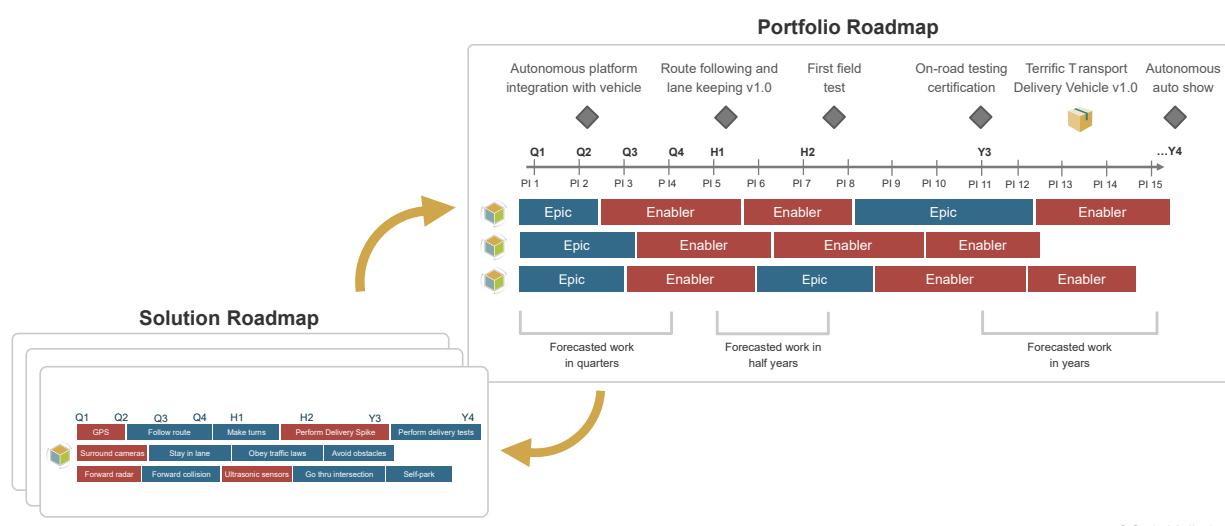


Figure 4. The portfolio roadmap communicates the longer-term vision

The portfolio roadmap integrates the various aspects of solution and PI roadmaps and their milestones into a comprehensive view across all the value streams in the portfolio. This roadmap builds the larger picture, communicating how the portfolio vision will be achieved over a specific timeframe to stakeholders and illustrates a high-level view of [Epics](#) within each value stream.

Since the solution and portfolio roadmap may span multiple years, both require estimating longer-term initiatives. However, every enterprise must be cautious about such forecasts. While many see long-term predictability as a goal, [Lean-Agile Leaders](#) know that *every long-term commitment decreases the agility of the enterprise*. If the future is too fixed, it's challenging to have [Business Agility](#). Therefore, regular forecast updates reflect new learning and changing market conditions.

## Flexible Roadmaps Improve Flow

Customer desires, competitive threats, technology choices, business expectations, revenue opportunities, and workforce demands now happen at *blistering speeds*. Consequently, roadmaps need to respond to change and be flexible.

[Principle #6, Make value flow without interruptions](#), informs us that working in smaller batches reduces queue lengths. Committing to longer-term roadmaps often creates queues. The math in Little's Law [2] tells us that longer queues increase the wait time for any new initiative.

One way to ensure a flexible roadmap is to focus on high-value items and limit longer-term fixed-date commitments. Let's explore two different scenarios to illustrate this situation:

- **Partially loaded PI roadmap** – Figure 2 above shows a PI roadmap with room for new features in the following PI. If an item can't fit in the current PI, it can be scheduled for the next, which is approximately ten weeks, depending on the chosen PI duration.
- **Fully loaded PI Roadmap** – Figure 5 below illustrates a roadmap where all five PIs are fully loaded and committed. In this case, the wait time for a new feature is more than 50 weeks (assuming a ten-week PI duration). Shifting from a traditional to an Agile mindset means keeping commitments as short and flexible as possible.

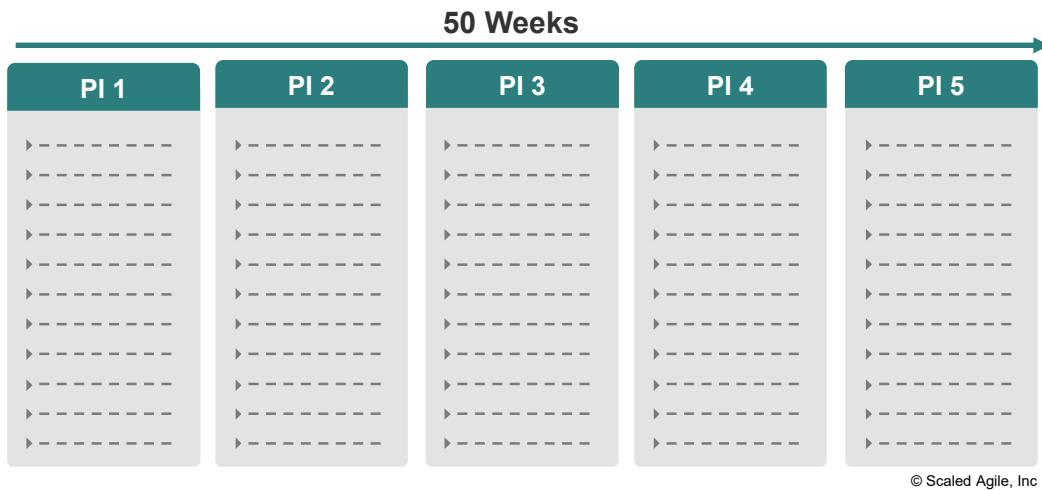


Figure 5. A fully committed roadmap becomes a queue, increasing the wait time

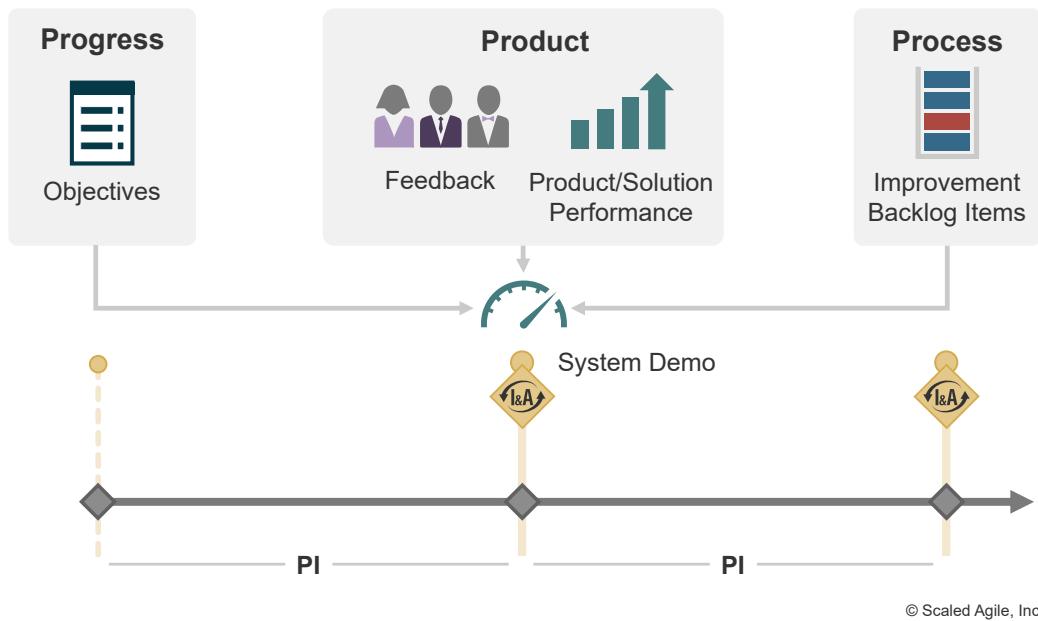
## Milestones Make Critical Events Visible

Internal and external milestones, and significant events, offer critical insights into building roadmaps [3]. They help stakeholders understand feature progress and visualize and leverage opportunities that can be predictable and require longer-term planning, as described in the following sections.

*Milestones* mark specific progress points on the development timeline and are critical for understanding and monitoring product evolution and risk. In the past, many progress milestones were based on phase-gate activities. But experience has shown that stage gate milestones generally do not reduce risk. (See [Principle #5 – Base milestones on objective evaluation of working systems](#))

SAFe progress milestones are represented by the fixed cadence of Iterations and [PIs](#), as illustrated in Figure 6. It defines three types of milestones:

**1. PI milestones** – Each PI creates an objective measure of progress, as shown in Figure 1. These milestones help evaluate progress toward the technical or business hypothesis.



© Scaled Agile, Inc.

Figure 6. PI milestones provide objective evidence

With SAFe's approach, the system is built in increments, each of which is an integration and knowledge point that demonstrates evidence of the solution's viability. Further, these objective evaluations are performed regularly on a PI cadence, which ensures periodic availability and evaluation. It also offers predetermined time boundaries for eliminating less desirable solution options (See [Principle #3, Assume variability; preserve options](#))

**2. Fixed-date milestones** – Every Lean-Agile enterprise wants to operate with the minimum possible constraints, providing more agility. However, fixed-date milestones are typically for traditional and Lean-Agile development. For example, they are required to communicate the following:

- **Events** include trade shows, customer demos, user group meetings, planned product announcements
- **Release dates** for internal or external business concerns
- **Contracts** with binding dates for value delivery, intermediate and payment milestones, demos, and more
- **Scheduling** larger-scale integration, including hardware, software, supplier solutions, and anything else where a fixed date provides an appropriate synchronization to bring together assets and validate them

Fixed-date milestones should be reflected in the relevant roadmaps, so stakeholders can plan and act accordingly.

**3. Learning milestones** – help validate business opportunities and hypotheses, as shown in Figure 7.

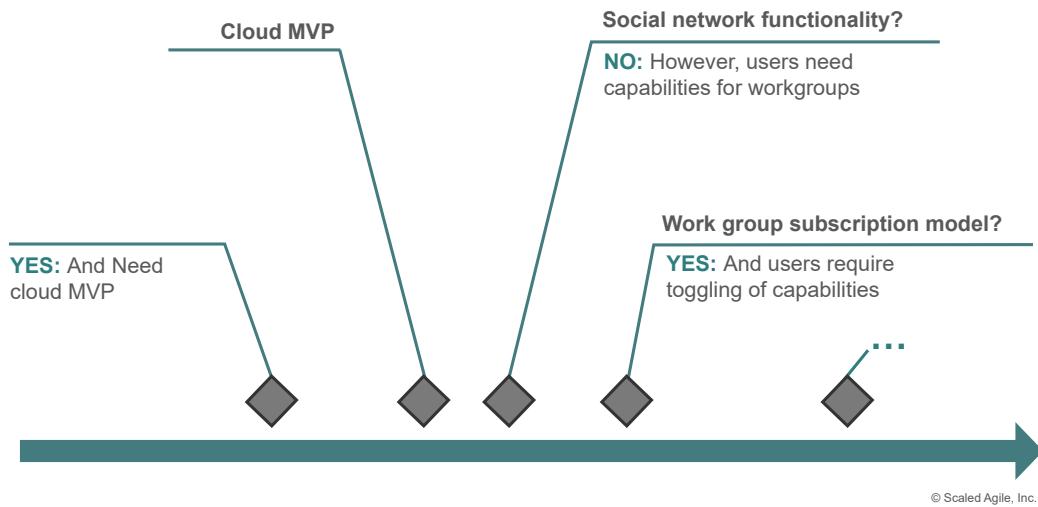


Figure 7. Learning milestones help evaluate progress toward the goal

© Scaled Agile, Inc.

Many other concerns also affect planning and road mapping. Things like patent filing, internal or regulatory audits and certification, user acceptance testing, and more must be identified and made visible on roadmaps.

## Market Rhythms and Market Events Influence Roadmaps

Most enterprises operate in a larger ecosystem of consumers, buyers, purchasing cycles, and supply chains affected by external calendar-driven factors and constraints. Understanding market rhythms and events help companies leverage predictable, calendar-based opportunities and require longer-term planning.

### Market Rhythms

A *market rhythm* is a set of significant events that occur periodically on a predictable cadence. For example, retailers routinely prepare for the holiday shopping season by upgrading their computer systems to get a competitive edge and support significantly higher transaction volumes. In these cases, release dates can dramatically affect the value delivered.

For example, Figure 8 illustrates the market rhythms for three different companies.

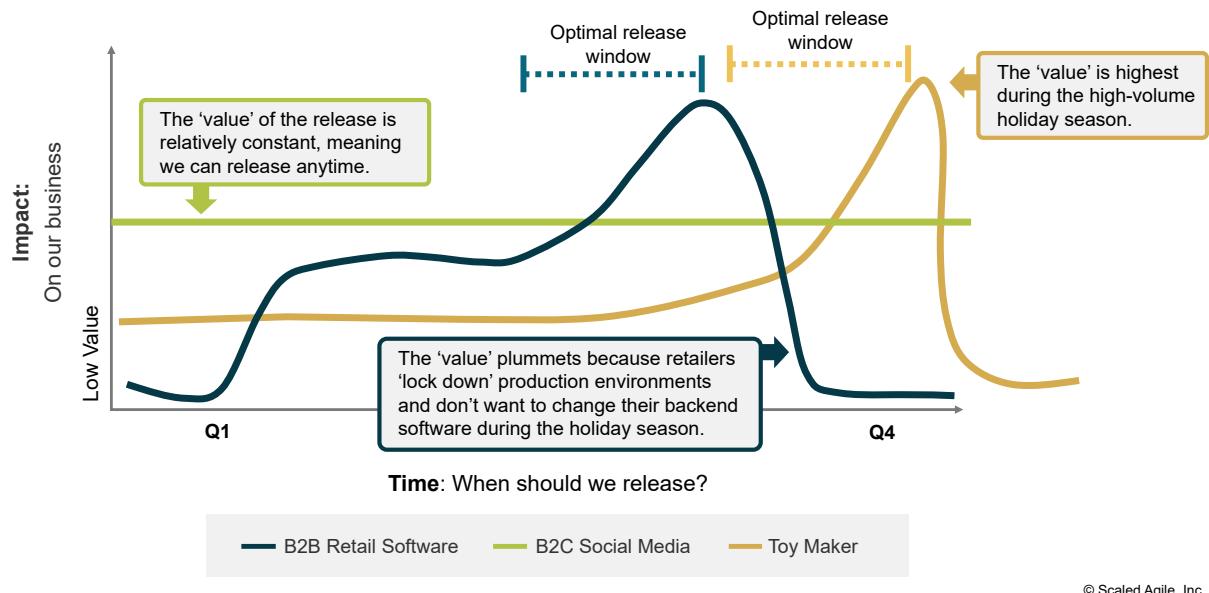


Figure 8. Example market rhythms for different companies

The vertical axis in Figure 8. shows the value delivered to a market, while the horizontal axis shows the value over time, usually a calendar or fiscal year.

1. **B2B retail software** – the blue line shows a similar market rhythm as the toy maker. However, this organization must release its solutions even sooner.
2. **B2C social media** – the green, flat horizontal line represents a company where the value over time is relatively constant, suggesting that market rhythms do not strongly influence it.
3. **Toy maker** – the orange line illustrates an organization that must have its products ready for sale before the annual holiday shopping season.

The *toy maker* and *B2B retail software* companies make most of their sales during the holiday season. In the extreme case, a product or service may have zero value if that market window is missed. This wide range of potential value correlated to the calendar can substantially impact business outcomes. Agile or not, planning is a critical activity.

## Market Events

A *market event* is a *one-time* future event with a high probability of materially affecting the value of one or more solutions. Market events can be external, such as the launch of government regulations, or they can be internally created, such as a company's annual user conference. Market events (Figure 9) are typically represented on the roadmap as milestones. They may strongly impact specific solution releases. Consequently, it may cause adjustments to the timing of features or solution development activities, often identified during [PI Planning](#).

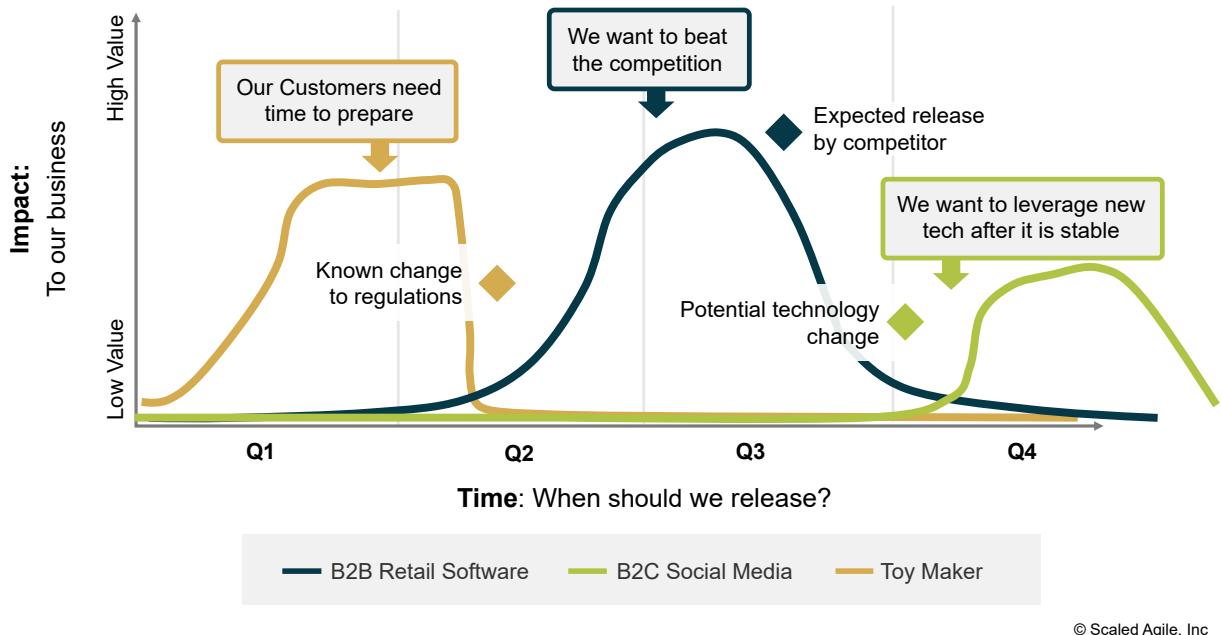


Figure 9. Example of market events

## Learn More

[1] *Manifesto for Agile Software Development*. <http://AgileManifesto.org>

[2] Little, J. D. C. (1961). "A Proof for the Queuing Formula:  $L = \lambda W$ ". *Operations Research*. 9 (3): 383–387. doi:10.1287/opre.9.3.383. JSTOR 167570.

[3] Hohmann, Luke. *Beyond Software Architecture: Creating and Sustaining Winning Solutions*. Addison-Wesley Professional, 2003.

Oosterwal, Dantar P. *The Lean Machine: How Harley-Davidson Drove Top-Line Growth and Profitability with Revolutionary Lean Product Development*. Amacom, 2010.

Last update: 6 May 2024

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

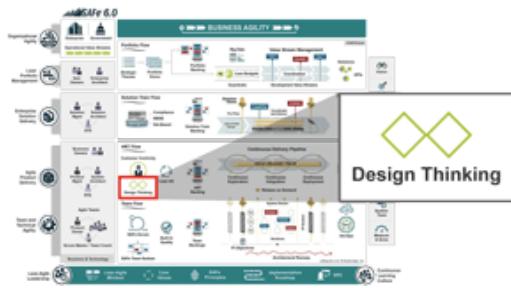
[Remote Training Policy](#)





+ Framework

ENGLISH (US) ▾



“Good design is actually a lot harder to notice than poor design, in part because good designs fit our needs so well that the design is invisible, serving us without drawing attention to itself. Bad design, on the other hand, screams out its inadequacies, making itself very noticeable.

—Don Norman, *The Design of Everyday Things*

## Design Thinking

Design Thinking is a customer-centric development process that creates desirable products that are profitable and sustainable over their lifecycle.

It goes beyond the traditional focus on the features and functions of a proposed product. Instead, it emphasizes understanding the problem to be solved, the context in which the solution will be used, and the evolution of that solution.



**Note:** This article focuses on the tools and practices associated with implementing design thinking. It should be read along with the [Customer Centricity](#) article, which focuses on the mindset and impact of customer centricity.

# Details

Traditional waterfall approaches to product development are sequential: requirements are defined, and solutions are designed, built, and delivered to the market. The focus tends to be on the most apparent problems. Often, success is determined by implementing a solution that meets the *requirements* instead of the *user's needs*. This results in products and services with unusable or ignored features that frustrate users and fail to meet the enterprise's business goals.

Design thinking (Figure 1) represents a profoundly different approach to product and [Solution](#) development, in which divergent and convergent techniques are applied to understand a problem, design a solution, and deliver that solution to the market.

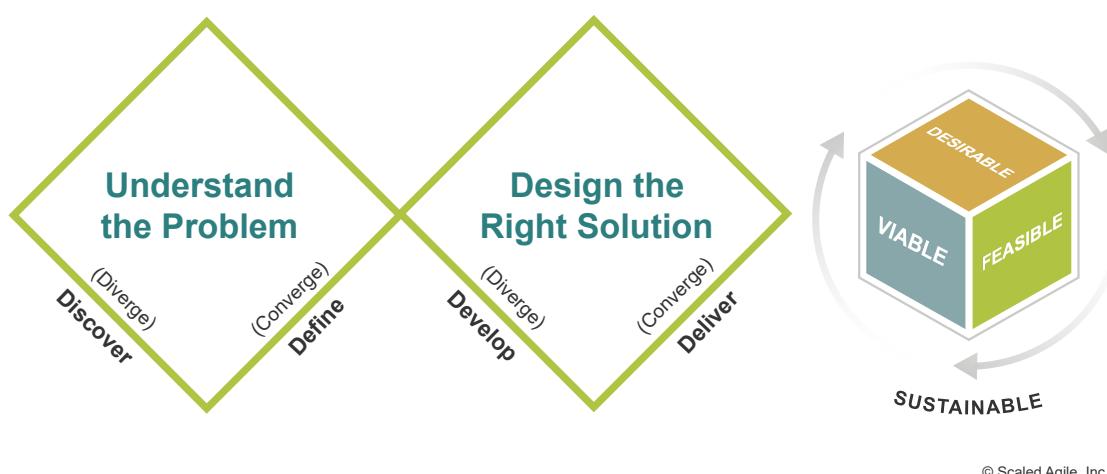


Figure 1. Design thinking activities

Design thinking also inspires new ways to measure the success of our efforts:

- **Desirable** – Do customers and end-users want the solution?
- **Feasible** – Can we deliver the right solution through a combination of build, buy, partner, or acquire activities?
- **Viable** – Is the way we build and offer the solution creating more value than cost? For example, in a for-profit enterprise, are we profitable?
- **Sustainable** – Are we proactively managing our solution to account for its expected product-market lifecycle? Are we delivering tangible economic, social, and environmental benefits throughout the product lifecycle?

Successive applications of design thinking advance the solution over its natural market lifecycle, as shown in Figure 2.

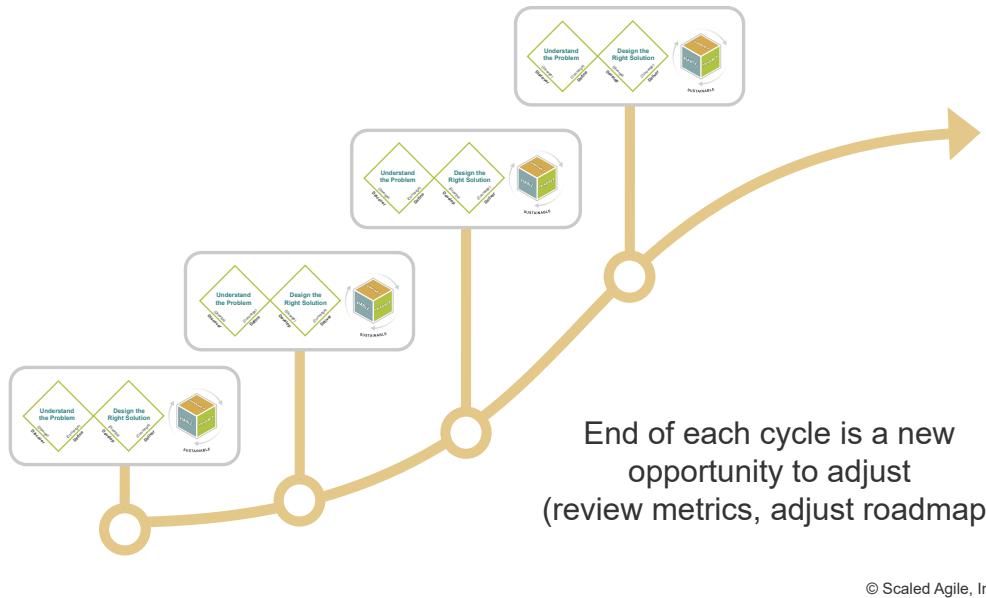


Figure 2. Advancing the solution over the lifecycle through design thinking

© Scaled Agile, Inc.

## Understanding the Problem and Solution Space

In Figure 1, the core design thinking processes appear as a 'double diamond.' This represents the focus on thoroughly exploring the problem space before creating solutions. Each diamond focuses on divergent thinking (understanding and exploring options) followed by convergent thinking (evaluating options and making choices).

The activities associated with exploring the problem are elaborated as follows:

- **Discover** – The discover phase seeks to *understand the problem* by engaging in market and user research to identify unmet needs. This research creates fresh perspectives that drive innovation. Unlike research that confirms or refutes a hypothesis, the inquiries associated with the discovery phase occur without preconceived notions about how users *should* work. Instead, it focuses on how users *work*. An essential research technique is *Gemba*, also known as 'going to the place where the work is done.'
- **Define** – The define phase focuses on the information gathered during the discover phase to generate insights into specific problems and unmet needs. These create opportunities for the business and new product development. Results of this phase typically include personas and empathy maps (described below) that focus the product team on the solutions the *Customer* would view as desirable. *Epics* and *Features* capture the perceived changes needed for existing products and solutions.

With a clear understanding of the target market and its problems, the focus can move toward designing a solution, the second diamond of design thinking. These are:

- **Develop** – The develop phase uses journey mapping, story mapping, and prototyping to design potential solutions to problems quickly and cost-effectively. Each of these

techniques is discussed more thoroughly later in this article. The develop phase also embraces [SAFe Principle #3 – Assume variability; preserve options](#). Design thinking techniques preserve options responsibly.

- **Deliver** – The deliver phase produces artifacts suitable for creating the solution and varies based on context. These artifacts often start as prototypes expressed as validated features in the ART Backlog for continuous delivery.

## Using Personas to Focus Design

Creating solutions for a *direct* customer—bespoke solutions—offer designers the advantage of speaking directly and frequently with a few target users, permitting them to participate in the design, [PI Planning](#), [System Demos](#), and other SAFe events. In some organizations, Customers are considered part of the team, so creating a Persona to represent them isn't typically needed but may be helpful when the organization is highly distributed.

In contrast, in an *indirect* customer market, which is common in B2C solutions, product teams need a way to maintain a connection with their target customers. So, they develop ‘personas,’ fictional consumers and users derived from user research. [2] They depict the people who might similarly use a product or solution, providing insights into how real users would engage with a solution. User personas also support market segmentation strategy by offering a concrete design tool to reinforce that products and solutions are created for people. Personas drive product development and several SAFe practices, as shown in Figure 3.

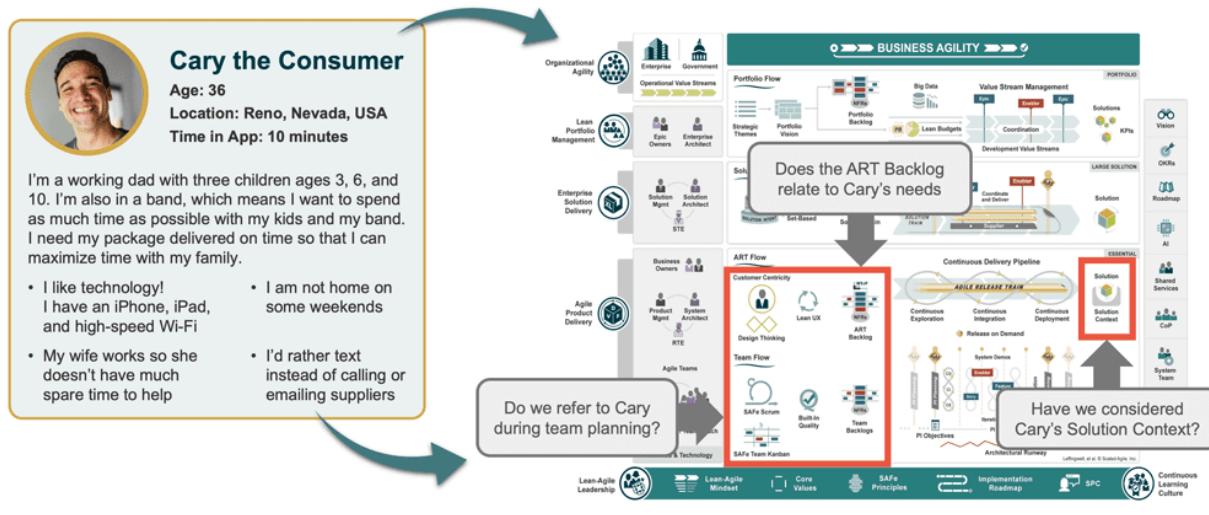


Figure 3. How Personas can drive key activities in SAFe

In addition to user personas, *buyer* personas extend design thinking to include the individuals and organizations that authorize purchasing decisions. They help ensure that the design encompasses the whole product purchase experience, including after-sales service, support, and operations.

# Establishing Empathy to Foster Customer-Centric Design

Customer-centric enterprises use empathy throughout the design process. Empathetic design dismisses preconceived ideas and uses the Customer's perspective to inform solution development.

Empathy maps [1] are a design thinking tool that promotes customer identification by helping teams develop a deep, shared understanding of others (Figure 4). They enable teams to imagine what a specific persona is thinking, feeling, hearing, and seeing as they use the product. The greater the degree of empathy a team has for its customers, the more likely it will be able to design desirable solutions.



Figure 4. Empathy map

## Designing the User Experience through Journey Maps

A *customer journey map* illustrates the user experience in an [Operational Value Stream](#) that provides products and services. Figure 5 shows how these journey maps are powerful design thinking tools. They allow teams to identify ways the specific deliverables for one or more [Development Value Streams](#) can be improved to create a better end-to-end user experience.



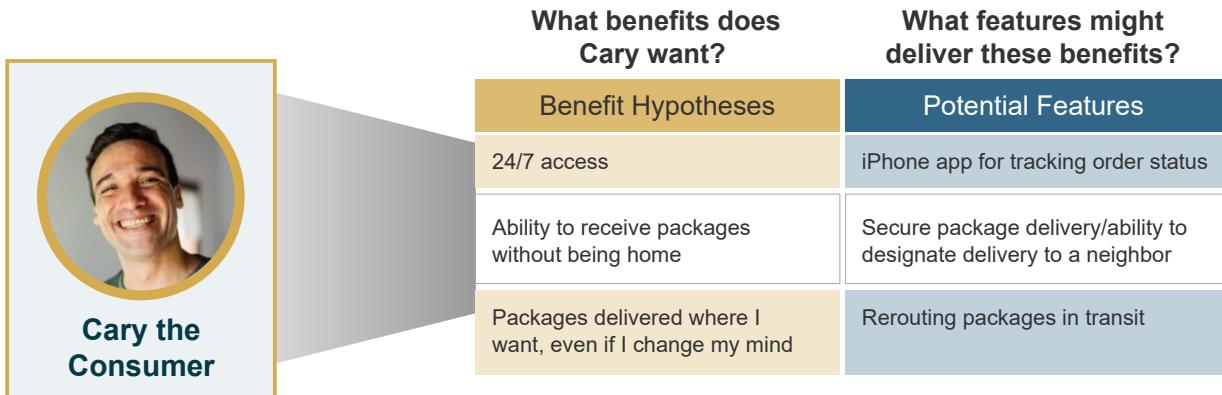
	<b>DECIDE</b> Access options	<b>LEARN</b> Clarify goals		<b>CHOOSE</b> Choose what to buy		<b>APPLY</b> ID & financial check		<b>PURCHASE</b> Negotiate sale	<b>COMPLETE</b> Celebrate
Young Couple	Decide to buy a home	Learn about the mortgage	Calculate the budget	Property search	Compare mortgages	Apply for mortgage loan	Assessment	Sign contract & deal	Move
Thinking & Feeling	We loved our neighborhood growing up and want to raise our kids the same way	A real estate agency will save money		Our real estate agent helped us clarify our goals	Just something that will pass quickly	The bank officer will handle everything	We're happy the bank officer kept us updated		
	We've never made a purchase this large before – it is scary!	Visiting bank for consultation = helpful		Selecting real estate agency – comparing & selecting property	Why does the bank need paper copies?		Negotiating a sale this big makes us nervous	We need to clean	
		Can we really find a home that we love that we can afford?		Form filing is time consuming & the bank should know our details already	Can't this all be online?		It took a lot longer to close the sale than expected		
Bank		Customer Service Officer • Explains products • Prepares offer • Advises to select real estate agency		Customer Service Officer • Fills in application form & copies ID cards • Provides list with required documents	Credit Department • Performs assessment of applicants • Evaluates property • Prepares approval		Customer Service Officer • Provides updates on loan status		
Technology Opportunities	Online resources for understanding mortgage options	Online advisory, budgeting, financial management, and purchasing options		Automatically gather data from public databases and integrate with social media data	AI-based assessments could improve speed and accuracy		Biometric authentication of electronic signatures	Automated referrals to partners	

© Scaled Agile, Inc.

Figure 5. Operational value streams and customer journey maps

## Delivering Benefits Through Features

While a journey map captures the high-level experience of the Customer in the operational value stream, product features manage the specific deliverables that fulfill a stakeholder's need. Features are commonly described through a *features and benefits (FAB) matrix*, using short phrases that provide context and a benefit hypothesis. Design thinking, however, promotes switching the order of the FAB to a *benefit-feature matrix*. In this case, the intended customer benefits are identified first, and then the teams determine what features might satisfy their needs. This approach helps [Agile Teams](#) explore better and faster ways to deliver the desired benefits (Figure 6).



© Scaled Agile, Inc.

Figure 6. Design thinking promotes switching the order of the FAB to a *benefit-feature matrix*

## Designing User Workflows or Journeys through Story Maps

Features that capture a workflow or user journey present a unique challenge to Agile teams. Because the backlog is a flat, one-dimensional list, it does not show the relationship between the user's goals, workflow activities, and the stories in the backlog. Story mapping is a brainstorming technique that can enable teams to design a solution focused on the Customer. Not all features will require story mapping. However, they are particularly useful for developing new end-user functionality for a workflow or customer journey.

## Why Story Maps?

Story maps help teams ideate, plan, and group activities in a *workflow or user journey*. They allow teams to address the most critical steps before improving existing steps or adding new functionality. Story maps are an important design thinking tool that enables [Customer Centricity](#) because they focus on delighting a user instead of merely implementing stories ordered by their value. Another benefit is avoiding releasing a feature (or solution) that is not usable because its functionality depends on stories that are lower in priority and further down the backlog.

Figure 7 illustrates how a feature with a workflow is captured in a story map [3], organizing the sequence of stories according to the activities (or steps) a user needs to accomplish their goal. The first set of stories is essential for the initial release, while the next set represents improvements for future releases.

## How to Create a Story Map

The following steps describe the process of creating a story map (Figure 7) for a new potential Feature that requires a workflow.

1. **Frame the purpose:** Identify the goal or customer problems the solution will solve and the intended users of the solution.
2. **Map the whole story:** Define the starting conditions for the user to accomplish their goals. Focus on describing the whole story and user activities and tasks, creating the backbone of the story map.
3. **Brainstorm:** Fill in the body of the story map by breaking down the larger user tasks into smaller subtasks and user interface details. Consider many possibilities without concern if the stories are in or out of scope. Affinity group the stories needed to complete the task under each activity.
4. **Identify the stories essential for the initial release:** The team identifies which stories can be released (in the next iteration or two) that will achieve a meaningful user outcome.
5. **Identify stories considered as improvements in future releases:** Stories that are not selected for the initial release will be added to the backlog as potential candidates for future releases.

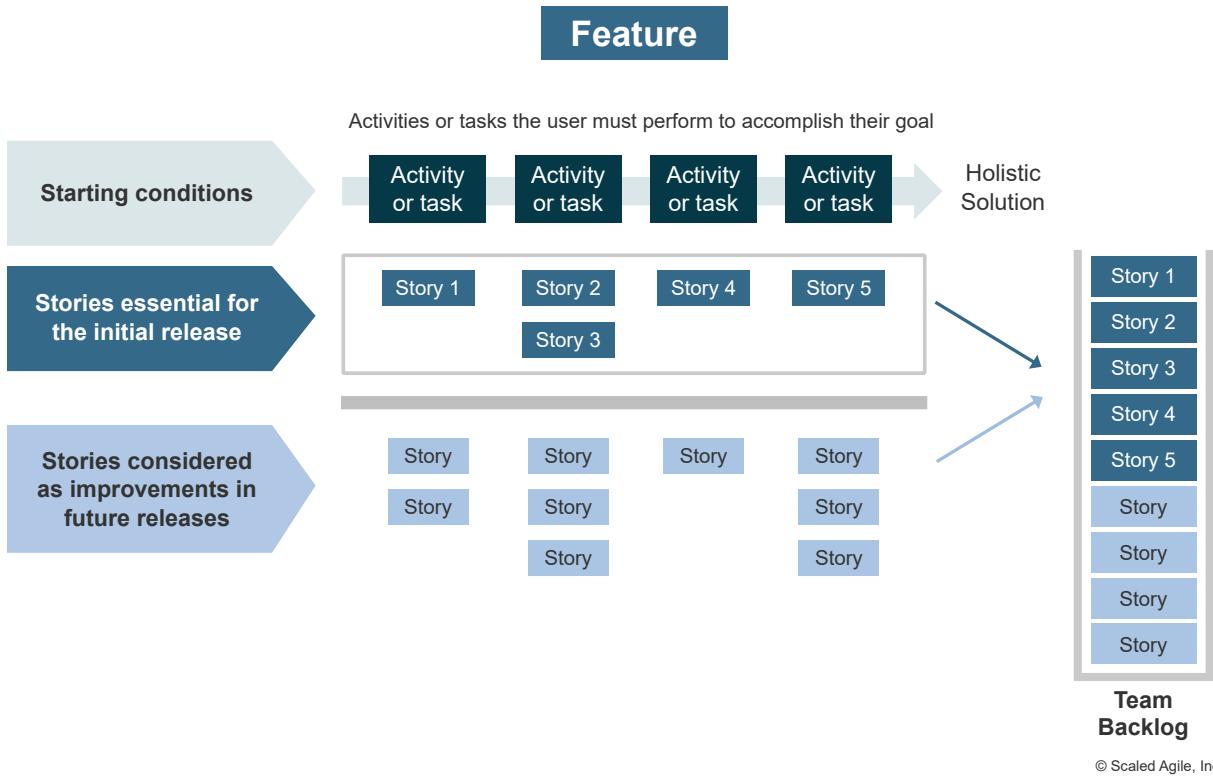


Figure 7. Story map

© Scaled Agile, Inc.

## Increasing Design Feedback Through Prototypes

A prototype is a basic functional model of a feature or product, usually built for demonstration purposes or as part of the development process. It helps the team clarify their understanding of the problem and reduces risk in designing and developing the solution before making further investments. Prototypes provide many benefits:

- **Fast feedback.** By definition, a prototype is cheaper and faster to produce than a complete solution. This enables faster feedback from users and customers, increased understanding of solution requirements, and greater confidence in the final designs.
- **Risk reduction.** Prototypes can reduce technical risk by enabling Agile teams to focus initial efforts on the aspects of the solution associated with the highest risk.
- **Intellectual property/patent filing.** Prototypes can be used to satisfy strategic requirements for managing intellectual property as early as possible in the development process.
- **Models for requirements.** Prototypes can provide more clarity in the requirements of the desired feature or solution than pages of documentation.

There are many kinds of prototypes, each optimized to provide different types of insights:

- **Paper prototypes** are typically hand-drawn sketches of the intended solution. They can be automated to illustrate workflows or validate user story maps.

- **Mid-Fi prototypes** are visually-complete representations of software-centric solutions but are not typically functionally integrated.
- **Hi-Fi prototypes** are visually-complete and interactive models which users and customers can directly explore.
- **Hardware prototypes** provide critical feedback on form factors, sizes, and operational requirements. For example, when exploring form factors to see how a new tablet might fit into existing backpacks, briefcases, and cars, one Silicon Valley company cut many plastic models from a single sheet of plastic. Later in this design process, this same team found they needed to redesign the power supply so that it would not unduly interfere with the WiFi signal.

Product teams should strive to leverage the lowest-cost, fastest form of prototyping to gain actionable feedback. Often, paper prototyping is the best choice. [4] [5]

---

## Learn More

[1] Empathy Map Canvas. <https://medium.com/the-xplane-collection/updated-empathy-map-canvas-46df22df3c8a>

[2] Cooper, Alan, Robert Reimann, David Cronin, and Christopher Noessel. *About Face: The Essentials of Interaction Design 4th Edition*. Wiley, 2014.

[3] Patton, Jeff, and Peter Economy. *User Story Mapping: Discover the Whole Story, Build the Right Product* 1st Edition. O'Reilly Media, 2014.

[4] Snyder, Carolyn. *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces*. Morgan Kaufmann, 2003.

[5] Gothelf, Jeff, and Josh Seiden. *Lean UX: Designing Great Products with Agile Teams*. O'Reilly Media, 2016.

Last updated: 13 February 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)





“ *The most important single thing is to focus obsessively on the customer. Our goal is to be earth's most customer-centric company.*

—Jeff Bezos

## Customer Centricity

Customers are the ultimate beneficiaries of the value of the solutions created and maintained by a portfolio's value streams.

Customer Centricity is a mindset that focuses on creating positive experiences for the customer through the full set of products and services that the enterprise offers.

Customer-centric organizations deliver whole-product solutions designed with a deep understanding of customer needs. This results in greater profits, increased employee engagement, and more satisfied customers in the private sector. Nonprofits and the public sector (governments) can achieve the resiliency, sustainability, and alignment needed to fulfill their mission.



**Note:** This article describes the *mindset* and impact of customer centricity. The related [Design Thinking](#) article provides the *tools and practices* to support creating desirable products that are profitable and sustainable over their lifecycle. Therefore, it's recommended that these two articles are read together.

# Details

Customer centricity is a mindset that helps organizations make decisions that are based on a deep understanding of its effect on customers and end-users that motivates the following behaviors:

- **Focusing on the customer** – aligning and focusing the organization on specific, targeted user segments
- **Understanding the customer's needs** – moving beyond merely listening to customers who ask for features and investing the time to identify the customer's fundamental and ongoing needs
- **Thinking and feeling like the customer** – striving to see the world from their customer's point of view
- **Building whole product solutions** – designing a complete solution for the user's needs, and ensuring that the initial and long-term customer experience is continually evolving toward the ideal solution
- **Knowing customer lifetime value** – moving beyond a transactional mentality, and focusing on creating longer-term relationships based on a clear understanding of how the customer gains value

## Driving Research

The foundation of the customer-centric enterprise is market and user research that creates actionable insights into the problems customers face, the [Solution Context](#), and requirements. Market research helps drive *strategy*, while user research drives *design*, as shown in Figure 1.

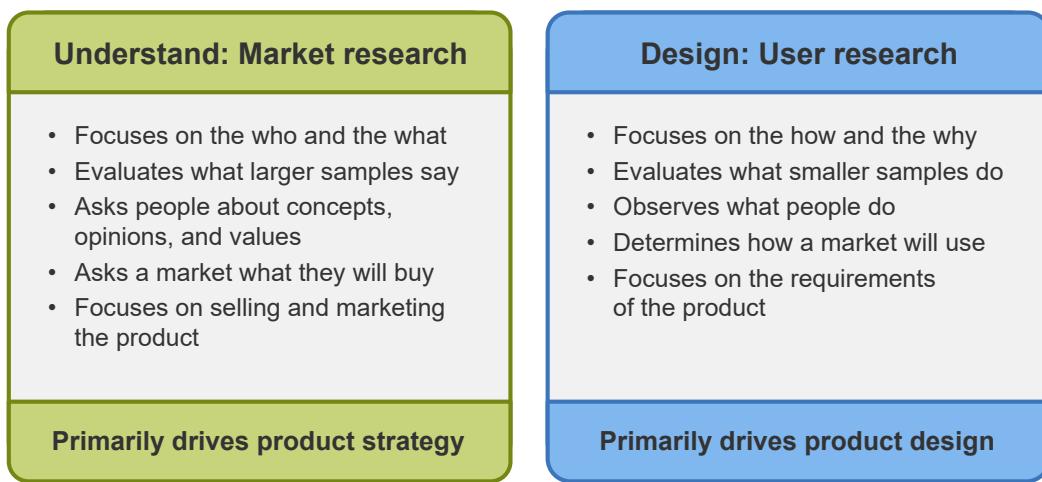


Figure 1. Market and user research explore different aspects of the problem and solution space

Ongoing research activities are supported through [Continuous Exploration](#), automated data collection, and the feedback loops between the solution and its solution context.

# Designing with Empathy

Designing with empathy puts aside preconceived ideas and helps create solutions from the customers' perspective. [1] It motivates teams to understand and experience the world from the customer's viewpoint, learning and appreciating the difficulties they face, their roles, and their context. It emphasizes user research, including activities such as Gemba walks (for example, going to the customer's workplace). Gemba helps [Agile Teams](#) build empathy better to understand the user's emotional and physical needs. The way they see, understand and interact with the world around them.

Customer-centric enterprises apply *empathic design* throughout the product lifecycle, guiding the development of solutions that move beyond functional needs and address:

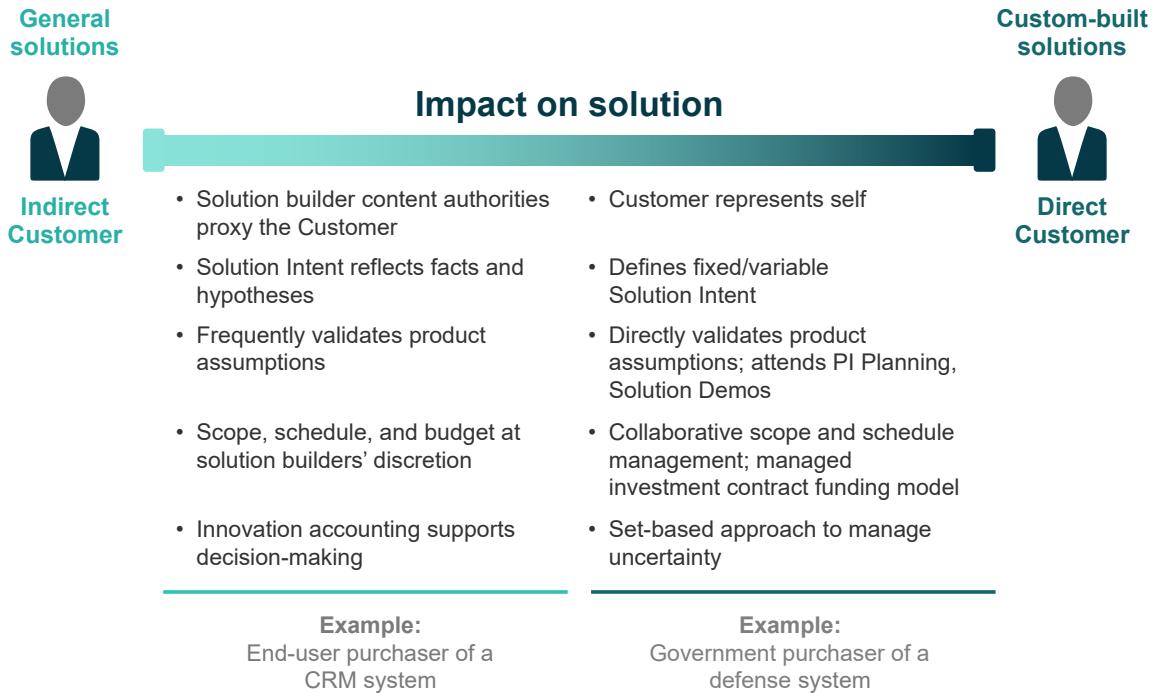
- Aesthetic and emotional needs
- Ergonomic requirements, such as the placement of physical features
- Product attributes that users may not explicitly request, such as performance, security, and compliance, but which are essential for viability
- An understanding of how the solution may impact the solution context
- The impact of the solution on related or affected groups
- The solution's architecture ensures operations, maintenance, and support of customer needs

# Understanding Customer Engagement

Market research helps determine the nature of customer relationships, which is largely determined by the type of solution:

- **General solutions** – intended for a broad segment of customers
- **Custom-built solutions** – built and designed for a specific customer

Figure 2 illustrates the level of indirect or direct customer engagement in each case



© Scaled Agile, Inc.

Figure 2. Customer engagement models in general and custom-built solutions

## General Solutions

General solutions must address the needs of a broader market or segment in which no single customer adequately represents the whole market. In this case, Product and Solution Management become the indirect customer proxy; they have authority over solution content. It's their responsibility to facilitate external interaction and ensure teams will hear the 'voice of the customer' and that the organization will continuously validate new ideas. Scope, schedule, and budget for development are generally at the discretion of the internal Business Owners.

Since customers are unlikely to regularly participate in planning and [System Demo](#) events, their interaction is often based on requirements workshops, focus groups, usability testing, and limited beta releases. The solution evolves through feedback from user behavior analysis, metrics, and business intelligence to validate various hypotheses.

## Custom-Built Solutions

External customers collaborate with Product and Solution Management for custom-built solutions in joint design efforts. While the customer is responsible for what gets built, implementation is the responsibility of the solution builder. Deliverables, sequencing, timing, and other development aspects require cooperation, coordination, and negotiation. This collaboration promotes incremental learning and creates opportunities to adjust plans based on the best available data.

SAFe's focus on cadence-based development directly supports the collaborations that create the best outcomes in custom-built solutions. For example, PI Planning provides the time and space to

align all stakeholders around the next set of deliverables. The successful completion of [PI Objectives](#) establishes a high degree of trust in the collaborative development process and generates data that improves forecasting and economic modeling.

## Deep and Narrow Solutions

Deep and narrow solutions are the middle ground between general and custom solutions. These solutions have a small number of customers who will often pay a significant amount of money for these products and services. For example, a solution to manage logistics for NFL stadiums with more than 50,000 people will serve a potential market of fewer than 32 customers. While maintaining the discipline of creating a single solution that answers a target market's needs, [Product](#) and [Solution](#) Management must leverage their familiarity with the small number of customers they're serving.

## Multi-Segment Solutions

Some solutions serve different markets, each using the solution slightly differently. In this situation, customer-centricity requires understanding the unique needs of each segment. The following examples highlight the need for multi-segment solutions:

- **B2C** – A B2C software company serving hundreds of thousands to millions of indirect customers via a website may also offer a set of developer APIs to partners
- **B2B** – Members of a B2B partner segment may act more like customers of custom-built solutions, each making specific requests of the software provider to adjust, extend, or improve the API to meet their unique needs better

## Whole Product Thinking

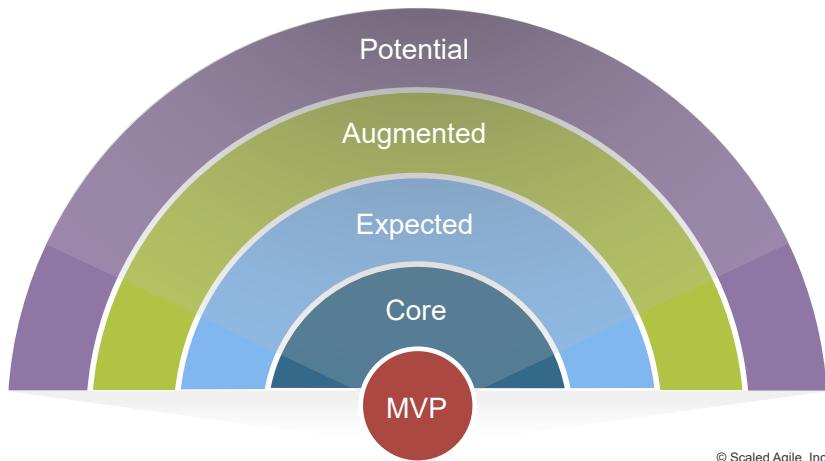
Just as [Systems Thinking](#) takes a holistic approach to solution development, *whole product thinking* involves viewing a product as more than just a sum of its features, but rather as everything involved with the experience customers have with the purchase, use, and support of the product.

Companies must deliver the core product at a minimum. However, customers will not be delighted if it's missing critical aspects of the ownership experience. For example, suppose you buy a smartphone that doesn't have a headphone jack (or Bluetooth), or you must buy a separate charger, battery, or SIM card to operate the phone, customers will not be happy. Similarly, your customers will be dissatisfied if you have an excellent phone with no warranty or customer support or if it doesn't connect to wifi. "You can have the most amazing product in the world, but if you don't pay attention to the Whole Product Concept, it may fail." [1]

Developing a complete product requires starting with an understanding of the real benefits that customers can expect to receive or experience from the product. After that, consider all aspects of the customer's experience, from purchase, first use, ongoing user experience, maintenance, add-

ons, and accessories—even to how the product may be upgraded and supported. Understand all the touchpoints the customer will have with the product, such as setting up the phone with the preferred cellular carrier or the home wifi.

Customers are only satisfied if the product's actual value is the same or *exceeds* their perceived value. Kotler, Levitt, and Moore devised variations of a whole product model that recognizes the levels of customer needs. Figure 3 provides a version of this model that has been adapted for SAFe, followed by a brief description of each level.



© Scaled Agile, Inc.

Figure 3. Whole Product Model [2]

1. **MVP** – is an early and minimal version of a new product used to prove or disprove the benefit hypothesis. The MVP helps answer the question: do customers and users want the solution, and can it be built? The SAFe MVP is an actual product that real customers can use and allows the enterprise to generate validated learning to determine the core product features and beyond. As indicated in Figure 3. the MVP is not a whole product but rather a starting point that proves the hypothesis.
2. **Core product** – addresses the basic functional needs of the customer. It's adequate to accomplish the jobs to be done minimally but lacks certain features or attributes of a product that the customer would expect.
3. **Expected product** – provides the attributes buyers usually expect and agree to when purchasing a product. The expected product includes the core product and other benefits the customer expects when purchasing it. For example, the product has online help, documentation, customer support during the warranty period, and so on.
4. **Augmented product** – provides the additional features, benefits, attributes, or related services that differentiate the product from its competitors to delight customers. For example, a laptop comes with free third-party add-ons, such as a password manager, VPN, touch screen, and more.
5. **Potential product** – envisions the features and other attributes necessary to attract and retain customers indefinitely. Informed by market and user research, the potential product

fuels longer-term strategic planning and creates opportunities for sustainable product advantages.

## Leveraging Market Rhythms and Events

The [Lean-Agile Mindset](#) drives the continuous and sustainable flow of value to customers, motivating organizations to understand how the timing of specific releases influences their perceived value. In other words, the *value* of a product to customers and the organization can vary significantly based on the timing of its release. To create the highest value for all stakeholders, customer-centric organizations leverage market rhythms and market events: [3]

- A **market rhythm** is a set of events *repeatedly occurring* on a *predictable cadence*. For example, retailers routinely prepare for the holiday shopping season by upgrading their systems to gain a competitive edge to support significantly higher transaction volumes.
- A **market event** is a *one-time* future event with a high probability of materially affecting one or more solutions. They can be external, such as the launch of government regulations, or internally created, such as a company's annual user conference.

### Market Rhythms

Market rhythms help companies recognize and capitalize on opportunities that are predictable and require longer-term planning. Figure 4 illustrates an example of the market rhythms of three different companies.

1. A **B2B retail software company** that offers real-time pricing updates must issue important alerts well before the shopping season. It must also update every point of sale terminal in 400 different stores and train all employees on the software's new capabilities.
2. A **B2C social media** company where the value over time is relatively constant suggests it is less affected by market rhythms [3].
3. A **toy maker** must deliver a 'hot new toy' in time for the holiday shopping season, or its price and value will drop significantly!

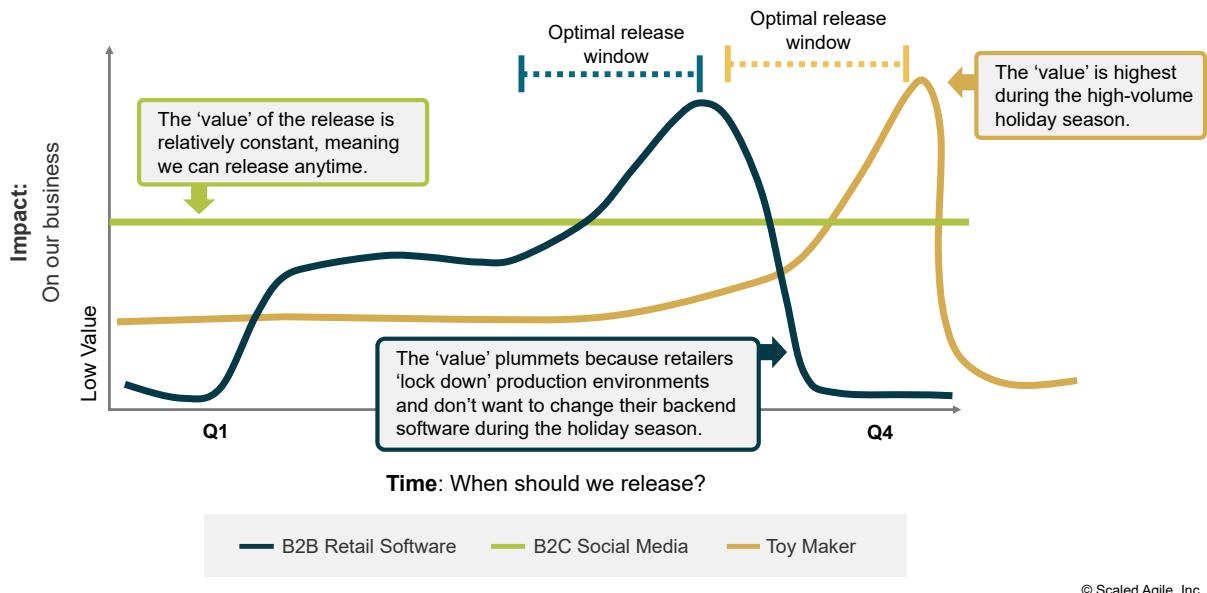


Figure 4. An example of market rhythms for three different types of companies

## Market Events

With an understanding of market rhythms, customer-centric road-mapping activities typically focus on the impact of market events. Figure 5 illustrates three market events, highlighted by the diamond-shaped milestones:

1. Known changes to regulations
2. Expected release by a competitor
3. Potential technology change

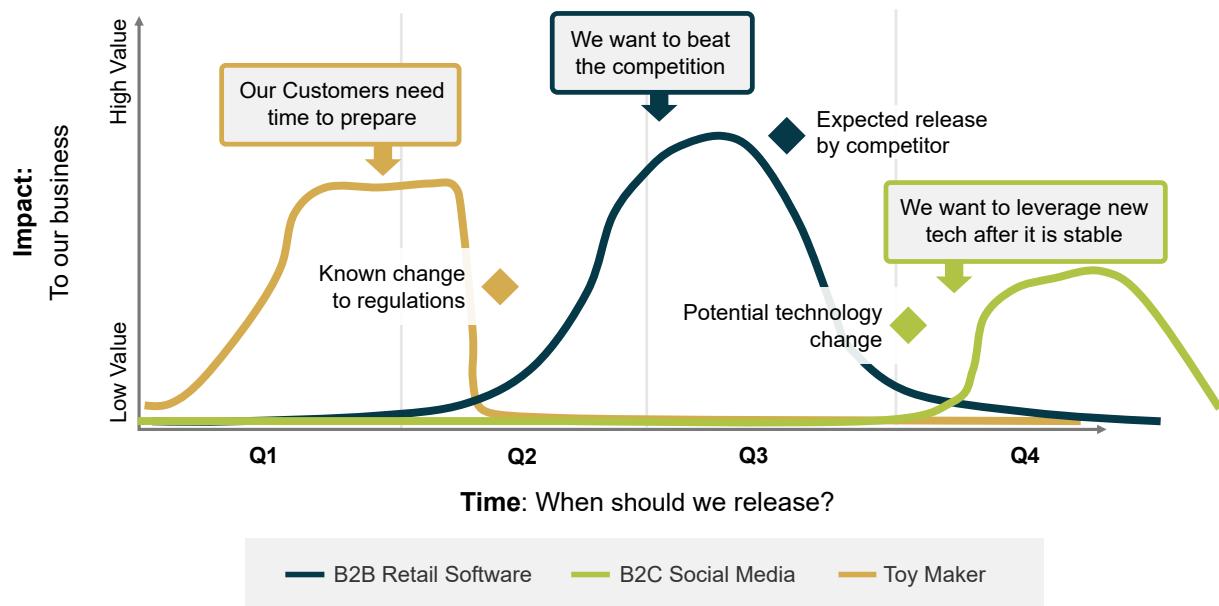


Figure 5. An example of market events

Market events are typically represented as milestones and strongly impact the timing for releasing solutions. They may also inform the content and timing of features or solution development activities identified during PI planning.

## Understanding the Solution Context

Insights gained from the Gemba walks and other research activities define the functional and operational requirements of the solution's operating environment. In SAFe, this is known as the [Solution Context](#), which captures environmental, installation, operation, and support needs.

Understanding the solution context is critical to value delivery. It identifies constraints outside the organization's control. For example, a self-driving vehicle must drive and navigate icy roads while complying with motorist regulations. In another situation, the solution context may describe negotiated constraints, such as when the organization uses principles of set-based design and collaborates with one or more [Suppliers](#) to optimize the total system's space, power requirements, and weight.

Accordingly, some aspects of the Solution Context are fixed, and some are negotiable; this creates a level of coupling between the Solution, Suppliers, and the Solution Context. The mandate of Business Agility motivates Product and Solution Managers to seek optimal solutions, including changing the Solution Context to encourage innovation.

## Understanding Customer Value

Creating viable and sustainable offerings requires a deep understanding of the customer's perception of value. Consider a for-profit enterprise that has identified a customer problem that will cost 800K. If the customer perceives less than 800K of value from the solution, the organization will be unable to sell it at a price that creates a viable offering. And even if the customer perceives more than 800K of value, suggesting the enterprise can make a profit, the solution may not be sustainable if the revenue is insufficient to fund new and ongoing work.

Figure 6 illustrates two primary ways customers derive value from products and solutions:

Cost Reduction	Revenue Enhancement
Less expensive	Accelerates time-to-market
Lower operational costs	Access to new markets
Streamline workflows	New product offerings
Reduce labor costs	Opportunities for service revenue
Reduce compliance costs	

© Scaled Agile, Inc.

Figure 6. Elements of customer value

There are several other aspects of value—for example, brand value and aligning the organization's values and beliefs with customers [3].

## Understanding Internal vs. External Customers

Customers determine the value of every solution and are, thus, an integral part of the Lean-Agile development process.

SAFe defines two types of customers:

- *Internal customers* are part of the enterprise. They receive solutions from one or more development value streams and leverage them in one or more operational value streams. For example, a team of underwriting managers at a bank may be internal customers of a credit scoring solution created by the IT department (Figure 7).

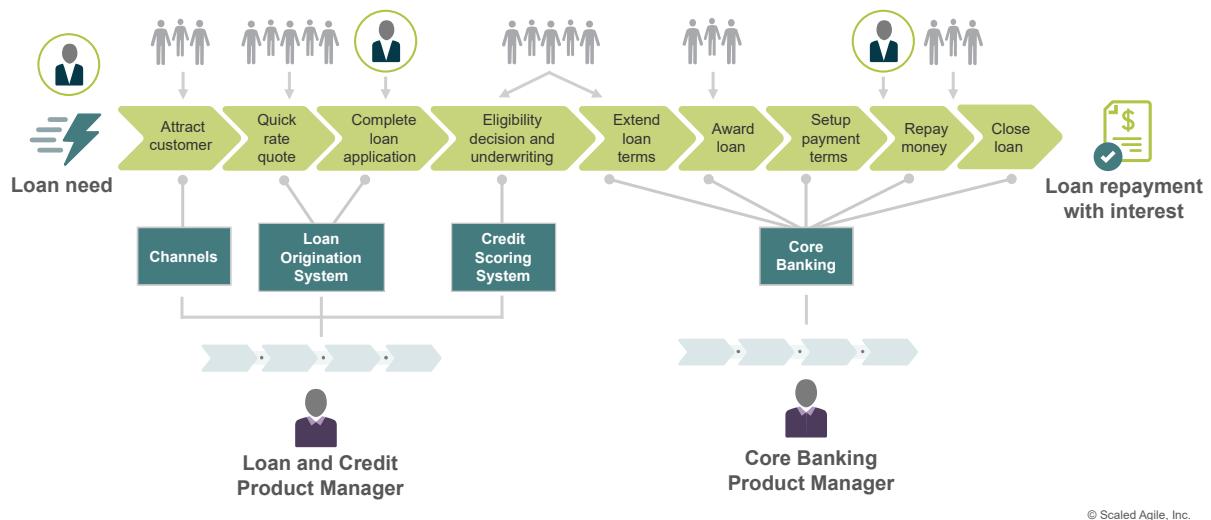


Figure 7. Product Management supports internal and external customers

- *External customers* are outside the enterprise. They purchase, license, or use solutions for their own benefit. Figure 3 depicts external customers that use solutions in the operational value stream to submit loan applications and repay loans. The relationship between the enterprise and external customers can take the form of business-to-business (B2B), business-to-consumer (B2C), or business-to-professional (B2P) interactions.

It is common for product managers throughout the organization to work together in developing a final solution that is composed of several integrated solutions (Figure 8).



Figure 8. Teams of Product Managers working together to create a solution

In this example, a mobile banking solution is delivered to external customers by a Mobile Banking Product Manager. The mobile banking app is built on a secure, compliant e-banking platform provided by a Digital Platform Product Manager. The core banking platform is built on scalable cloud infrastructure delivered by a Cloud Product Manager. Each oversees a solution that delivers a valuable product or service that is either consumed or further enhanced by the immediate customer. This scenario is especially common in the development of [Large Solutions](#).

## Learn More

[1] Leonard, Dorothy, and Jeffrey F. Rayport. *Spark Innovation Through Empathic Design*. Harvard Business Review, December 1997.

[2] Kotler, Philip, and Kevin Keller. *Marketing Management* 15th Edition.

[3] Hohmann, Luke. *Beyond Software Architecture: Creating and Sustaining Winning Solutions*. Addison-Wesley Professional, 2003.

[4] Moore, Geoffrey. *Escape Velocity: Free Your Company's Future from the Pull of the Past*. Harper Business, 2011.

[5] Levitt, Theodore. *Marketing Success Through Differentiation—of Anything*. Harvard Business Review, January 1980.

[6] <https://expertprogrammanagement.com/2017/10/five-product-levels/>

Last update: 15 October 2024

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



[Home](#) » Agile Release Train

## Agile Release Train

“ The more alignment you have, the more autonomy you can grant. The one enables the other.

—Stephen Bungay, author and strategy consultant [1]

**Definition:** An Agile Release Train (ART) is a long-lived team of Agile Teams that work together to incrementally develop, deliver, and often operate one or more solutions.

## Summary

Agile Release Trains (ARTs) consist of multiple Agile Teams, all aligned to a common vision and roadmap. These teams utilize SAFe Scrum and SAFe Kanban methods to optimize value delivery. ARTs are supported by the key roles of Product Management, System Architect, Release Train Engineer (RTE), and Business Owners. Each role contributes to the successful execution and strategic direction of the ART. Additionally, roles like Shared Services, Epic Owners, and System Teams provide specialized support. Cadence and synchronization through Planning Intervals (PIs) and regular ART events ensure alignment, commitment to objectives, and iterative solution delivery. ARTs actively involve the customer for optimal outcomes.

## What is an Agile Release Train?

An Agile Release Train (ART) is a team of Agile Teams aligned to a set of shared business and technology goals. ARTs are generally made up of 50 -125 people. They are cross-functional and have all the capabilities needed to define, build, validate, release, and, where applicable, operate one or more solutions.

When a product or solution is too large for a single Agile Team to deliver, an ART effectively creates alignment across multiple teams and a common way of working. This approach helps Agile Teams work together more smoothly and ensures that the value they are developing is ready for use at the same time. Agile Teams can deliver products to the market more efficiently and with better quality, surpassing what the enterprise could do with a less organized approach.

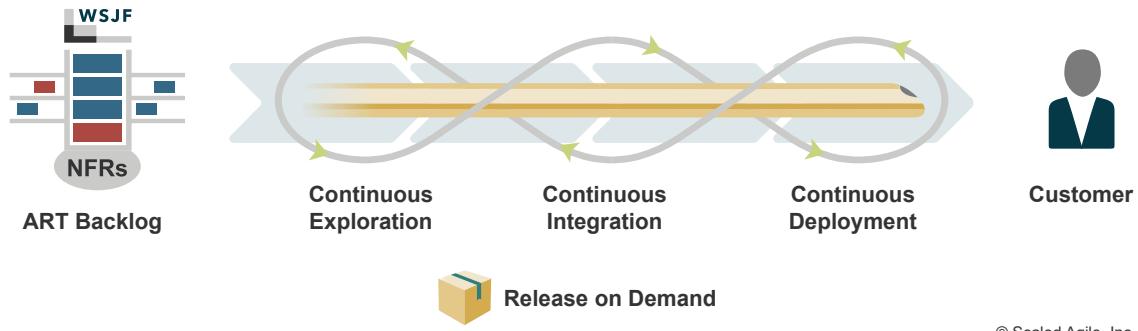


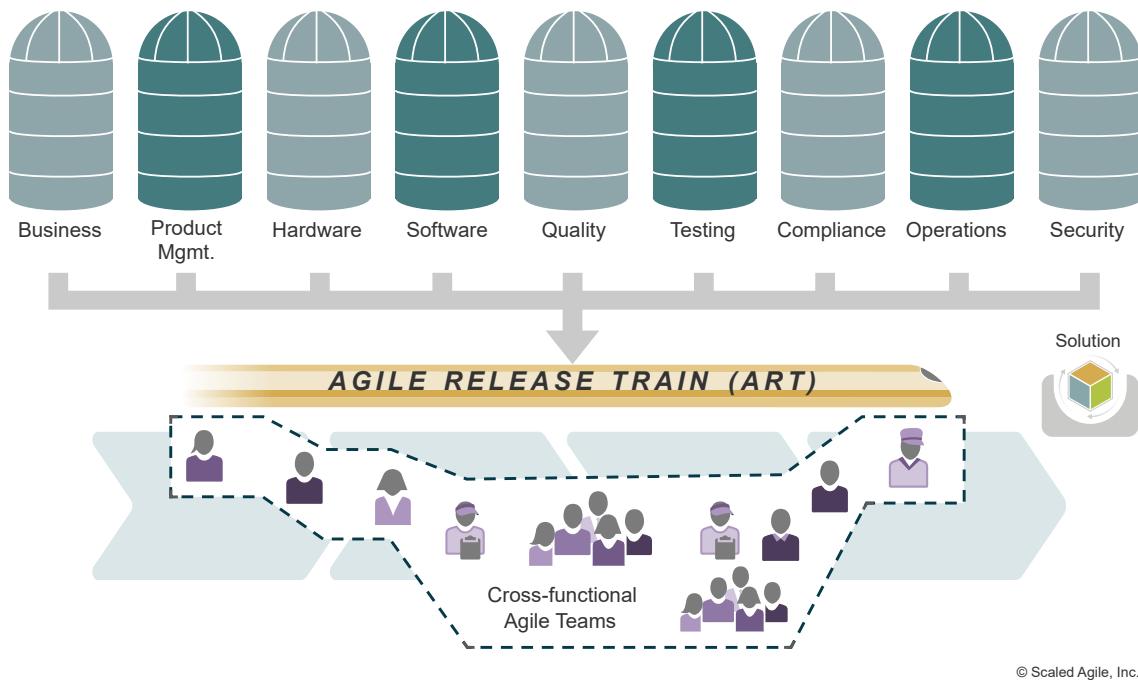
Figure 1. An Agile Release Train delivers customer solutions

Each ART works from an ART backlog, which contains the features that the Agile Teams on that ART develop to meet their customers' needs. The ART creates and maintains a continuous delivery pipeline needed to develop and release value.

## How to organize an Agile Release Train?

In a “functionally siloed” organization, developers work with other developers, testers work with testers, architects and systems engineers stick together, and operations staff work alone. This setup came to be for various reasons, but it slows things down because work must move through all these separate groups. Managers must coordinate the handoff of the work from one group to another, which slows things down and introduces delays.

In contrast, an ART has all the people needed to define, deliver, and operate the solution, eliminating these functional silos (Figure 2). This facilitates the fast flow of value from ideation through deployment and release. The ART is self-organizing and self-managing, so work gets done faster and smoother without unnecessary overheads.



**Figure 2. Agile Release Trains are fully cross-functional**

Read more about organizing Agile Release Trains:

Organize Around Value

## Who is on the ART?

Agile Teams power the ART. Agile Teams on an ART use SAFe Scrum, SAFe Kanban, or a mix of both, depending on what works best for each team. Collaborating with other Agile Teams, they build entire solutions working from a common ART.

Agile Teams may be technical teams focused on building software or hardware products, business teams such as marketing, legal, or finance, or a blend of each. Regardless, all the Agile Teams on the ART are aligned to a common vision and roadmap and participate in ART events. Together, the teams on the ART continually optimize their practices, accelerating value delivery.

ARTs also have specific ART leadership roles that support and coordinate the Agile Teams. The following ART leadership roles aid the successful execution of the ART:

- **Product Management** – One or more individuals primarily in charge of determining what gets built, based on the Vision, Roadmap, and Features in the ART Backlog. They

collaborate with customers, teams, and Product Owners to understand their needs and help validate the solutions being developed.

- **System Architect**—An individual or team that defines the system's overall architecture. Their job often involves deciding on the system's key performance and operational standards (Nonfunctional Requirements—NFRs), its primary components and subsystems, and how these parts connect and interact.
- **Release Train Engineer (RTE)** – An individual who focuses on ART execution, removes obstacles, accelerates value delivery, manages risks and dependencies, and fosters continuous improvement.
- **Business Owner(s)** – The primary stakeholder(s) in the Agile Release Train (ART). They are responsible for the ART's business outcomes, including return on investment (ROI), governance, and compliance.

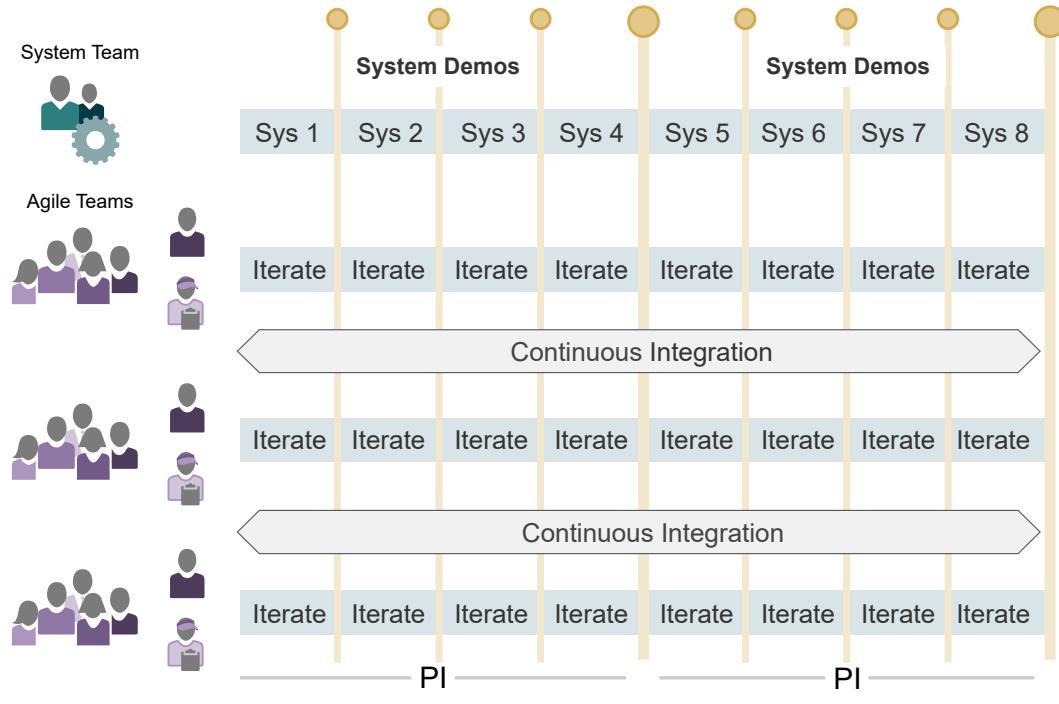
In addition to the above critical ART roles, the following play an important role in success:

- **Shared Services** are specialists who cannot be dedicated to a single ART. They often include data security, legal or compliance specialists, site reliability engineering (SRE), and many more.
- **Epic Owners** collaborate with stakeholders to create Epics, which represent significant solution development initiatives. This includes defining a Lean Business Case and MVP. They also collaborate with the ART and Agile Teams to guide the Epic through development. ARTs do not always work on Epics, so Epic Owners are not always involved with every ART.
- A **System Team** is a specialized Agile Team that may be created to assist in building and maintaining development, continuous integration, and test environments.
- **Customers** are the ultimate buyers or users of the solution. Agile Teams and ART leaders consistently apply customer-centric practices. When possible, customer involvement in ART events and activities can greatly improve outcomes.

## How does an Agile Release Train operate?

ARTs address one of the most common problems with traditional Agile development: teams working on the same solution operate independently and asynchronously. This makes it extremely difficult to integrate the entire system routinely. In other words, 'The teams are iterating, but the system isn't.' This increases the risk of issues and problems being discovered late.

Instead, the Agile Teams on an ART are aligned on a common cadence. They synchronize key activities, as shown in Figure 3, to ensure that the system is iterating together. Cadence and synchronization help keep everyone focused on improving and checking the whole system, not just parts of it.



© Scaled Agile, Inc.

**Figure 3. The entire ART is iterating**

Planning Intervals (PIs) provide the development rhythm for ARTs. PIs are typically 8-12 weeks long. A typical format for a PI includes four or five iterations of development followed by one iteration focused on innovation and planning. During each PI, the ART events ensure a complete Plan-Do-Check-Adjust cycle, which resembles a scaled-up version of the events that each Agile Team follows, as shown in Figure 4 below.

PI Planning creates the opportunity for everyone on the ART to align to the highest priority work for the business. The Agile Teams have the time needed to estimate what they can deliver, identify dependencies, and commit to a set of PI Objectives. During the PI the teams maintain alignment to the plan through ART synch events. They demonstrate the integrated solution in each iteration at a System Demo. At the end of the PI, the entire ART engages in an Inspect and Adapt event focused on identifying improvement actions to take into the next PI.

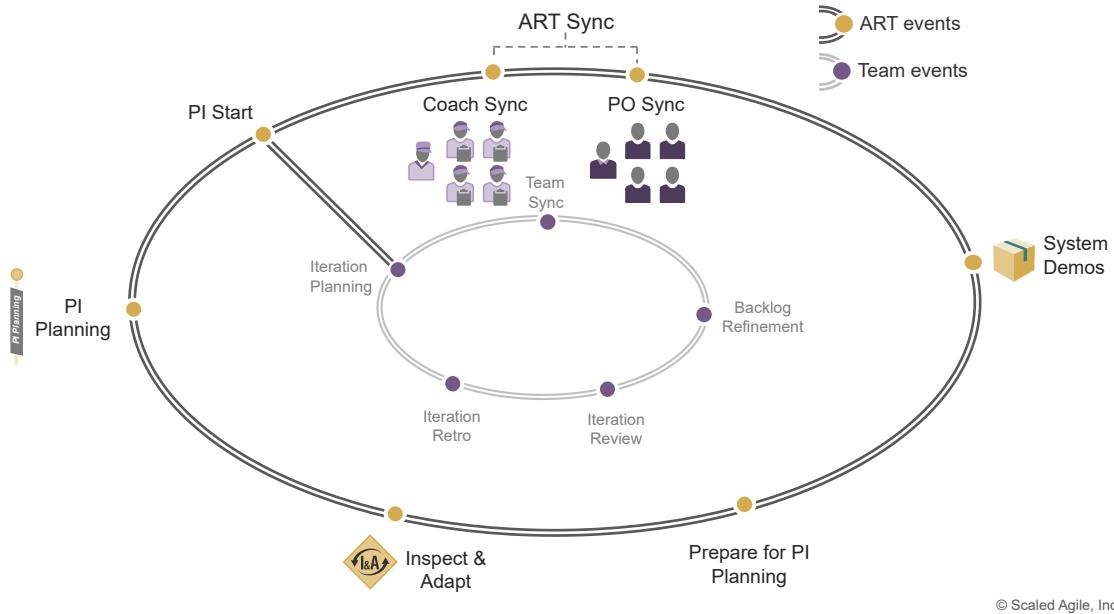


Figure 4. ART events enable the Agile Teams to align on cadence.

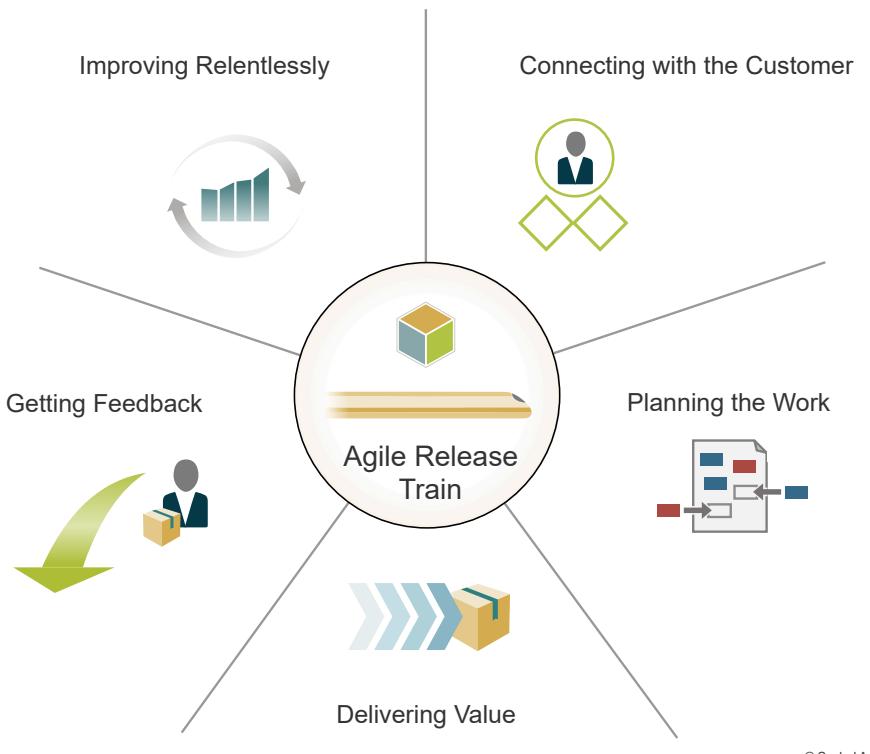
Read more about ART Events:

Planning Interval (PI)

## What are the responsibilities of an Agile Release Train?

The ultimate purpose of every ART is to deliver valuable solutions to the customer. To achieve that, an ART develops the solution iteratively, constantly engaging with the customer and adjusting the course of action toward an optimal solution.

Figure 5 shows the critical areas of responsibility of an ART that help achieve that objective. The responsibilities of an Agile Team and an ART are the same. Explained below are the specifics to consider when Agile Teams are formed into an ART:



© Scaled Agile, Inc.

Figure 5. ART responsibilities

## Connecting with the Customer



Customers are the ultimate economic buyers or value users of the solution an ART delivers. However, really connecting with the customer takes serious work and a good grasp of how to use lean and agile ways inside each ART's context.

- **Apply customer centricity** – An ART routinely focuses on customer needs and opportunities to benefit the customer. Customer Centricity is a necessary mindset for the entire ART. The ART works to increase and maintain customer empathy and continuously research better ways to solve customer problems. Sharing individual team research with the broader ART is critical to aligning the flow of value and innovation across the ART.
- **Use design thinking** – Design thinking includes a repeated process that helps teams create that are viable, desirable, feasible, and sustainable solutions. It encourages exploring many ideas and then narrowing them down to find innovative solutions that benefit users. ART leadership provides the opportunity for Agile Teams to apply and share design thinking methods. These methods help ensure that the team focuses on fulfilling the users' needs, continuously delivering innovative improvements.
- **Involve the customer in the development process** – There is no substitute for direct customer input. Including it in a routine development process helps an ART move at a much higher speed. The ART avoids the costly mistake of building capabilities the customer doesn't need or cannot use. Preparation for PI planning, the PI planning itself, and system demos provide venues for customer interaction.

## Planning the Work



Planning crucial activities for an ART enables alignment across teams and stakeholders in terms of what and how to build within the next timebox. Alignment is one of the Core Values of SAFe, and ARTs, as a building block of a SAFe organization, have built-in means for achieving and sustaining alignment.

- **Align ART priorities with portfolio strategy** – Every ART should align with the portfolio's overall strategy, guided by Strategic Themes toward shared goals. Alignment involves active engagement with portfolio stakeholders and including ART representatives in portfolio discussions. This coordination is aligned with the PI cadence. Epic Owners and Business Owners have primary roles in linking portfolio strategy to ART execution.
- **Prepare for PI Planning** – ART leadership, stakeholders, and the Agile Teams prepare for PI Planning. ART leadership maintains the vision and agrees on priorities for the next PI. Agile Teams refine features, assess capacity, and share emerging efforts, ensuring a smooth backlog flow across the ART.
- **Plan the PI** – PI planning creates alignment across the ART. Business Owners provide business and customer insights to the teams. Agile Teams then develop a plan to leverage technology and delivery capabilities for maximum business value. Agile teams create and agree with the ART leadership on the PI Objectives they can commit to delivering in the upcoming PI to achieve that business value.

## Delivering Value



ARTs develop solution features by applying a cadence that involves key activities to keep the train on track. At certain points, an ART will release the newly created value to the customer.

- **Frequently integrate and test** – A fast development rhythm requires frequent integration and testing. This helps uncover technology and implementation problems early and gives the teams enough time to respond to the findings. An ART operates in excessive uncertainty and variability without recurring integration and testing. Built-in Quality and Team and Technical Agility provide guidance on these practices.
- **Develop in short increments of value** – An ART implements the PI as a series of short increments, each representing a small batch of integrated, tested, and demonstrable value. The ART's iteration cadence provides a natural pace to create these increments. Each helps the ART learn about potential implementation challenges, get customer feedback, and agree on a decision point with possible course corrections for the rest of the PI.
- **Regularly synchronize and adjust** – While executing the PI, an ART has multiple checkpoints in the form of an ART Sync, which includes a Coaches Sync and PO Sync. These events increase visibility into the progress toward the current PI objectives and help the ART make timely adjustments.
- **Build a continuous delivery pipeline** – To effectively implement faster, more reliable development flow across the ART, it's important to create a Continuous Delivery Pipeline (CDP). The CDP allows teams to explore and integrate their work quickly with fewer errors

for faster feedback. The ART, particularly the System Architect, also ensures system designs are loosely coupled, enabling teams to deploy independently.

- **Establish a release process** – Each ART develops a release process to fit its cycle, ensuring alignment with strategy, compliance with standards, customer impact, and support for release tasks. Frequent releases aim to speed up market delivery and a consistent schedule helps quickly overcome technical hurdles, boosting progress and satisfaction across Agile Teams within the ART.

## Getting Feedback



Getting fast feedback is the primary component of an ART's high development velocity: speed comes from fast learning and adaptation rather than from 'working harder.' Technology feedback results from integration and testing as well as running technical spikes. The feedback on the product value comes from the customer and business stakeholders. ARTs routinely:

- **Measure business outcomes and usage** – Customer use of solutions may reveal issues and opportunities that otherwise might remain invisible to the ART. Creating the data capture and analytics capabilities, however, requires investment in the train's capacity, a proactive approach, and the use of Architectural Runway. Additionally, an ART must measure whether delivered solutions enable the desired business outcomes—the ultimate purpose of the ART's effort.
- **Perform routine testing** – Successful solution development is contingent upon an ART's ability to navigate the unknowns and make effective decisions. A/B testing enables effective decision-making and improves an ART's development speed. Instead of prematurely committing to certain functionality, the ART creates two or more options and validates them with users, thus gaining a real sense of which alternative is performing better.
- **Test User Experience** – User Experience (UX) is essential to fully realizing the solution potential. But to provide a great experience, there needs to be a clear UX design and testing strategy. This includes creating hypotheses and building and assessing the basic features needed by users, asking them questions, or analyzing data.

## Improving Relentlessly



An ART seeks to continuously improve the practices and interactions it uses to deliver customer value. There are many ways to measure and improve different aspects to achieve this goal:

- **Measure competency, flow, and outcomes** – Every ART should regularly assess against key applicable competencies. They also monitor their progress and use techniques to enhance workflow for continued improvement, often utilizing flow accelerators to identify and improve. Moreover, ARTs utilize the Key Performance Indicators (KPIs) and

Objectives and Key Results (OKRs) to measure outcomes aimed at delivering customer and business value.

- **Inspect & Adapt at regular intervals** – At every PI boundary, an ART has an opportunity to look back at the last PI, identify problems, and take corrective action during the Inspect & Adapt (I&A) event. This is dedicated time to identify significant, systemic improvement opportunities.
- **Make improvements on the fly** – Every ART routinely discovers small, local, and tactical improvement opportunities. In most cases, it is best to address these as they occur without waiting for a formal improvement event. This achieves quick wins and preserves the I&A for issues that require more attention.
- **Leverage Innovation & Planning Iteration** – The IP Iteration offers an opportunity to allocate uninterrupted time to innovation and learning. This helps the ART to further advance its solution, technical infrastructure, and various processes.

Read more about Agile Release Train responsibilities:

[Inspect & Adapt](#)

[PI Planning](#)

## Login to Access SAFe Studio Practical Tools



[Value Stream and ART Identification Workshop](#)

## References

- [1] Bungay, Stephen. *The Art of Action: How Leaders Close the Gaps Between Plans, Actions and Results* (10th Anniversary Edition). Nicholas Brealey, 2022.

### In this article

**What is an Agile Release Train?**

**How to organize an Agile Release Train?**

**Who is on the ART?**

**How does an Agile Release Train operate?**

**What are the responsibilities of an Agile Release Train?**

### Key Takeaways

- An Agile Release Train (ART) is a team of agile teams that collaborate for solution development.
- ARTs are cross-functional and contain all the necessary capabilities for the solution they deliver.
- Product Management, a System Architect, a Release Train Engineer, and Business Owners, support the ART in achieving its objectives.
- Organizing ARTs around value allows for rapid ideation, development, deployment, and release of customer solutions.

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



+ Framework

 ENGLISH (US) ▾

“ A Scrum Master is like an orchestra conductor, guiding a group of individuals to create something that no one of them could create alone.

—Mike Cohn, Paraphrased from *Succeeding with Agile* [1]

## SAFe Scrum Master/Team Coach



**Note:** For more on SAFe Scrum, please read the additional Framework articles in the Scrum series, including [SAFe Scrum](#), [Iterations](#), [Iteration Planning](#), [Iteration Goals](#), [Iteration Review](#), and [Iteration Retrospective](#)

### Scrum Master Stories: Yolanda

# Scrum Master/Team Coach Stories

Yolanda Berea

## Ready to start learning?

Use our finder to explore current offerings or learn more about a specific course

### Course

SAFe Scrum Master

[See available classes](#)

[Learn more about the course](#)

The SAFe Scrum Master/Team Coach (SM/TC) is a servant leader and coach for an Agile team who facilitates team events and processes, and supports teams and ARTs in delivering value.

They help educate the team in Scrum, Built-in-Quality, Kanban, and SAFe and ensure that the agreed Agile processes are followed. They also help remove impediments and foster an environment for high-performing team dynamics, continuous flow, and relentless improvement.

## Details

In SAFe, the Scrum Master/Team Coach (SM/TC) assists the team in meeting their delivery goals. They coach teams in self-organization and self-management and help them coordinate and participate in [Agile Release Trains \(ARTs\)](#) events, increasing the effectiveness of SAFe across the organization.

SAFe SM/TCs are integral members of an [Agile Team](#) and share responsibilities with the team for their overall performance. The SM/TC has the specialty skills that support adopting [SAFe Scrum](#) practices, ensuring no substantial gaps, and that the team knows how to plan, execute, review and retrospect. In addition, SM/TCs can actively coach [SAFe Team Kanban](#) teams and help each Agile Team achieve [Team Flow](#).

# Characteristics of a SAFe Scrum Master/Team Coach

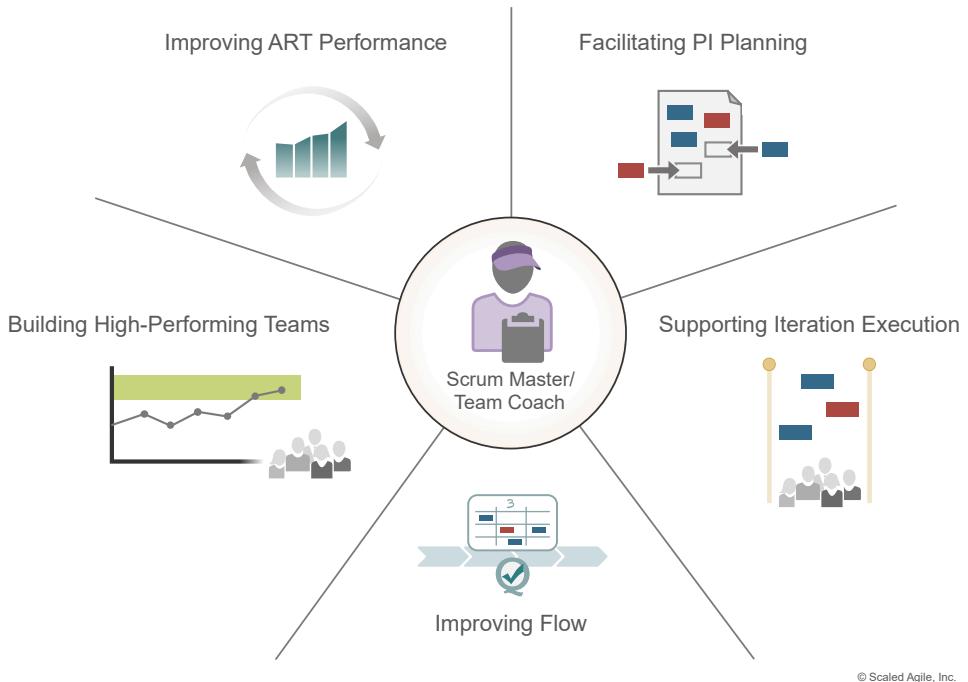
The SM/TC role is a team member who has the responsibility to help the team achieve its goals. They do this by teaching and coaching SAFe Scrum and SAFe Team Kanban, and by supporting SAFe principles and practices. They also help identify and eliminate bottlenecks to flow.

SM/TCs come from various backgrounds and roles, and they are in high demand. Although they are not typically people managers, SM/TCs are influential members of an Agile Team, and they should have the following attributes:

- **Empathetic** – Support the team by displaying an authentic understanding and concern for a team member's beliefs or feelings. In turn, the team is more likely to build relationships with others, resulting in higher levels of collaboration and performance. Empathy is a crucial ingredient of trust, which is essential for people to accept and welcome coaching.
- **Conflict navigator** – Supports team members in resolving interpersonal conflicts, problem-solving, and decision-making. Agile coach and author Lyssa Atkins opines, "Navigating conflict is our new mindset, in which we help teams move from conflict to constructive disagreement as a catapult to high performance." [2]
- **Servant leader** – Persuades rather than uses authority. As servant leaders, SM/TCs focus on the needs of team members and those they serve, intending to achieve results aligned with the organization's values, principles, and business objectives. [3] They have choices in how they collaborate with the team depending on the situation and their accountability for team performance. SM/TCs should have options for achieving their responsibilities. For example, when it comes to events their accountability should be 'ensuring that all team events take place and are positive, productive, and kept within the timebox.' SM/TCs can facilitate the events or let the team self-manage and facilitate their own events. Rotating the responsibilities for facilitating events and meetings is essential to the team's growth and its ability to self-manage.
- **Mentor** – Supports the personal development of team members, helping them gain a continuous learning mindset. They guide the team to find solutions to their problems independently instead of being given the answers.
- **Transparent** – Transparency is a [Core Value](#) of SAFe and one of the pillars of empiricism. The SM/TC is open to feedback and appreciates transparency from others. They help the team provide transparency by ensuring artifacts are inspected, identifying significant differences between expected and actual results, and detecting anti-patterns.
- **Coach** – The SM/TC understands and educates the team on methods beyond Scrum, such as SAFe, Kanban, Flow, Built-in Quality, and more. They often have advanced training and experience in one or more technical and business domains.

## Responsibilities

The SM/TC fulfills many critical responsibilities in performing the role, as illustrated in Figure 1. Each of these responsibilities is described in the sections that follow.



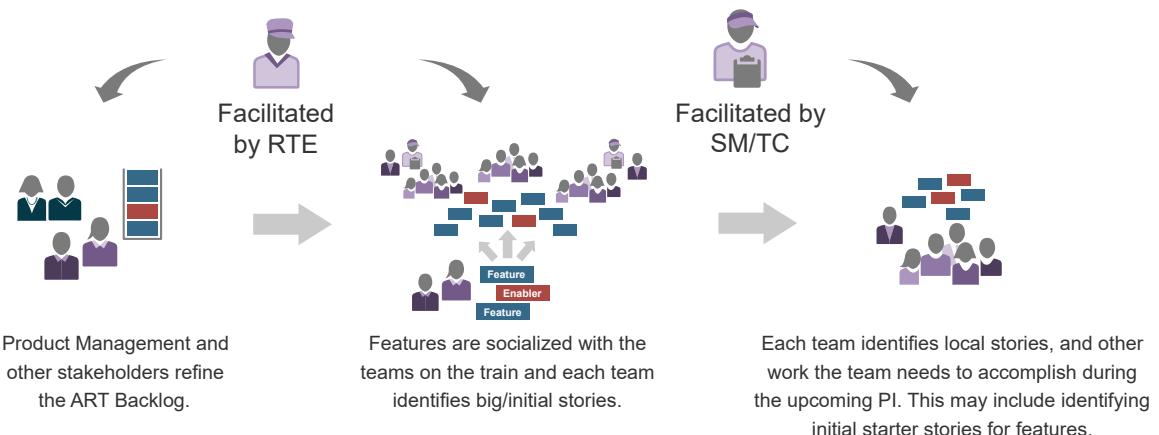
© Scaled Agile, Inc.

Figure 1. SAFe SM/TC responsibility areas

## Facilitating PI Planning

SM/TCs play an essential role in [PI Planning](#). They collaborate with other SM/TCs and the [Release Train Engineer \(RTE\)](#), working actively with the team during PI planning. An effective SM/TC is critical to a successful event, and they typically do the following activities to help facilitate PI planning:

- **Prepare for PI Planning** – Before the event, the SM/TC ensures the team is briefed on upcoming features by [Product Managers](#), [Business Owners](#), and other stakeholders, as illustrated in Figure 2. They help Agile Teams and the [Product Owner](#) identify local stories, maintenance, defects, tech debt, and other work the team needs to accomplish during the upcoming PI.



© Scaled Agile, Inc.

Figure 2. Preparing for PI planning

- **Draft PI plans** – The SM/TC facilitates the team in creating a draft PI plan for the PI's iterations, writing draft [PI Objectives](#), and identifying ART risks and issues. The SM/TC also helps the team set up their digital or physical planning areas, providing visual radiators that create transparency and collaboration. They help the team determine their capacity and keep within this constraint.
- **Coordinate with other teams** – SM/TCs help ensure cooperation and communication during the event. During PI planning, they usually secure subject matter experts (SMEs) and ART stakeholders and foster communication with other teams to determine how they will collaborate on feature development and resolve dependencies.
- **Create team PI objectives** – SM/TCs help teams create team PI objectives, the things they intend to accomplish in the upcoming PI. They ensure the objectives are written before the draft plan review and that a proper mix of committed and uncommitted goals is present.
- **Review final plans and business value** – Before the final review, SM/TCs help ensure PI objectives are SMART (Specific, Measurable, Achievable, Realistic, and Time-bound) and are written in a way everyone can understand. The SM/TC often facilitates Business Owner and team collaboration during business value assignments.

## Supporting Iteration Execution

SM/TCs support Agile Teams during the iteration, increasing the likelihood of achieving its iteration goals and PI objectives. For example they:

- **Facilitate team events** – Agile Teams use cadence-based events to coordinate and sync their efforts. While Scrum and Team Kanban operate somewhat differently, all teams need to plan, sync, review, inspect their work, and hold retrospectives. Figure 3 illustrates the events (or activities) that Agile Teams typically do during an iteration.

Iteration Events	Scrum Master/Team Coach's Role (SM/TC)
<b>Backlog Refinement</b>	The SM/TC assists the Product Owner and team to review the user stories prioritized at the top of the team backlog to ensure there are always stories ready for implementation.
<b>Team Planning</b>	All Agile Teams plan their work. Scrum Teams plan in a specific event called Iteration Planning. Kanban teams plan their work at iteration boundaries, weekly, or ad hoc as they so determine. The SM/TC ensures that planning events are held and effective in accordance with the planning method chosen by the team.
<b>Team Sync</b>	The SM/TC typically facilitates the team sync, enabling the team to plan and coordinate the day's work and inspect progress toward the iteration and PI goals.
<b>Review</b>	The SM/TC ensures that teams review and demo their progress. Scrum teams have a specific <i>iteration review</i> event. Kanban teams periodically review and demo their progress as needed. The SM/TC helps ensure the team receives feedback to improve the solution under development and collaborates with the Product Owner to determine future work.
<b>Retrospective</b>	The SM/TC facilitates team retrospectives and creation of an improvement backlog. Scrum teams have a specific <i>iteration retrospective</i> event. Kanban teams conduct a team retrospective as needed. The SM/TC may need to address ART-related problems at the ART Sync or with the team at the I&A event.

© Scaled Agile, Inc.

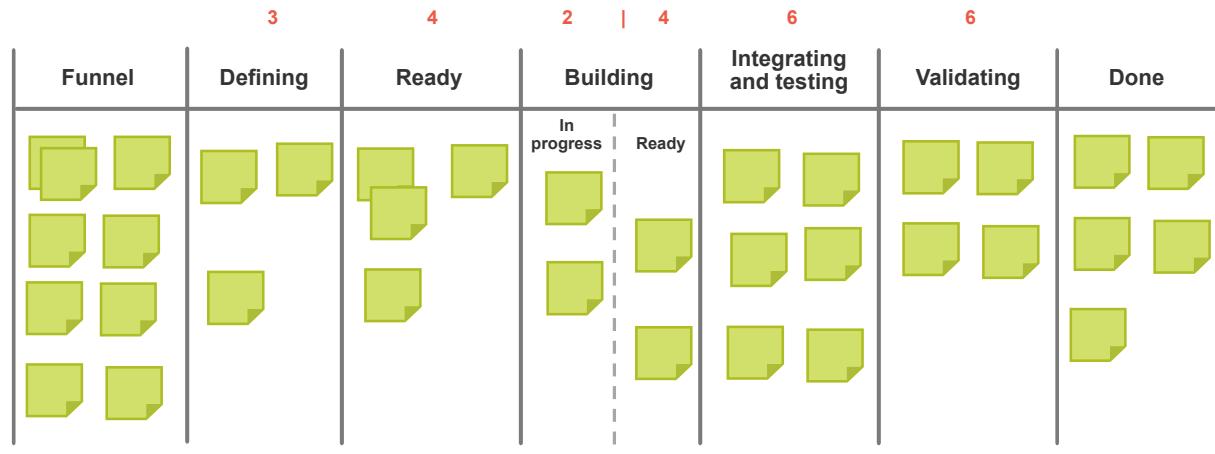
Figure 3. SM/TCs' role in ensuring successful team events

- **Work within the ART's cadence** – SM/TCs help teams apply Scrum or Kanban within the development cadence and synchronization requirements of the [Agile Release Train \(ART\)](#). This cadence and synchronization facilitate alignment, dependency management, Release on Demand, and fast integrated learning cycles ([SAFe Principle #4](#)).
- **Collaborate with the PO** – Since the [Product Owner \(PO\)](#) is accountable for maximizing the solution's value resulting from the team's work, an essential aspect of the SM/TC's role is supporting the PO. They do this by:
  - Helping the team understand and apply the tools and techniques for [Customer-Centricity](#) and [Design Thinking](#) to build the right thing at the right time
  - Ensuring the team understands the need for clear and concise team backlog items and aligns to the ART's capacity allocation for each work item type.
  - Helping the team apply empirical planning and development where progress is evaluated based on observation and experimentation of working solutions in small increments
  - Facilitating stakeholder collaboration as requested or needed

## Improving Flow

SM/TC can significantly improve the team's flow of work, eliminating bottlenecks, delays, and waste. This coaching often includes the following activities:

- **Establish the team Kanban board** – SAFe teams use a Kanban board to visualize their work and enhance flow. Implementing an effective Kanban system adapted to meet the needs of a specific Agile Team is based on the type of work performed (marketing, software development, hardware), the team members' skills, and their role in the ART. Creating the Kanban system is best done by involving the entire Agile Team with the guidance and facilitation of an experienced coach like the SM/TC. The SAFe extended guidance article, [Applying Kanban in SAFe](#), describes how to establish a Kanban system and how the Kanban systems are connected in SAFe. Figure 4 illustrates an example of a Team Kanban board.



© Scaled Agile, Inc.

Figure 4. Example Team Kanban board

- **Measure and optimize flow** – SM/TCs help the team establish metrics to assess and improve its overall performance. Specific measures for *flow*, *competency*, and *Outcomes* are described in [Measure and Grow](#). Flow Metrics help the SM/TC and the team evolve its process iteratively and continuously adapt to fit the team's needs. After defining the initial process and WIP limits and executing for a while, bottlenecks should become visible. If not, the team refines the process or further reduces some WIP limits until it becomes evident that a workflow state is overloaded or starving. Other coaching opportunities for optimizing flow might include merging or splitting steps, adding buffers, swim lanes, and classes of service, or redefining workflow states.
- **Build quality in** – Agile Teams operate in a fast, flow-based system to develop and release high-quality business capabilities quickly. The SM/TC helps achieve this by coaching [Built-in quality](#) practices, which enable fast, reliable execution and helps ensure that needed and frequent changes are made efficiently and effectively.

## Building High-Performing Teams

Creating healthy Agile Teams is essential to creating high-value increments of working solutions. Fortunately, many of the ingredients for high-performing teams are built into SAFe by design. For example, Agile Teams in SAFe are small, cross-functional, and self-organizing. They are empowered to define and execute the work needed to accomplish the team's objectives and those of the ART. Everyone agrees that all increments should meet a shared, scalable definition of

done. SM/TCs play a critical role in building high-performing teams and accomplish this through the following types of activities:

- **Foster and support Agile Team attributes** – While every team is different, there are common characteristics that high-performing teams share. SM/TCs are responsible for supporting and fostering the following Agile Team attributes:
  - Self-management and taking ownership and accountability
  - Aligned and collaborative
  - Success focused on clear goals and purpose
  - Influential decision-makers who understand their work's impact on others
  - Operate with open and transparent communication and trust
  - Value diversity and healthy conflict
  - Provide effective, timely feedback
  - Highly engaged and have fun with work, and each other
- **Encourage high-performing team dynamics** – SM/TCs foster an environment for high-performing team dynamics, continuous flow, and relentless improvement. The SM/TC mentors the team and creates an atmosphere of mutual respect, helping resolve interpersonal conflicts, and identifying growth opportunities. They assist the team in *focusing* on creating increments of high value for each iteration.
- **Become a more effective Scrum Master/Team Coach** – Every servant leader knows that their growth comes from facilitating the development of others who deliver the results. SM/TCs serve the team and the larger organization. The SM/TC supports the overall adoption of SAFe across the enterprise by coaching stakeholders and non-agile teams on effective interactions with Agile Teams, participating in the SM/TC Community of Practice, and supporting the organization's [SAFe Practice Consultants \(SPC\)](#).
- **Serve as Lean-Agile Leaders** – SM/TCs also advance the adoption of SAFe. They lead by example and incorporate the [Lean-Agile Mindset](#) and [SAFe Lean-Agile Principles](#). They integrate these concepts into their responsibilities and serve as a role model for others to follow.
- **Foster collaboration on the team** – The SM/TC role fosters more effective and cohesive teams, enabling better business outcomes, solutions, and products. They offer observations, feedback, guidance, and advice based on what they know and have seen work.
- **Coach with powerful questions** – However, SM/TCs do not have all the answers. Instead, they can ask *powerful questions* to uncover what's essential, then guide others to tap into their knowledge and expertise. Some examples of powerful questions include:
  - What brings us to this inquiry?
  - What other possibilities or options exist?

- What is it we're not seeing?
- What do we need to do to reach a deeper level of understanding?
- If success was guaranteed, what actions would you take?

By asking powerful questions, SM/TCs help teams improve their performance, solve problems more independently, make better decisions, learn new skills, and better reach their goals.

- **Resolve team conflicts** – Teamwork is the ultimate competitive advantage. However, many teams are dysfunctional, according to Patrick Lencioni, consultant and author of *Five Dysfunctions of a Team*. In his book, Lencioni suggests that an absence of trust leads to the other four dysfunctions. [4] SM/TCs helps address these five dysfunctions with the SAFe practices, as illustrated in Figure 5.

Inattention to results	Results are empirically reviewed at the end of every iteration and release. Iteration Retrospectives drive continuous improvement.
Avoidance of accountability	Stakeholders, peer pressure, and review of results drive accountability.
Lack of commitment	Teams make shared commitments to each other and to the external stakeholders.
Fear of conflict	Scrum creates a safe environment for conflict; The Scrum Master/Team Coach encourages discussion of disagreements. Shared commitment avoids conflict that occurs when objectives are not aligned.
Absence of trust	The environment is safe. The team shares commitment and goals, displays hyper-transparency, and engages in retrospectives.

© Scaled Agile, Inc.

Figure 5. The SAFe SM/TC helps address the five dysfunctions of a team

- **Develop team skillsets** – SM/TCs work with team members and their functional managers to help them acquire T-shaped skills. A *T-shaped* individual has broad, general expertise in many areas and is an expert in one of these disciplines. SM/TCs encourage team members to pair with others to expand their skills, take on tasks in another discipline and business domain, and participate in training courses and reading books to become continuous learners.

## Improving ART Performance

SM/TCs help Agile Teams improve the overall ART performance through the following activities:

- **Facilitate cross-team collaboration** – Cross-team collaboration is a hallmark of high-performing teams. Agile Teams cooperate across departments to bring whole product solutions to market. SAFe SM/TCs nurture an environment where cross-team collaboration

thrives and is supported by practices that offer opportunities for teams to work together, for example:

- Alignment to ART PI objectives, Vision, and Strategic Themes during PI planning and addressing dependencies using the ART board
- Representing the team in the Coach Sync, PO Sync, and ART Syncs
- Attending other team's events and demos with relevant team members
- Participating in the ART's [System Demos](#) and [Inspect & Adapt](#) events

One of the significant benefits of working on and across teams is that colleagues learn from one another. On an Agile Team, learning new skills makes everyone more valuable to the organization and better equipped to support each other's work. It also guards against specialty skills becoming a bottleneck, which increases delays and reduces quality.

- **Build trust with stakeholders** – The SM/TC helps the team build trust. SAFe relies on a rolling wave of short-term commitments from Agile Teams and ARTs to assist with business planning and outcomes, resulting in improved alignment and trust between development and business stakeholders. While solution development is uncertain by its very nature, the business depends on teams for some amount of reliable, predictable forecasting. Too little predictability and the company can't plan. Too much, and the organization has committed to longer-term plans, which are unreliable and limit agility. Business and technology stakeholders need something in between, which is the primary purpose of PI objectives.
- **Coach the IP Iteration** – SM/TCs help ensure the team does not schedule any work for the IP Iteration during PI planning. Instead, they coach teams to use this iteration as an estimating buffer for meeting PI objectives and providing dedicated time for innovation, continuing education, PI planning, and the Inspect and Adapt (I&A) events.
- **Help the team inspect & adapt** – Ensures the team is prepared for the Inspect & Adapt event, including the PI System Demo, quantitative and qualitative measurement, and the retrospective and problem-solving workshop. They help guide the team in the I&A activities and stay within the allotted timeboxes.
- **Facilitate the problem-solving workshop** – SM/TCs coach teams in root cause analysis, the 'five whys,' [5] and Pareto analysis [6]. They ensure that the relevant work needed to deliver the identified improvements is planned and added to the Team Backlog.

## Full or Part-Time Role?

The SM/TC can be a part-time or full-time role, depending on the size of the team, the context, and other responsibilities. However, it can be challenging for an [Enterprise](#) to justify the need for a full-time SM/TC for each Agile Team. SAFe takes a pragmatic approach, where sometimes a team member assumes the role along with other duties, or an accomplished SM/TC can support more than one team. However, during initial SAFe adoption, the job can be more intensive. It's often beneficial to hire external SM/TC consultants to mentor the teams and help them become

experienced in their roles and SAFe. These consultants will work with multiple teams and new SM/TCs. And, of course, adequate training and experience are required to be effective.

---

## Learn More

- [1] Cohn, Mike. *Succeeding with Agile: Software Development Using Scrum*. Addison-Wesley Professional, 2009.
  - [2] Zanetti, Alessandro. *Scrum Master as Conflict Navigator*. Medium, June 21, 2021. Retrieved October 12, 2023, from <https://medium.com/serious-scrum/scrum-master-as-conflict-navigator-de5c6a162fe>
  - [3] Overeem, Barry. *The Scrum Master as a Servant-Leader*. Scrum.org, July 20, 2015. Retrieved October 12, 2023, from [https://www.scrum.org/resources/blog/scrum-master-servant-leader#\\_ftnref7](https://www.scrum.org/resources/blog/scrum-master-servant-leader#_ftnref7)
  - [4] Lencioni, Patrick. *The Five Dysfunctions of a Team: A Leadership Fable*. Jossey-Bass, 2002.
  - [5] *5 Whys: The Ultimate Root Cause Analysis Tool*. Businessmap. Retrieved October 12, 2023, from <https://businessmap.io/lean-management/improvement/5-whys-analysis-tool>
  - [6] Kenton, Will. *What Is Pareto Analysis? How to Create a Pareto Chart and Example*. Investopedia, December 30, 2022. Retrieved October 12, 2023, from <https://www.investopedia.com/terms/p/pareto-analysis.asp>
- Gurteen, David. *Designing powerful questions*. Conversational Leadership. Retrieved October 12, 2023, from <https://conversational-leadership.net/powerful-questions/>

Last Update: 12 October 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## **Training**

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## **Content & Trademarks**

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## **Scaled Agile, Inc**

### **Contact Us**

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### **Business Hours**

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



+ Framework

 ENGLISH (US) ▾

“ Business people and developers must work together daily throughout the project.

—Agile Manifesto [1]

## Product Owner

### Ready to start learning?

Use our finder to explore current offerings or learn more about a specific course

Course

SAFe Product Owner/Product Manager ▾

[See available classes](#)

[Learn more about the course](#)

The Product Owner (PO) is the Agile team member primarily responsible for maximizing the value delivered by the team by ensuring that the team backlog is aligned with customer and stakeholder needs.

As a member of the extended [Product Management](#) function, the PO is the team's primary customer advocate and primary link to business and technology strategy. This enables the team to balance the needs of multiple stakeholders while continuously evolving the Solution.

## Details

For most enterprises moving to Agile, this is a new—and typically full-time—role for each [Agile Team](#). Each PO represents the needs of customers and the business within a particular Solution domain, typically co-represented by a Product Manager. Together, they ensure that product strategy and implementation remain connected throughout the value stream.

Serving as the 'voice of the customer' for the team entails a broad range of responsibilities. The PO must build and manage key relationships, synthesize information from multiple sources, maintain business alignment in the [Team Backlog](#), and communicate effectively with various audiences—all with a bias toward delivering and learning quickly.

## Responsibilities of the Product Owner

The PO's responsibilities can generally be categorized into five primary areas, as shown in Figure 1.



© Scaled Agile, Inc.

Figure 1. Product Owner areas of responsibility

Each of these areas of responsibility is described in the sections below.

## Connecting with the Customer



Ensuring that ARTs are continually building the right things and building them right is a never-ending process. Product strategy, design, and implementation must evolve with ever-changing customer desires and business needs. The PO, in close partnership with Product Management, applies a customer-centric mindset along with design thinking tools to guide the ART toward delivering solutions that are desirable, viable, feasible, and sustainable. The PO applies [Customer Centricity](#) and [Design Thinking](#) in the following ways:

- **Know the customer** – Value is determined by the customer; therefore, the PO is keenly aware of the needs of the people to whom their products are delivered. Customers may be internal or external to the enterprise and may have direct or indirect relationships with the PO. Whether they consume products, services, systems, APIs, platforms, or other solutions, customers' wants, needs, and preferences are continually explored by the PO.
- **Know the stakeholders** – Product design and implementation must also reflect the needs of non-customer stakeholders. [Business Owners](#), [Lean Portfolio Management](#), Product Management, [System Architects](#), and fellow POs, for example, rely on the cadence and quality of the team's output. The PO identifies key stakeholders and balances their needs with those of the customer.
- **Identify the problem to be solved** – Good products solve specific problems. What's more, they solve specific problems that are *worth* solving. Identifying problems that customers

want to be solved is the first element of design thinking. In this context, the PO discovers a range of customer needs through divergent thinking tools, then identifies the 'jobs to be done' that are most worth pursuing.

- **Develop whole-product solutions** – Solutions that address a range of customer needs are more valuable than those that target a single need. POs aim to deliver whole-product solutions by understanding the desired customer experience, guiding the development of candidate designs through the [Lean UX](#) process, and delivering tested concepts that maximize customer satisfaction and loyalty.

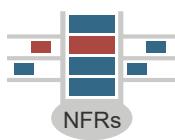
## Contributing to the Vision and Roadmap



While product managers contemplate the solutions and experiences an ART *should* deliver, POs understand what solutions and experiences the ART *can* deliver. This practical insight is a valuable contribution to the vision and roadmaps that guide solution implementation. The PO applies this pragmatic insight in the following ways:

- **Understand market forces** – Market rhythms, market events, sudden opportunities, competitive threats, and changing regulations significantly influence product strategy. POs regularly engage with Product Management to analyze market research and understand the business drivers that trigger Feature requests.
- **Represent the end user** – Through frequent interviews, gemba, and reporting, POs are strongly connected to the needs and experiences of their products' end users. Objective insights about how end-users interact with solutions and the features they desire most ensure that the vision and roadmap contain real value.
- **Assist with ART Backlog prioritization** – In collaboration with Product Management, System Architects, Release Train Engineers (RTEs), and other stakeholders, POs guide the sequencing of features over time toward the best economic outcomes. Through their understanding of which problems need to be solved, which solutions would best solve them, and the feasibility of delivering those solutions, POs help ensure that the vision and roadmap are reflected in the [ART Backlog](#).
- **Educate the ART during PI Planning** – The [Vision](#) and [Roadmap](#) are living artifacts created and adjusted in alignment with business and technical strategy, but a portion of them is always in scope for implementation. The PO assists with communicating the vision and roadmap during PI Planning to ensure teams are aligned and ready to execute against them.

## Managing and Prioritizing the Team Backlog



With input from Product Management, System Architecture, and other stakeholders, the PO is primarily responsible for maintaining the content and the conceptual and technical integrity of the team backlog. Consisting of user stories, enablers, and defects, the backlog must always contain work ready to be pulled

for implementation by the team and aligned with the most current needs of customers and stakeholders. The PO manages the ongoing integrity of the team backlog through the following activities:

- **Guide Story creation** – While any team member can write stories at any time, it is the PO's responsibility to ensure that they are well-formed and aligned with product strategy. The PO clarifies story details, applies user-story voice, ensures 'INVEST' [2] characteristics are present, assists with story splitting, defines enablers, and incorporates behavior-driven development (BDD) to ensure stories support continuous value flow. The PO also allows space for 'local' stories and spikes that advance product design but are not derived explicitly from ART-level features.
- **Prioritize backlog items** – Achieving continuous value flow requires that the highest-value backlog items are delivered in the shortest sustainable lead time and in the right sequence. The PO enables this by regularly ordering backlog items according to their cost of delay and communicating that sequence to the team during backlog refinement and [Iteration Planning](#).
- **Accept Stories** – The PO works with the team to agree on accepted story completion. This includes validating that the story meets acceptance criteria, that it has the appropriate, persistent acceptance tests, and that it otherwise complies with its Definition of Done (DoD). In so doing, the PO assures that quality is built in.
- **Support Architectural Runway** – POs do not typically drive technological decisions, but they make space in the backlog to support the implementation of [Architectural Runway](#). They collaborate with System Architects to craft enablers and work with stakeholders to establish appropriate capacity allocations.

## Supporting the Team in Delivering Value



Value is created when Agile teams pull from the backlog, implement stories, integrate and test changes, and deliver a solution increment. These value-creation activities occur primarily during iteration execution. As an integral member of the team and their primary customer proxy, the PO provides daily insights that guide development toward the highest-value outputs and the team toward meeting iteration goals. This enables the team and, in turn, the ART, to deliver continuous value.

- **Balance stakeholder perspectives** – POs constantly receive input, feedback, and insights from customers, stakeholders, teams, and tools that can impact solution development. This information can validate, invalidate, or challenge implementation decisions unexpectedly. Moreover, these sources often conflict with one another. POs balance these perspectives by understanding the needs that drive them, remaining customer-centric, respecting capacity allocations, evaluating the cost of delay, and collaborating with stakeholders and teams to make implementation decisions that produce the most favorable outcomes.
- **Elaborate Stories** – Stories are typically created before iteration execution but require ongoing elaboration. POs facilitate frequent conversations with their teams to resolve

questions, manage dependencies, and communicate priorities that emerge as stories are implemented. This information also helps the team slice stories effectively to achieve increased velocity and shortened learning cycles.

- **Foster Built-In Quality** – As the primary proxy for customers and stakeholders on the team, the PO plays a pivotal role in evaluating the value delivered from the backlog. The PO regularly evaluates progress toward story acceptance criteria, including compliance with [Built-In Quality](#) criteria, such as the scalable definition of done, and nonfunctional requirements (NFRs). The PO works closely with the team to detect quality issues as they are introduced and correct them in or near real-time.
- **Participate in team and ART events** – As a member of the Agile team, the PO, naturally, attends and actively participates in team events during PI execution. During iteration planning, backlog refinement, iteration reviews, team retrospectives, and team syncs, the PO provides crucial feedback on the team's work from an outside-in, customer-centric point of view. By participating in [PO Sync](#) and [System Demos](#), the PO helps the team satisfy dependencies, demonstrate incremental value, and maintain cadence with the ART.

## Getting and Applying Feedback



The PO is responsible for maximizing the value delivered by an Agile team. This, of course, implies that value is known. That knowledge comes from frequent feedback from customers and stakeholders—not just upon delivery but throughout the entire delivery life cycle. The PO is critical in enabling the continuous feedback loops that fuel the value stream. The PO seeks quantitative and qualitative feedback to develop a comprehensive understanding of where solutions are and are not providing real value. The following activities enable the PO to gather and apply feedback from several key sources:

- **Test benefit hypotheses** – Value can only be realized when a working solution is in the hands of the customer. Even then, it is not guaranteed. During development, therefore, value can only be speculated. To mitigate the risk of delivering solutions with little value, the PO collaborates with Product Management to define benefit hypotheses based on extensive customer and market knowledge. These hypotheses drive implementation and are validated (or invalidated) by feedback the PO gathers from customers and stakeholders throughout the product life cycle.
- **Obtain feedback from customers and stakeholders** – Customers derive value by using delivered solutions. Their feedback indicates how well solutions meet their needs, which drives solution adoption and loyalty. Stakeholders derive value from revenue, cost savings, or decreased risk stemming from customers' use of delivered solutions. The PO gathers this feedback directly via empathy interviews, Gemba walks, iteration reviews, and system demos and indirectly via application telemetry, usage analytics, financial reporting, and secondary market data.
- **Share feedback with the ART** – Because solution delivery requires coordination and synchronization across the value stream, the feedback collected by the PO is valuable to the whole ART. The PO shares this information with Product Management and System

Architects as part of [Continuous Exploration](#), with other POs during PO Sync, with their teams during backlog refinement, iteration planning, and iteration reviews, and with the ART during PI planning, system demos and, if applicable, [Inspect and Adapt](#) events.

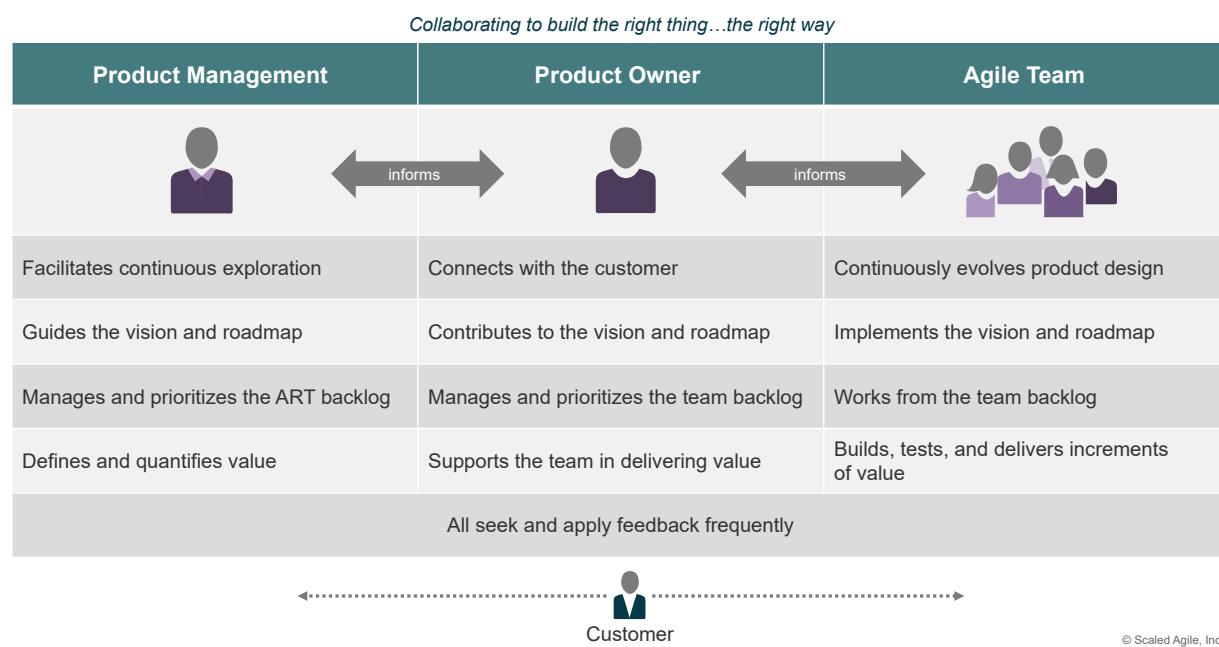
- **Evolve solution design** – Frequent, customer-centric feedback cycles fuel the Plan-Do-Check-Adjust cycle that enables continuous value delivery and the ongoing, relentless improvement of the value stream itself. By gathering and sharing these critical insights, the PO enables the continuous refinement of the product vision, roadmap, strategy, and design toward optimum business value.

## Key Partnerships

The PO is ultimately responsible for maximizing the value delivered by the Agile team, which requires the PO to ensure that the *right solutions are built* and that they are *built the right way*. However, the PO cannot accomplish this alone.

Building the right solutions requires deep knowledge of business strategy, customer segmentation, market dynamics, and value stream economics. The PO establishes a close relationship with Product Management to derive these macro-level insights and apply them to specific product domains. Building solutions the right way requires [Team and Technical Agility](#), [DevOps](#) practices, and a [Continuous Delivery Pipeline](#). These technical capabilities determine the speed and quality with which value can be delivered, and the PO relies on the Agile team to provide them.

The PO provides a crucial link in the bi-directional information flow between Product Management and the Agile team. As shown in Figure 2, the PO keeps the Agile team informed of the strategy that drives product design and keeps Product Management informed of the innovations that influence the evolution of product strategy. Customer feedback aligns thinking from strategy through execution and is accessible to all roles.



## Learn More

[1] *Manifesto for Agile Software Development.* <http://AgileManifesto.org/>

[2] Wake, Bill. *INVEST in Good Stories, and SMART Tasks.* XP123, August 17, 2003. Retrieved October 12, 2023, from <https://xp123.com/articles/invest-in-good-stories-and-smart-tasks/>

Last update: 12 October 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

# Scaled Agile, Inc

## Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

## Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



[Home](#) » Agile Teams

## Agile Teams

“ Nothing beats an Agile Team

– SAFe Mantra

**Definition:** An Agile Team is a cross-functional group of typically ten or fewer individuals with all the skills necessary to define, build, test, and deliver increments of value.

## Summary

An Agile Team is self-directed and equipped with all the necessary skills to deliver value in increments. They can be technical, such as hardware or software teams, or they may be an Agile Team with a business focus. Agile Teams aim for rapid delivery and frequent customer feedback, iterating towards their goals. Multiple Agile Teams may form an Agile Release Train (ART), collaborating to develop a single solution, working from a shared ART backlog, and aligning to a common vision and roadmap. Each Agile Team includes two critical roles: the Product Owner and the Scrum Master/Team Coach.

## What is an Agile Team in SAFe?

An Agile Team is a cross-functional group of typically ten or fewer individuals with all the skills necessary to define, build, test, and deliver increments of value. Each team member is typically dedicated to one Agile Team.

Each Agile Team is self-directed and manages its work to meet its goals and the needs of its stakeholders and customers. This makes assigning tasks to specific team members unnecessary, giving teams and individuals more freedom and decision-making power and improving

engagement and enjoyment in the work. Creating long-lasting Agile Teams helps eliminate the inefficiencies and delays traditionally found in shorter-lived project-based teams. Leaders support Agile teams by providing guidance and freedom and recognizing high performance.

Agile Teams may be technical teams focused on building software or hardware products, business teams such as marketing, legal, or finance, or a blend of each. By quickly delivering work in small batches, all Agile Teams strive for fast learning and frequent customer feedback, adjusting their plans accordingly based on the results.

In SAFe, multiple Agile Teams work together as an Agile Release Train (ART) when a product is bigger or more complex than a single team can build. Collaborating with other Agile Teams, they build entire solutions working from a common ART backlog. All the Agile Teams on the ART are aligned to a common vision and roadmap and participate in ART events. Together, the teams on the ART continually optimize their practices, accelerating value delivery.

Great Agile Teams are more than just talented individuals. Agile Teams utilize ART and team events to make commitments alongside other teams on which the business can depend.

## How to organize Agile Teams in SAFe?

SAFe Principle #10, 'Organize around value,' guides enterprises to organize people and teams to accelerate customer value delivery.

The book "Team Topologies [1]" suggests organizing teams in four main ways that are commonly used within SAFe to accomplish this:

1. **Stream-aligned teams** are capable of delivering value directly to customers.
2. **Complicated subsystem teams** work on complex and very technical areas, removing the need for other teams to do this highly specialized work.
3. **Platform teams** build common services and tools that other teams use.
4. **Enabling teams** offer support and expertise to help other teams when needed.

Read more about Team Topologies:

Organizing Agile Teams and ARTs: Team Topologies at Scale

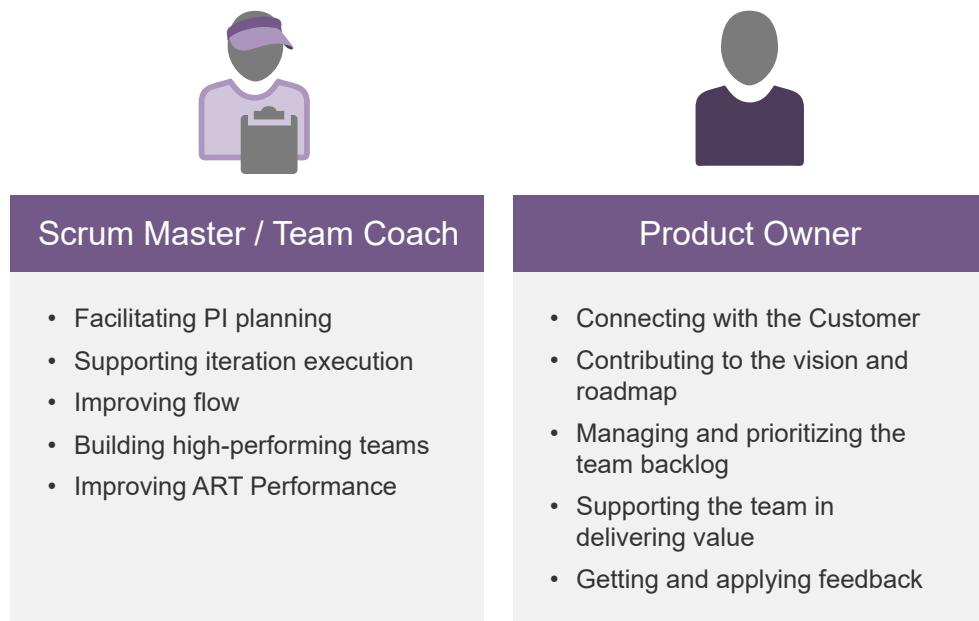
## Who is on an Agile Team in SAFe?

The roles on an Agile Team will depend on the type of work each Agile Team does. Importantly, each Agile Team should have all the individuals required to define, build, test, and release an increment of value in their context.

In addition, each Agile Team contains two specialty roles.

1. **The Product Owner (PO)** contributes to the vision and roadmap of the ART and works with the Agile Team to define and prioritize the team's work within the larger product and business setting.
2. **The Scrum Master / Team Coach (SM/TC)** implements and maintains Agile methods, boosts team performance, and collaborates with the Release Train Engineer (RTE) to support ART improvements and streamline value delivery.

The responsibilities of these roles are further outlined in Figure 1 below.



© Scaled Agile, Inc.

Figure 1. Agile Teams include two specialty roles with focused responsibility areas

Read more about the two Agile Team specialty roles:

[Scrum Master / Team Coach](#)

[Product Owner](#)

## How does an Agile Team operate in SAFe?

Agile teams typically apply SAFe Scrum or SAFe Team Kanban. Many will use a combination of these and adapt their practices over time.

SAFe Scrum and SAFe Team Kanban provide a set of practices that guide the team. These include events, techniques, and communication strategies that enable the progress of the work.

Most teams start their Agile journey by adopting SAFe Scrum. Practices like iteration planning, commitment to iteration goals, frequent retrospectives, a daily sync, and adhering to a short iteration timebox become routine.

However, some teams' work is better suited to responding to frequent and less plannable events. In this case, SAFe Team Kanban is often the preferred team operating model. SAFe Team Kanban is less dependent on iteration timeboxes, focusing more on a continuous flow of work through the backlog to the customer.

Regardless of the approach, all Agile Teams benefit from using Kanban boards to visualize and manage their backlogs and focus on improving flow.

Both methods are highly effective and are more alike than they are different. They both promote the delivery of value to customers more efficiently by:

- Working in small batches
- Keeping work-in-process (WIP) under control
- Removing delays
- Incorporating customer feedback to improve the product
- Reviewing and improving practices regularly

## What are the responsibilities of an Agile Team in SAFe?

Agile Teams fulfill five primary areas of responsibility, as shown in Figure 2.



© Scaled Agile, Inc.

Figure 2. Areas of responsibility of an Agile Team

Each is described in the sections below.

## Connecting with the Customer



Agile Teams focus on the customer by taking a Customer-Centric approach, which helps them fully grasp their experiences and needs. Agile Teams employ design thinking to understand the challenges and opportunities and develop appropriate solutions.

**Build Empathy with the Customer** – Agile Teams need to empathize with customers to more deeply understand their needs and foster stronger connections. To achieve this, Agile Teams should:

- Leverage Product Owners' expertise.
- Observe and communicate directly with customers.
- Participate in product support.
- Use telemetry to monitor product usage.

**Participate in Product Definition** – Team members use their customer insights to craft user stories and acceptance criteria in collaboration with the Product Owner (PO).

**Design and Execute Experiments** – Agile Teams learn more about their customers and the solutions they need by doing different experiments. This means they might research a topic deeply or build early versions of a product to get feedback fast.

## Planning the Work



Agile Teams plan their own work. Planning involves all team members and relies on collaboration and transparency. Effective planning facilitates alignment to a common goal while leveraging the flexibility and autonomy of each team member in achieving their objectives.

**Create a PI Plan** – PI Planning is the cadenced event where each Agile Team gains alignment with the rest of the teams on the ART for the upcoming PI. PI Planning provides the larger system view necessary to achieve shared goals. As a result of PI Planning, each team creates a set of Team PI Objectives. They also make enough of a story-level outline of the planned order of their work across iterations to feel confident in their objectives. This becomes their initial team backlog for the PI.

**Iterate Team Plans throughout the PI** – Once ART alignment has been established, teams perform shorter-term Iteration planning regularly during the PI. This planning aims to use the team and ART's learnings throughout the PI to plan the next increment of value.

**Refine the Team Backlog** – As knowledge emerges, teams continuously refresh and refine their backlog. The backlog identifies and prioritizes the upcoming work to deliver the team's committed value.

## Delivering Value



Agile Teams deliver new functionality to customers.

**Frequently integrate and test** – Teams must combine developed pieces of functionality and check them often. This helps uncover problems early and gives the teams enough time to fix them.

**Regularly synchronize with the rest of the ART** – While executing the PI, a team has multiple checkpoints with the other teams on the ART, such as ART Syncs and System Demos. These events help everyone see the progress toward current PI objectives and help the ART adjust as needed.

**Build the continuous delivery pipeline** – An effective Agile development process depends on a continuous delivery pipeline optimized for Continuous Exploration, Continuous Integration, and Continuous Deployment.

**Release frequently** – Some Agile Teams can release directly to the customer. These teams may establish their own release process in alignment with other teams on the ART. Major release

timings may be decided upon during PI Planning; routine deployments are often governed at the iteration level. Other releases can be event-driven.

## Getting Feedback



Development speed is closely related to how quickly and accurately a team can get feedback. Without this feedback, mistakes pile up, and solutions become ineffective and delayed. Both customer insights and technical feedback are crucial for making progress.

**Know the customer** – Direct customer feedback from users provides valuable insights for the Agile Team. Where direct interaction with customers and the Agile Team is not possible, the Product Owner often acts as a bridge to the customer.

**Frequently validate technical concerns** – To address technical concerns, teams must constantly check their assumptions about the product's architecture and how it's being built. This involves regular integration, testing, deployment, research, and prototyping to explore different technical approaches efficiently.

**Gather Data** – Data is crucial for Agile Teams to understand how users interact with the functionality that the team creates. Data identifies areas for improvement, helps prioritize upcoming work, proves and disproves assumptions, and ensures the resulting products meet customer needs. By analyzing data, Agile Teams gain insights that are unavailable through intuition alone.

## Improving relentlessly



Relentless improvement is a core value of SAFe. Agile teams constantly seek ways to improve their process and the outcomes they are responsible for. As a part of the improvement effort, Agile Teams do the following:

**Run routine improvement events** – Agile Teams use regular team-level retrospectives throughout the PI, most commonly every iteration. Teams use these events to identify ways they can improve their processes, practices, and behaviors. This focus helps each team to become more high-performing. Additionally, all teams on an ART participate in a joint Inspect and Adapt event to identify improvements that provide benefits across the entire ART for the upcoming PI.

**Improve some things immediately** – Some problems should be addressed as they occur, without waiting for the next improvement event. Addressing issues as they emerge is part of a culture of continuous improvement.

**Share and align with other Agile Teams** – Agile Teams share what they learn as they improve their working practices. This sharing promotes transparency and helps to create a learning culture across the ART and the company.

Read more about some of the practices used by Agile Teams:

[Continuous Delivery Pipeline](#)

[Customer Centricity](#)

---

## References

[1] Skelton, Matthew, and Manuel Pais. *Team Topologies: Organizing Business and Technology Teams for Fast Flow*. IT Revolution Press, 2019.

### In this article

[What is an Agile Team in SAFe?](#)

[How to organize Agile Teams in SAFe?](#)

[Who is on an Agile Team in SAFe?](#)

[How does an Agile Team operate in SAFe?](#)

[What are the responsibilities of an Agile Team in SAFe?](#)

### Key Takeaways

- Agile Teams are organized to deliver value to customers.
- Agile Release Trains are made up of multiple Agile Teams.
- Agile Teams are supported by Lean-Agile leaders
- Agile Teams have 5 critical responsibility areas

## Ready to start learning?

Use our finder to explore current offerings or learn more about a specific course

### Course

SAFe for Teams

[See available classes](#)

[Learn more about the course](#)

Last update: 15 October 2024

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

### Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

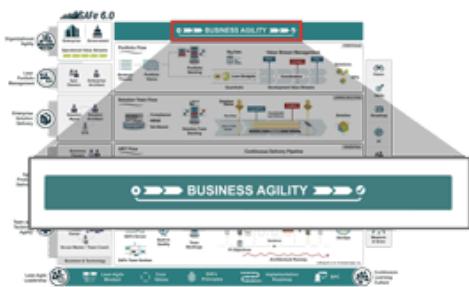
[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



## + Framework



“ The productivity of software delivery at enterprise organizations falls woefully behind that of the tech giants, and the digital transformations that should be turning the tide are failing to deliver business results.

—Mik Kersten, *Project to Product* [1]

## Business Agility

Business Agility is the ability to compete and thrive in the digital age by quickly responding to market changes and emerging opportunities with innovative, digitally-enabled business solutions.

Everything moves fast in the digital age. Customer desires, competitive threats, technology choices, business expectations, revenue opportunities, and workforce demands now happen at *blistering speeds*.

Today, achieving customer delight at the speed of the market requires validating innovations with customers and then ‘pivoting without mercy or guilt’ when the hypothesis needs to change. Moreover, significant technological advances are unlocking new ways to create this value. For example, [AI](#), [Big Data](#), [Cloud](#), and [DevOps](#) enable enterprises to expand their product lines, modernize their existing offerings, scale to mass markets, make better fact and insight-based decisions, and streamline solution development.

## Competing in the Age of Software

In her book *Technological Revolutions and Financial Capital*, Carlota Perez [2] explains the evolution of business, society, and financial cycles based on her analysis of five significant technological revolutions over the past three centuries. Her research begins with the *Industrial Revolution*, leading to the *Age of Steam and Railways*, the *Age of Steel and Heavy Engineering*, and the present *Age of Software and Digital*, as illustrated in Figure 1.

Perez concludes that these revolutions lead to massive social change, market disruption, and an all-new economic order. Indeed, these are *world-shaking* disruptions that typically occur once in a generation.

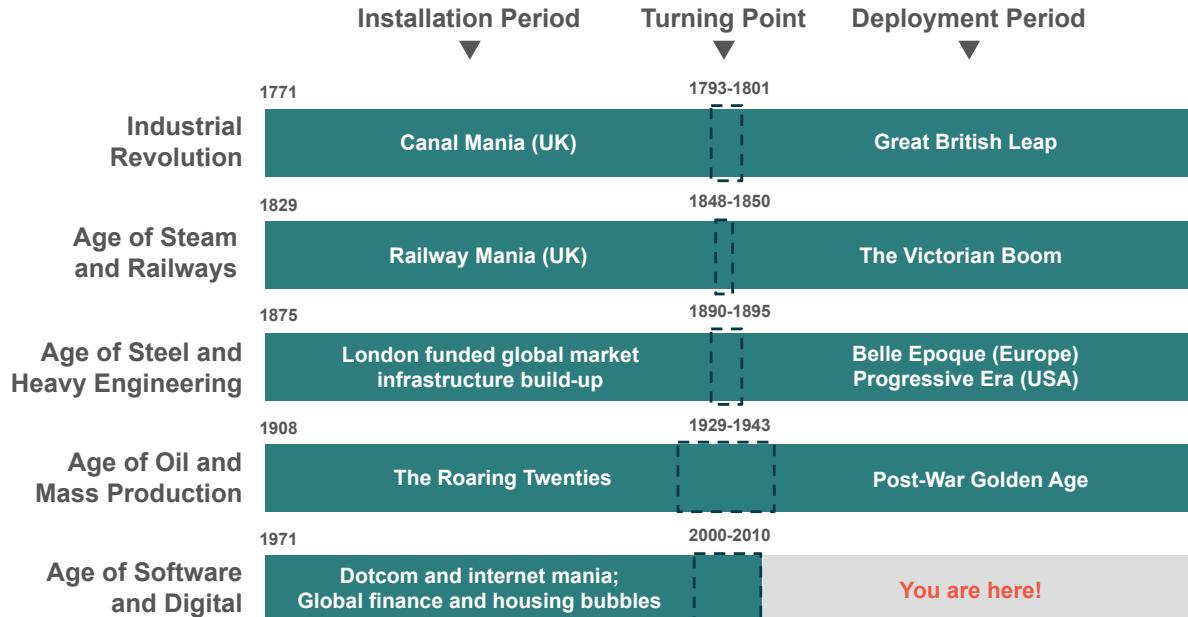


Figure 1. Technological revolutions change society

We are in the midst of one of those ages now, the deployment period of the **age of software and digital**. This period is when every business is a software business. Put simply, competing in this age requires large-scale software and system development capability that enables true business agility.

## Why Organizations Struggle to Achieve Business Agility

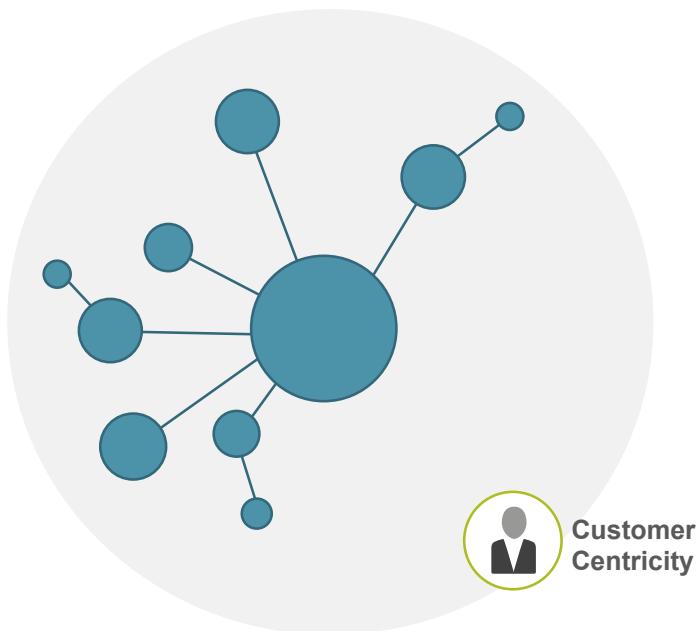
“*The organizations we created in the 20th century were designed much more for reliability and efficiency than for agility and speed.*

— John P. Kotter, *Accelerate* [3]

Most leaders in traditional organizations are aware of the digital disruption threat, yet many fail to transition to take their place in the next economy [4]. The question is, why?

## Organizations Start as a Fast Adaptive Network

As an organizational researcher and author, John Kotter illustrates in his book, *Accelerate: Building Strategic Agility for a Faster-Moving World*, successful enterprises don't start as large and cumbersome. Instead, they typically began as a fast-moving, adaptive network of motivated individuals focused on responding to the customer and the new business opportunity. Roles and reporting relationships are fluid, and people collaborate organically to identify customer needs, explore potential solutions, and deliver value in any way they can. In other words, it's an adaptive 'entrepreneurial network' of people working to leverage an opportunity (Figure 2).



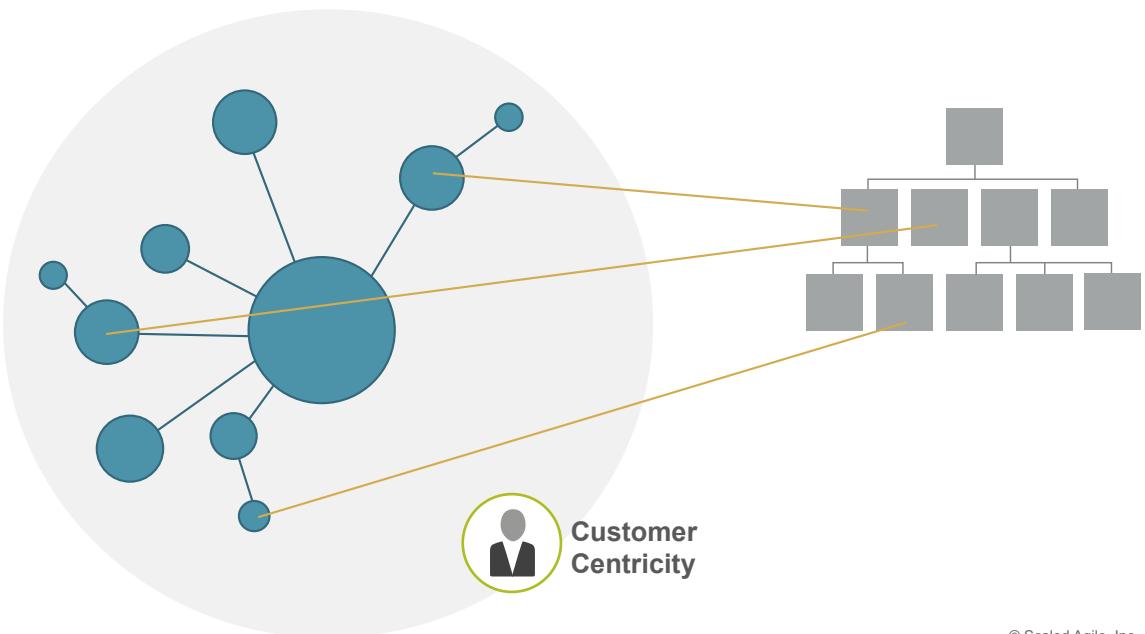
© Scaled Agile, Inc.

Figure 2. New enterprises operate as networks focused on the customer and new business opportunities

## Hierarchy Forms, then Grows and Grows

As the enterprise succeeds, it naturally wants to expand on its success and grow. This growth means that individual responsibilities must become clearer. As a result, the enterprise hires specialists to add expertise, creating new functional areas. Policies and procedures ensure legal adherence and compliance, driving repeatable, cost-efficient operations. The business starts to organize functionally to scale, causing silos to form. Meanwhile, operating in parallel, the network continues to seek new opportunities to deliver value (Figure 3).

Achieving larger economies of scale requires the hierarchy to grow. And it grows until it conflicts with the entrepreneurial network.

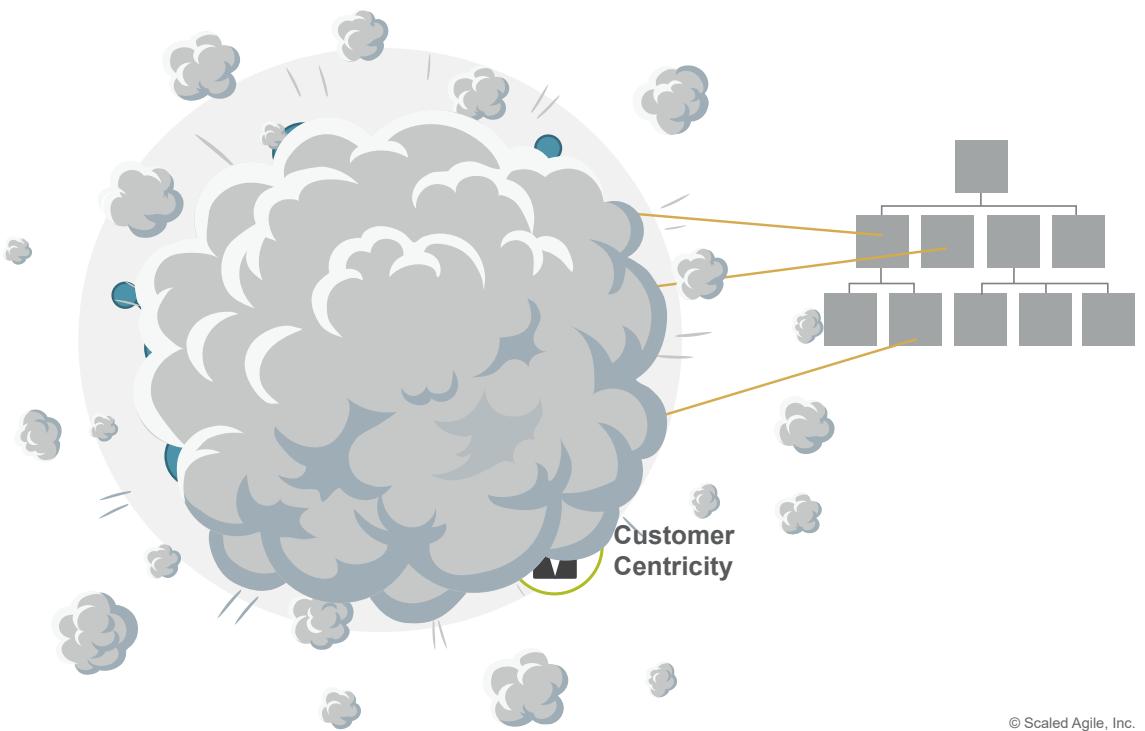


© Scaled Agile, Inc.

**Figure 3. Growing hierarchy running in parallel with an entrepreneurial network**

## The Hierarchy and Adaptive Network Collide

Eventually, the hierarchy collides with the faster-moving, more adaptive network. The result? The adaptive network gets crushed. The focus on the customer is often one of the main casualties (Figure 4.)



© Scaled Agile, Inc.

**Figure 4. Entrepreneurial network collides with a growing hierarchy**

Without the entrepreneurial network, the organization lacks the agility to respond when the customer needs shift dramatically or when a disruptive technology or competitor emerges. An urgent crisis erupts, and the company's survival is now at stake.

However, the organizational hierarchies built over the last fifty years have provided time-tested structures, practices, and policies. They support the recruiting, retention, and growth of thousands of employees across the globe. Simply put, they are still needed.

In addressing this dilemma, Kotter points out, "*The solution is not to trash what we know and start over but instead to reintroduce a more agile, network-like structure*" that operates in concert with the hierarchy to create what he calls a 'dual operating system.' This system, illustrated in Figure 5 and described in the following section, allows companies to capitalize on rapid-fire strategic challenges and retain their stability [3].

## The Solution: SAFe is the Second Operating System

The existing hierarchy, people, and management still have a purpose and largely remain in place. However, SAFe creates a second *virtual* operating system organized around [Development Value Streams](#) instead of functional silos (or departments) to form the entrepreneurial network.

Each development value stream creates one or more [Agile Release Trains \(ARTs\)](#) with a shared business and technology mission. Each ART plans, commits, develops, and deploys together. They are an integral part of the entrepreneurial network that develops innovative products ([Solutions](#) and services).

Although the management reporting structure in the hierarchy may remain largely the same, the teams in an ART are self-organizing and self-managing, and they no longer need daily task direction. ARTs are virtual organizations with the people needed to define, deliver, and operate the solution. This new virtual organization breaks down the traditional functional silos that inhibit flow and innovation.

By organizing the second operating system around value streams instead of departments, SAFe offers a way for organizations to focus on customers, products, innovation, and growth in harmony with their existing hierarchical structure.

Moreover, this operating system is *flexible*. It is built on time-tested Lean-Agile SAFe practices. It can organize and *quickly reorganize* without completely disrupting the existing hierarchy, as illustrated in Figure 5. That's what business agility demands.

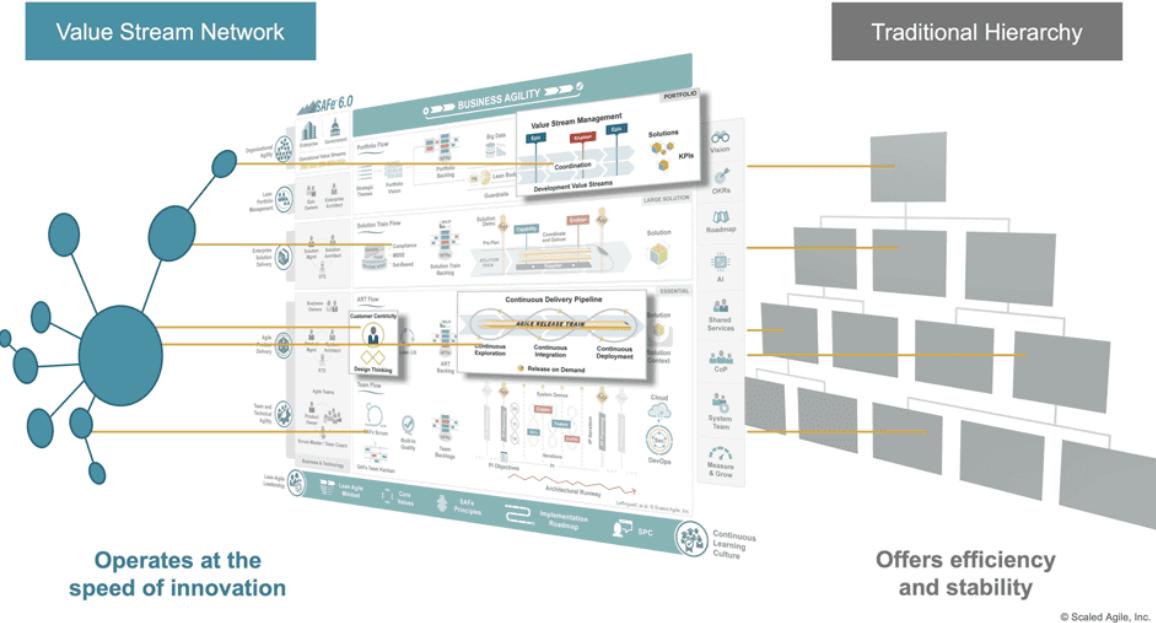


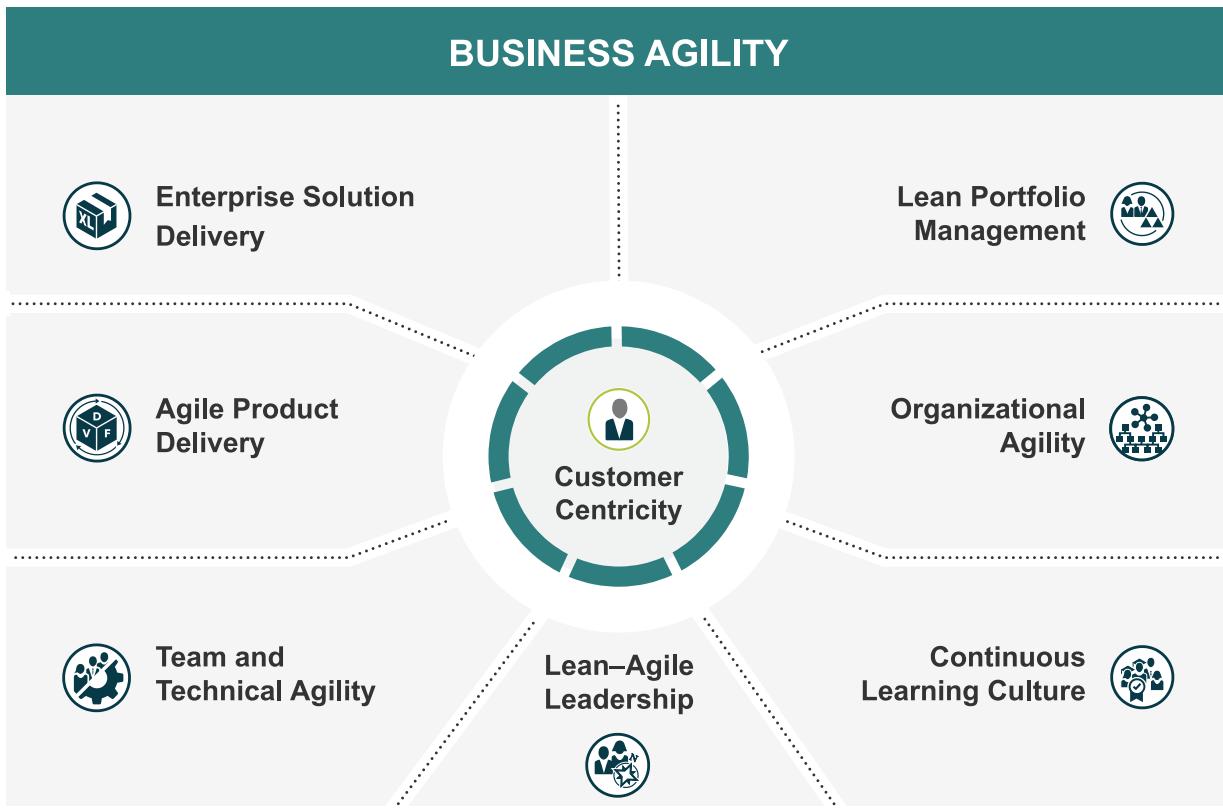
Figure 5. SAFe as the second organizational operating system

Responding to market changes and emerging opportunities is vital to surviving in the digital age, where disruption is the norm rather than the exception. Technological advances have drastically shifted the dynamics of the competitive game by opening up various ways to win in the market. The ability to respond quickly with innovative business solutions—what we call Business Agility—is the deciding factor between success and failure. Therefore, enabling business agility is a mission-critical goal for every organizational leader.

An organization must first understand and then apply the SAFe Business Agility Value Stream (BAVS) to deliver value faster to keep today's customers and win new ones. The BAVS helps organizations visualize the steps and implement the SAFe core competencies needed to move from identifying an opportunity to delivering customer value in the shortest possible time. The following sections describe the BAVS in greater detail.

## Core Competencies of Business Agility

Achieving business agility requires this dual operating system and a significant degree of expertise across SAFe's seven core competencies, as illustrated in Figure 6. Fortunately, they are part of SAFe, and as organizations adopt the Framework, they acquire the competencies incrementally over time. While each competency can deliver value by itself, true business agility is only present when the enterprise achieves significant mastery of all. It's a tall order, but the path is clear.



© Scaled Agile, Inc.

Figure 6. SAFe Core Competencies

Each competency is described in an article of that same name. You can easily navigate each competency from the SAFe Overview tab or the SAFe home page.

## Why the Business Agility Value Stream (BAVS)?

Digital transformation is advancing in almost all business processes. Technologies such as AI, big data, and cloud computing are unlocking possibilities for creating new customer value. New business opportunities emerge more frequently, and many have the potential to disrupt market incumbents. Companies that continuously leverage these technologies will acquire more customers and improve value to existing customers. Ultimately, they will dominate their markets. For example:

- Customers who can seamlessly fulfill their primary banking needs with an easy-to-use mobile app will remain loyal to that bank.
- Health organizations that pioneer virtual urgent care at home will earn the gratitude and loyalty of all those who need their help.
- A driver whose vehicle gets smarter and safer every day is more likely to remain loyal to that automotive manufacturer.

## Traditional Development Won't Get You There

In this new reality, competitiveness equates to rapidly delivering digitally enabled solutions. As Figure 7 illustrates, responding to a traditional, phase-gate development process may mean missing the opportunity.

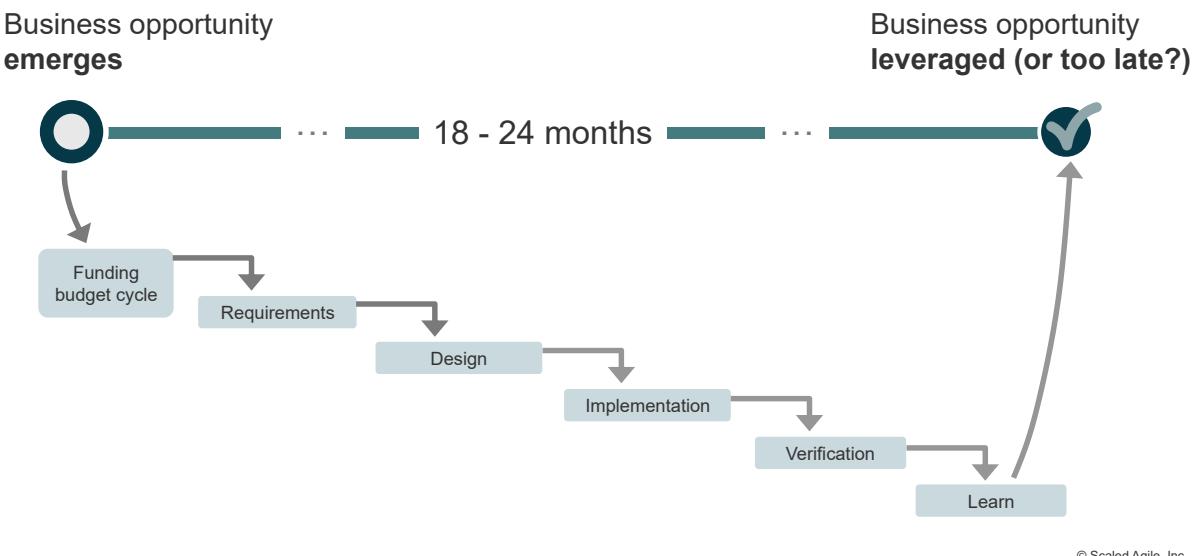


Figure 7. The traditional approach to leveraging a business opportunity is just too slow

Waiting for a funding cycle, big-design-upfront, and a long development cycle results in delayed technical and customer feedback. And more than likely, the learning at the end of this cycle isn't positive. Simply put, it is difficult, if not impossible, to understand and adapt to customer needs and quickly deliver a solution with a traditional development approach.

## Introducing the Business Agility Value Stream

Instead, what is needed is a rapid cycle of sensing and responding that helps the Enterprise navigate the unknowns and arrive at a desirable solution *before* the window of opportunity closes. This is the Business Agility Value Stream (BAVS), illustrated in Figure 8. And it is explicitly designed to foster rapid learning and enable more favorable business outcomes. By implementing SAFe, enterprises inherently develop the Lean, Agile, and DevOps capabilities that will allow incremental delivery at scale. Although these capabilities are essential, establishing true business agility requires flow to be cultivated and accelerated through the entire BAVS, from sensing an emerging opportunity to delivering the right solution (Figure 8).

## Business Agility Value Stream



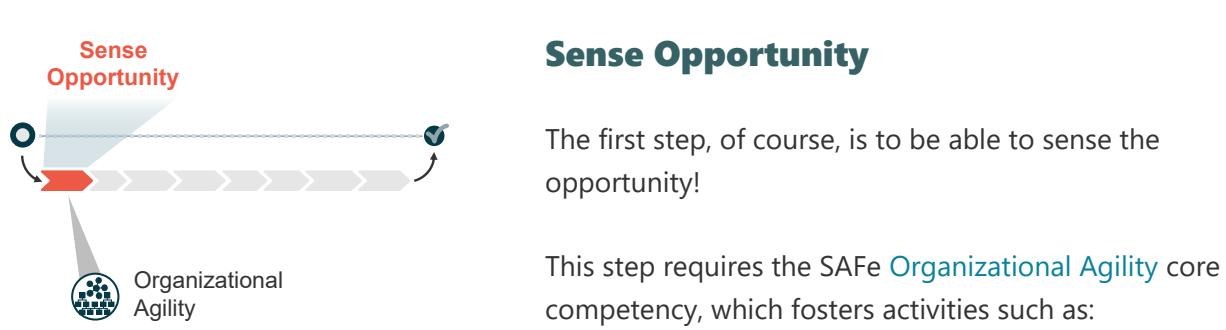
© Scaled Agile, Inc.

Figure 8. A faster way to leverage a new business opportunity

## Steps in the Business Agility Value Stream

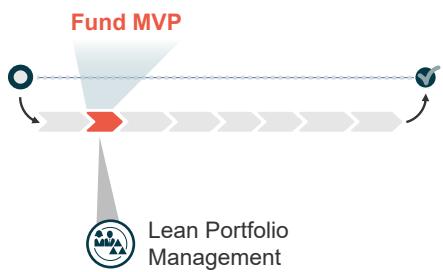
While the specific implementation of the BAVS depends on the business context (organizational, market, customer, technology, and solution), the steps and basic structure are essentially the same. In addition, the knowledge, skills, and behaviors required to succeed with each step of the BAVS are described in the related SAFe Core Competency, as we will see below.

This agility requires all functions, processes, activities, teams, and events from end to end to be aligned and optimized for maximum speed and quality. The following steps form this value stream:



- Market research
- Analysis of quantitative and qualitative data
- Direct and indirect customer feedback
- Direct observation of the customers in the marketplace

Most directly savvy, Lean-thinking leaders 'go see' and spend significant time where customers perform work. They return with current, relevant, direct, and specific information about the realities of their products and services and identify opportunities for innovative new solutions.

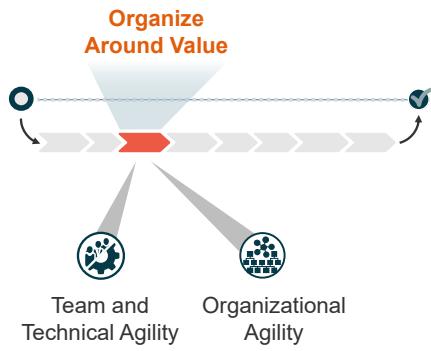


## Fund MVP

An enterprise must be able to respond quickly to these opportunities with nimble funding. [Lean Portfolio Management](#) (LPM) is the core competency needed to support this step.

With LPM, Lean Budgeting provides the ability to quickly allocate sufficient funds to build a Minimum Viable Product (MVP)—an early version of the solution used to evaluate the primary business hypothesis. The ‘Minimum’ in MVP refers to the low cost of the experiment needed to test the hypothesis and establish solution viability.

These funding decisions become visible and are addressed as the new initiative flows through the SAFe Portfolio Kanban system.

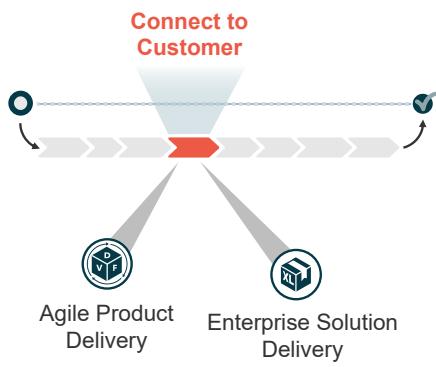


## Organize Around Value

The next step is to [organize](#)—or reorganize as necessary—to address the new opportunity. An MVP can often be built by existing Agile Teams or Agile Release Trains (ARTs) as the new work finds its way into their respective backlogs. However, creating an MVP may also involve modifying existing teams and ARTs. An entirely new development value stream may need

to be formed in a more extreme case.

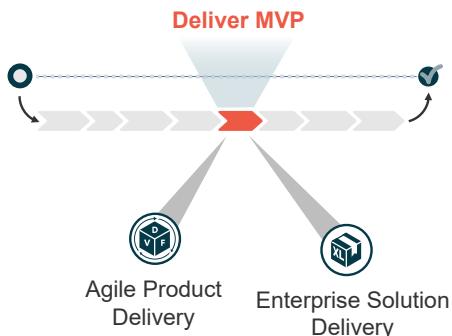
Two core competencies—[Team and Technical Agility](#) and [Organizational Agility](#)—enable this flexibility.



## Connect to Customer

Agile development is inherently focused on assuring a direct connection to the customer, and Customer Centricity is the mindset that underpins it. This way of doing business focuses on creating positive experiences for the customer across the enterprise’s complete set of products and services and throughout the entire customer journey.

Design Thinking provides the tools that help teams and ARTs achieve these ideals by empathizing with the user to design the right solution. [Agile Product Delivery](#) is the core competency that enables this connection.



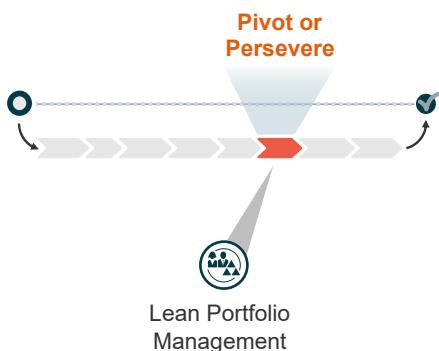
## Deliver MVP

The proof is in the doing. Agile teams and ARTs deliver the MVP iteratively and incrementally, following Lean-Agile practices.

However, there are differences in how teams and trains work on an MVP compared to evolving functionality in a mature solution. There is more risk and uncertainty

for a start. Unknowns may manifest in critical areas, including technology choices, implementation strategy, organizational expertise, deployment, operations, customer acceptance, and business benefits. More experimentation and even faster feedback are required. But that is exactly what SAFe is optimized for.

Depending on the scope of the solution, [Agile Product Delivery](#) and [Enterprise Solution Delivery](#) are the two core competencies that enable this step.

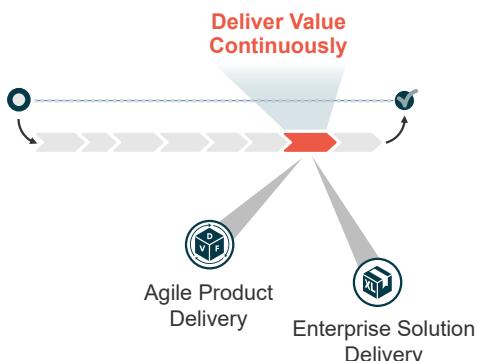


## Pivot or Persevere

The result of the MVP is a set of facts that support a decision regarding whether to proceed with further solution development. If the hypothesis is disproven, the organization accepts the sunk cost and moves on to other business opportunities. If the hypothesis proves beneficial, additional funding follows to enable further development. However, the MVP outcome is

not always a simple yes or no. The experiment may yield vital insights that reveal new alternative solutions.

This decision point is a crucial investment milestone and is a critical stage in the portfolio Kanban system. Again, Lean Portfolio Management is the core competency that enables this step.

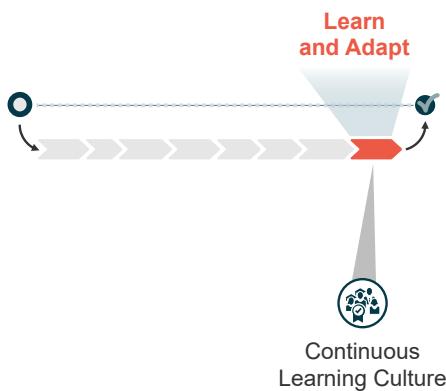


## Deliver Value Continuously

A successful MVP that confirms the hypothesis opens the gates to deliver value continuously with additional solution features. This process relies on Agile Product Delivery that fosters iterative and incremental development powered directly by the ART.

Building on DevOps, these practices include optimizing a Continuous Delivery Pipeline that ensures a steady flow of value and the ability to release on-demand to meet the needs of customers and businesses.

For some organizations, these solutions represent large, significant, and complex applications and cyber-physical systems that require thousands of developers and many capable suppliers to coordinate their efforts within a Solution Train. Enterprise Solution Delivery is the core competency enabling this step in this case.



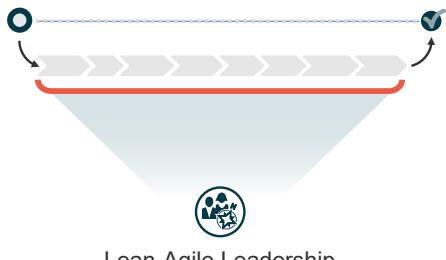
## Learn and Adapt

Learn and Adapt is not the final step. A single initiative rarely determines business outcomes. Instead, the enterprise learns from the BAVS and the process and adapts based on these learnings.

**Measurement** is an integral part of improvement. As Figure 9 illustrates below, three measurement domains—Competency, Flow, and Outcomes—provide critical

perspectives and measures of organizational performance that help identify impediments and opportunities for improvement.

The **Continuous Learning Culture** core competency is the primary driving force behind positive change. A learning organization has a sense of urgency, constantly looking for new business opportunities and improvements in existing processes and solutions. Indeed, the BAVS is an example of continuous learning, enabling continuous innovation. In addition, other forms of learning and adaptation happen through regular Inspect & Adapt events at every level of the SAFe operating system.



## Lean-Agile Leadership Enables the BAVS

None of this happens without **Lean-Agile Leadership**, the foundation of SAFe and the BAVS. The BAVS represents a new way of working for most enterprises, which is substantially different from the status quo and

affects most aspects of a modern enterprise.

Lean-Agile leaders view the organization as a dynamic set of business agility value streams that pursue and leverage critical business opportunities. And importantly, they lead the change to arrive at this new state. After that, they focus the organization on successful BAVS execution and improving BAVS performance over time. Only then can business agility be achieved.

# Measuring the Business Agility Value Stream

As Figure 9 illustrates, SAFe provides three measurement domains—*flow*, *outcomes*, and *competency*—suitable for measuring and improving any value stream, including the all-important BAVS.

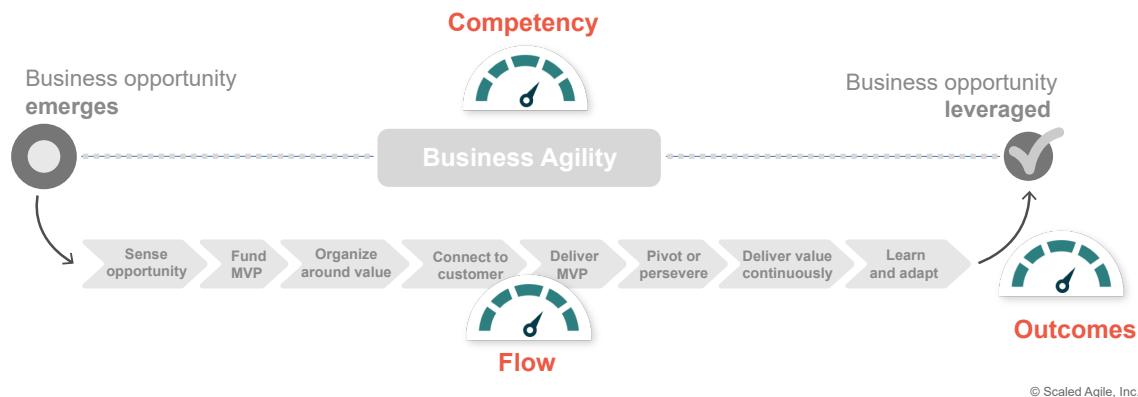


Figure 9. Three SAFe measurement domains support the goal of business agility

- **Flow** metrics help determine how fast the value stream creates and delivers value, represented by flow *distribution*, *velocity*, *time*, *load*, *efficiency*, and *predictability*.
- **Outcomes** metrics help ensure the solution delivered benefits the customer and the business. [Value Stream KPIs](#) are primarily used to measure these outcomes.
- **Competency** measures evaluate organizational proficiency on two levels:
  1. The *SAFe Business Agility Assessment* offers business and portfolio stakeholders a way to assess their overall progress.
  2. The individual *SAFe Core Competency Assessments* help teams and ARTs improve their technical and business practices to achieve the portfolio's goal.

See the [Measure and Grow](#) article for more details.

The road to business agility is long and never-ending. Measuring it helps enterprises understand where they are on their journey and reminds them to celebrate small successes.

## Summary

Welcome to the age of software and digital, where business agility will be the most significant factor in deciding the winners and losers in the new economy:

- Lean-Agile *commercial* businesses will create higher profits, increase employee engagement, and more thoroughly satisfy customer needs
- Lean-Agile *nonprofits* will build resilience, sustainability, and the alignment needed to fulfill their mission
- Lean-Agile *governments* will deliver systems that better ensure the public's safety, economy, and general welfare

These three market segments depend on delivering innovative business solutions faster and more efficiently than ever. Each will need a dual operating system: a hierarchy for efficiency and scale and a second, customer-centric network operating system that delivers innovative solutions. SAFe's seven core competencies will enable this all-important dual operating system. Those who master these competencies will survive and thrive in the digital age.

---

## Learn More

[1] Kersten, Mik. *Project to Product: How to Survive and Thrive in the Age of Digital Disruption with the Flow Framework*. IT Revolution Press, 2018.

[2] Perez, Carlota. *Technological Revolutions and Financial Capital: The Dynamics of Bubbles and Golden Ages*. Edward Elgar Publishing, 2003.

[3] Kotter, John P. *Accelerate: Building Strategic Agility for a Faster-Moving World*. Harvard Business Review Press, 2014.

[4] Thangavelu, Poonkulali. *Companies That Failed to Innovate and Went Bankrupt*. Investopedia, 2022. <https://www.investopedia.com/articles/investing/072115/companies-went-bankrupt-innovation-lag>.

Last update: 13 July 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## **Training**

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## **Content & Trademarks**

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## **Scaled Agile, Inc**

### **Contact Us**

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### **Business Hours**

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



+ Framework

ENGLISH (US) ▾



“*The impression that ‘our problems are different’ is a common disease that afflicts management the world over. They are different, to be sure, but the principles that will help to improve the quality of product and service are universal in nature.*

—W. Edwards Deming, *Out of the Crisis* [1]

## SAFe Lean-Agile Principles

SAFe is based on ten immutable, underlying [Lean-Agile](#) principles. These tenets and economic concepts inspire and inform the roles and practices of SAFe.

**#1 Take an economic view**

---

**#2 Apply systems thinking**

---

**#3 Assume variability; preserve options**

---

**#4 Build incrementally with fast, integrated learning cycles**

---

**#5 Base milestones on objective evaluation of working systems**

---

**#6 Make value flow without interruptions**

---

**#7 Apply cadence, synchronize with cross-domain planning**

---

**#8 Unlock the intrinsic motivation of knowledge workers**

---

**#9 Decentralize decision-making**

---

**#10 Organize around value**

© Scaled Agile, Inc.

Figure 1. SAFe Lean-Agile Principles

## Why the Focus on Principles?

Building enterprise-class software and cyber-physical systems are among the most complex challenges our industry faces today. And, of course, the enterprises that build these systems are also increasingly sophisticated. They are bigger and more distributed than ever. Mergers and acquisitions, distributed multinational (and multilingual) development, offshoring, and rapid growth are all part of the solution. But they're also part of the problem.

Fortunately, we have an amazing and growing body of knowledge that can help. It includes Agile principles and methods, Lean and systems thinking, product development flow practices, and Lean processes. Thought leaders have traveled this path before us and left a trail in hundreds of books and references to draw on.

The goal of SAFe is to synthesize this body of knowledge, along with the lessons learned from hundreds of deployments. This creates a system of integrated, proven practices that have improved employee engagement, time-to-market, solution quality, and team productivity. Given

the complexities, however, there's no off-the-shelf solution for the unique challenges each enterprise faces. Not every SAFe recommended practice will apply equally in every circumstance. This is why we work hard to ensure that SAFe practices are grounded in fundamentally stable principles. That way, we can be confident the practices apply in most situations.

And if those practices do fall short, the underlying principles will guide the teams to make sure that they are moving continuously on the path to the goal of the lean: "shortest sustainable lead time, with best quality and value to people and society." There is value in that, too.

SAFe is based on ten fundamental concepts that have evolved from Agile principles and methods, Lean product development, systems thinking, and observation of successful enterprises. Each is described in detail in an article by that principle's name. In addition, the embodiment of the principles appears throughout the Framework. They are summarized in the following sections, and each has a full article behind the link.

## #1 – Take an economic view

Delivering the 'best value and quality for people and society in the shortest sustainable lead time' requires a fundamental understanding of the economics of building systems. Everyday decisions must be made in a proper economic context. This includes the strategy for incremental value delivery and the broader economic framework for each value stream. This framework highlights the trade-offs between risk, Cost of Delay (CoD), manufacturing, operational, and development costs. In addition, every development value stream must operate within the context of an approved budget and be compliant with the guardrails which support decentralized decision-making.

## #2 – Apply systems thinking

Deming observed that addressing the challenges in the workplace and the marketplace requires an understanding of the systems within which workers and users operate [2]. Such systems are complex, and they consist of many interrelated components. But optimizing a component does not optimize the system. To improve, everyone must understand the larger aim of the system. In SAFe, systems thinking is applied to the system under development, as well as to the organization that builds the system.

## #3 – Assume variability; preserve options

Traditional design and life cycle practices encourage choosing a single design-and-requirements option early in the development process. Unfortunately, if that starting point proves to be the wrong choice, then future adjustments take too long and can lead to a suboptimal design. A better approach is to maintain multiple requirements and design options for a longer period in the development cycle. Empirical data is then used to narrow the focus, resulting in a design that creates optimum economic outcomes.

## #4 – Build incrementally with fast, integrated learning cycles

Developing solutions incrementally in a series of short iterations allows for faster customer feedback and mitigates risk. Subsequent increments build on the previous ones. Since the ‘system always runs,’ some increments may serve as prototypes for market testing and validation; others become minimum viable products (MVPs). Still others extend the system with new and valuable functionality. In addition, these early, fast feedback points help determine when to ‘pivot’ where necessary to an alternate course of action.

## #5 – Base milestones on objective evaluation of working systems

Business owners, developers, and customers have a shared responsibility to ensure that investment in new solutions will deliver economic benefits. The sequential, phase-gate development model was designed to meet this challenge, but experience shows that it does not mitigate risk as intended. In Lean-Agile development, integration points provide objective milestones at which to evaluate the solution throughout the development life cycle. This regular evaluation provides the financial, technical, and fitness-for-purpose governance needed to ensure that a continuing investment will produce a commensurate return.

## #6 – Make value flow without interruptions

The third principle in Lean Thinking is to ‘make value flow without interruptions.’ Doing so requires an understanding of what flow is, what the various properties of a flow system are, and how these properties can accelerate or impede the flow of value through any particular system. Principle #6 highlights the eight common properties of a flow-based system and provides specific recommendations for eliminating impediments to flow.

## #7 – Apply cadence, synchronize with cross-domain planning

Cadence creates predictability and provides a rhythm for development. Synchronization causes multiple perspectives to be understood, resolved and integrated at the same time. Applying development cadence and synchronization, coupled with periodic cross-domain planning, provides the mechanisms needed to operate effectively in the presence of inherent development uncertainty.

## #8 – Unlock the intrinsic motivation of knowledge workers

Lean-Agile leaders understand that ideation, innovation, and employee engagement are not generally motivated by individual incentive compensation. Such individual incentives can create internal competition and destroy the cooperation necessary to achieve the larger aim of the system. Providing autonomy and purpose, minimizing constraints, creating an environment of mutual influence, and better understanding the role of compensation are keys to higher levels of employee engagement. This approach yields better outcomes for individuals, customers, and the enterprise.

## #9 – Decentralize decision-making

Achieving fast value delivery requires decentralized decision-making. This reduces delays, improves product development flow, enables faster feedback, and creates more innovative solutions designed by those closest to the local knowledge. However, some decisions are strategic and global and have economies of scale that justify centralized decision-making. Since both types of decisions occur, creating a reliable decision-making framework is a critical step in empowering employees and ensuring a fast flow of value.

## #10 – Organize around value

Many enterprises today are organized around principles developed during the last century. In the name of intended efficiency, most are organized around functional expertise. But in the digital age, the only sustainable competitive advantage is the speed with which an organization can respond to the needs of its customers with new and innovative solutions. These solutions require cooperation amongst all the functional areas, with their incumbent dependencies, handoffs, waste, and delays. Instead, Business Agility demands that enterprises organize around value to deliver more quickly. And when market and customer demands change, the enterprise must quickly and seamlessly reorganize around that new value flow.

---

## Learn More

[1] Deming, W. Edwards. *Out of the Crisis*. The MIT Press, 2018.

[2] Deming, W. Edwards. *The New Economics for Industry, Government, Education*. The MIT Press, 2018.

Last update: 4 May 2023

## **Framework**

[Download SAFe Posters & Graphics](#)

[Blog](#)

## **Training**

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## **Content & Trademarks**

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## **Scaled Agile, Inc**

### **Contact Us**

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### **Business Hours**

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)



+ Framework

ENGLISH (US) ▾



“Agility is principally about mindset, not practices.

—Jim Highsmith, *Agile Project Management* [1]

## Lean-Agile Mindset

The Lean-Agile Mindset is the combination of beliefs, assumptions, attitudes, and actions of SAFe leaders and practitioners who embrace the concepts of Lean Thinking and the Agile Manifesto.

(Courtesy of Womack & Jones from *Lean Thinking*, and the *Agile Manifesto* [2,3])

It's the personal, intellectual, and leadership foundation for adopting and applying SAFe principles and practices.

The Lean-Agile mindset forms the cornerstone of a new way of working and an enhanced company culture that enables Business Agility. It provides leaders and change agents with the tools needed to drive a successful SAFe transformation, helping individuals and enterprises achieve their goals.

## Details

### Mindset Awareness and Openness to Change

What exactly is a 'mindset?' A mindset is a mental lens through which we view the world around us. It is how the human brain simplifies, categorizes, and interprets the vast amount of information it receives daily. We form our mindsets through a lifetime of structured learning (classes, reading) and unstructured lessons (life events, work experience). They reside in the subconscious mind and

manifest themselves as deeply held beliefs, attitudes, assumptions, and influences. Consequently, individuals are often unaware of how their mindsets influence how they carry out their responsibilities and interact with others. While many mindsets are positive and serve us well, others may need to change over time [4].

So how can mindsets be changed? It begins with an awareness of one's current mindsets and how they were formed. It's also vital to cultivate the belief that mindsets can be developed and improved (a 'growth' mindset, as illustrated in Figure 1).

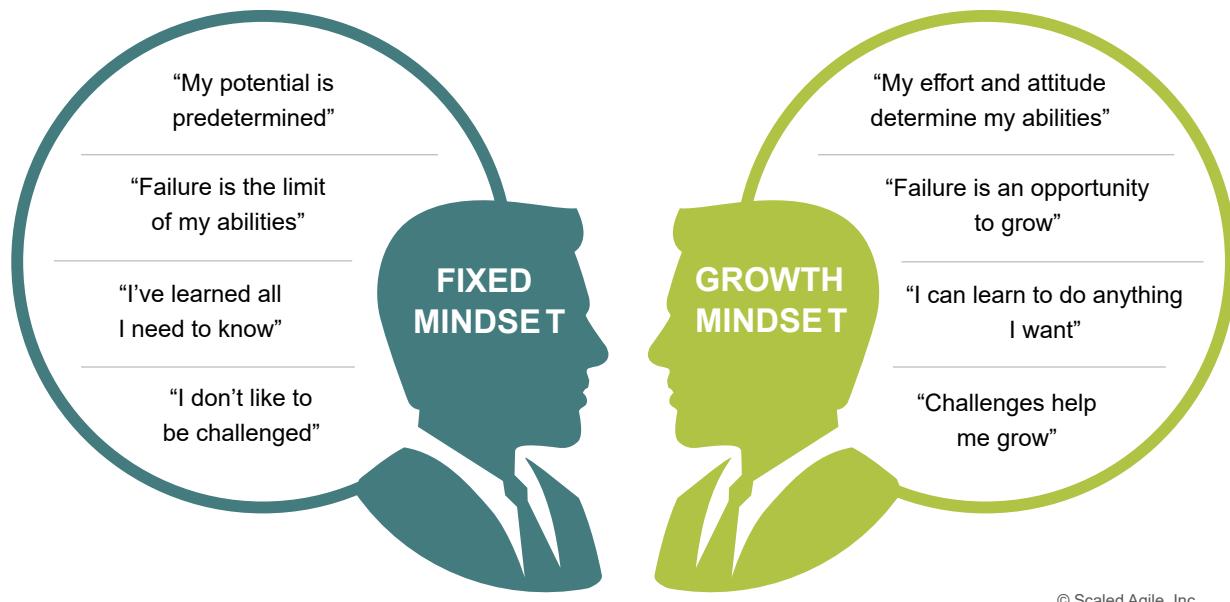


Figure 1. Adopting a new mindset requires a belief that new abilities can be developed with effort

Changing mindsets is a vital topic in transitioning to SAFe because, too often, leaders and practitioners in organizations go through the motions of mimicking SAFe practices and using SAFe terms without internalizing and embracing the underlying values and principles that truly represent a new way of working. This 'SAFe in name only' approach may produce some small successes in the short term. However, in the long term, such a shallow adoption of the Lean-Agile mindset will inevitably fail to produce the real, long-lasting business results leaders hoped for when they decided to 'go SAFe.'

To fully embrace SAFe requires a *growth* mindset open to learning the core values and principles of two primary underlying bodies of knowledge: Lean Thinking and Agile. Each has a rich and deep history of published guidance and case studies. Their respective values and principles need to be understood and practiced so that the ideals of both Lean and Agile permeate the organization's language, practices, and decision-making. Ultimately, it simply becomes 'our way of working' and is deeply ingrained in the culture of the enterprise.

The following two sections describe the key elements of Lean Thinking and Agile (summarized in Figure 2) that form the basis of the Lean-Agile mindset.

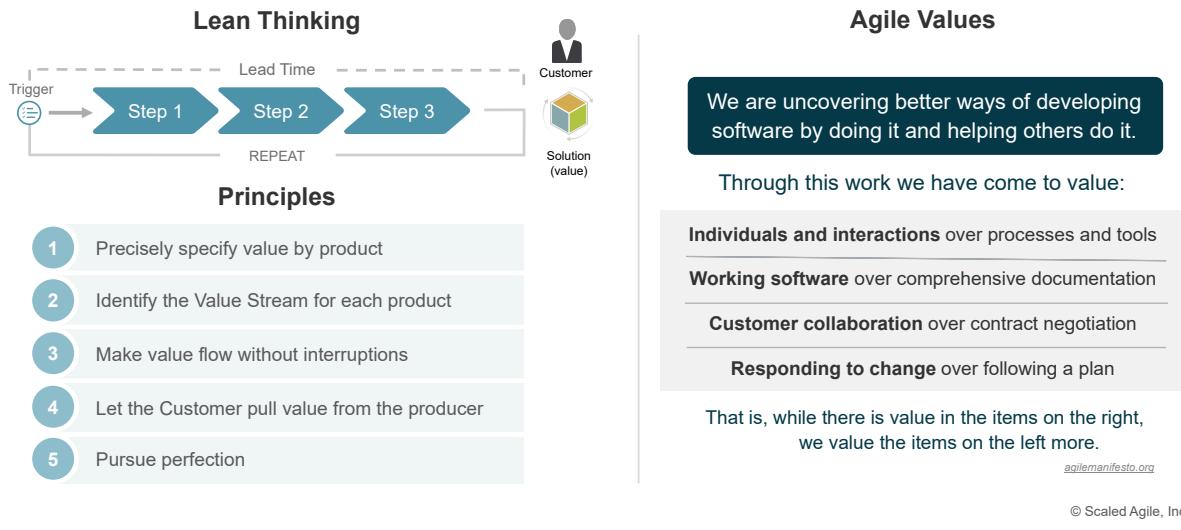


Figure 2. Lean Thinking and Agile are the core building blocks of SAFe

## Lean Thinking

‘ *Adopting Lean product development can double labor productivity through the entire system, cut time-to-market for new products in half, and enable a wider variety of products within product families to be offered at very modest additional cost.*

—James Womack & Daniel Jones, *Lean Thinking* [2]

Initially derived from Lean manufacturing, the principles and practices of Lean thinking as applied to software, product, and systems development are now deep and extensive [5]. For example, Ward [6], Reinertsen [7], Poppendieck [8], Kersten [9], Leffingwell [10], and others have described aspects of Lean thinking, placing many of the core principles and practices within a product development context. Applying Lean Thinking to product development, thereby shifting from the traditional batch-and-queue production system to continuous flow with an effective pull by the customer, can lead to dramatic improvements.

Lean thinking can be summarized as shown in Figure 3:



© Scaled Agile, Inc.

**Figure 3. The core principles of Lean Thinking**

As Figure 3 illustrates, the goal of Lean Thinking is to deliver the maximum value (a solution) to the customer in the shortest sustainable lead time from the trigger (the identification of the need or opportunity) to the point at which the customer receives the value. How value is created also matters. High quality, respect for people and society, high morale, safety, and customer delight are also essential goals and benefits of Lean Thinking. Achieving these goals requires applying the five basic principles of Lean, illustrated in Figure 3 and described in the following sections.

## Precisely specify value by specific product

Every enterprise is built to deliver value. Value can only be defined by the ultimate customer. And it's only meaningful when expressed in terms of a specific product (a good or a service, and often both at once) that meets the customer's needs at a specific price at a specific time. Therefore, the first principle of Lean Thinking underscores the importance of understanding customers' needs and quantifying the value inherent in the solutions delivered to them. The solution itself holds the value—not the project, initiative, or process that produces it—and the customer ultimately determines that value.

## Identify the Value Stream for each product

Once 'value' is defined for each product and type of customer, the following principle in Lean Thinking is to articulate how the enterprise creates that value, from identifying a need or opportunity to delivering the solution. This flow of work is the value stream and contains all the people, processes, tools, and information necessary to deliver value. Delays anywhere in this system result in delayed delivery of value to customers.

## Make value flow without interruptions

The third principle in Lean Thinking is establishing a continuous, uninterrupted flow of work that supports incremental value delivery based on constant feedback and adjustment. Enabled by [Built-In Quality](#) practices, relentless improvement, and evidence-based governance, continuous flow enables faster, sustainable value delivery.

Achieving a continuous flow of value requires applying and understanding the eight fundamental properties of flow: visualizing and limiting work-in-process (WIP), addressing bottlenecks, minimizing handoffs and dependencies, getting fast feedback, working in small batches, managing queue lengths, optimizing time ‘in the zone,’ and remediating legacy policies and practices. These flow properties are described in greater detail in the [SAFe Principle 6](#) article and in the [Team Flow](#), [ART Flow](#), [Solution Train Flow](#), and [Portfolio Flow](#) articles. The SAFe [Core Values](#) and [SAFe Principles](#) help teams achieve a continuous flow of value at scale in large, complex enterprises.

## Let the customer pull value from the producer

The next Lean principle guides organizations to configure value streams to deliver solutions that customers *pull* into the market based on their actual needs rather than solutions that teams *push* into the market based on what they ‘think’ customers need. This is key to calibrating the capacity of the value stream. Too much capacity compared to market pull results in waste, which is antithetical to Lean thinking. Conversely, too little capacity creates bottlenecks and delays, defeating the intent to provide the customer with a continuous flow of value.

## Pursue perfection

The final principle of Lean Thinking is expressed as ‘pursue perfection.’ It reflects that no matter how closely the first five principles are followed, creating a fast and effective flow of value is not a one-time activity. Market dynamics, customer needs, and available technologies are just some of the many factors that can require value streams to be refined and, in some cases, rebuilt entirely.

## Agile

‘ *Agile is an attitude, not a technique with boundaries. An attitude has no boundaries, so we wouldn’t ask ‘can I use Agile here,’ but rather ‘how would I act in the Agile way here,’ or ‘how Agile can we be, here?’*

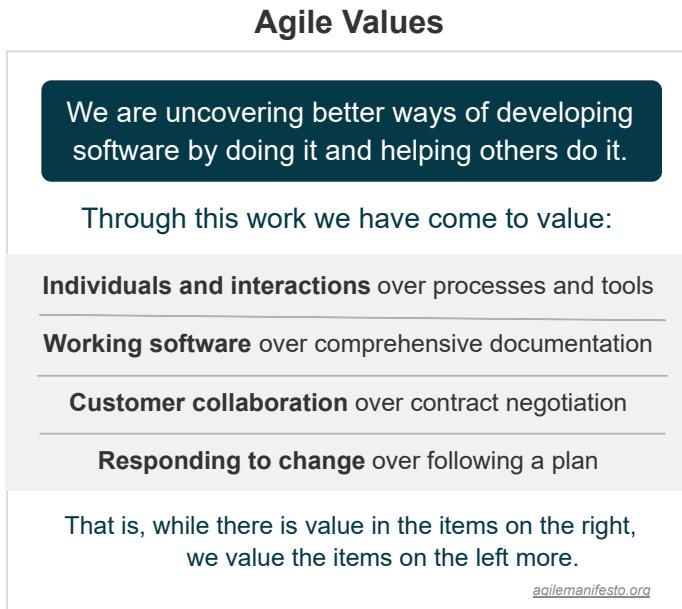
—Alistair Cockburn, *Agile Software Development* [11]

In the 1990s, some lighter-weight and more iterative development methods emerged in response to the many challenges of waterfall processes. In 2001, many leaders of these frameworks came

together to express their shared values and beliefs in the Manifesto for Agile Software Development. This turning point clarified the new approach and started to bring the benefits of these innovative methods to the whole development industry [3]. Since the Manifesto was first published, Agile has been adopted by domains outside of software development, including hardware systems, infrastructure, operations, and support. More recently, business teams outside of technology have also embraced Agile principles for planning and executing their work.

## The Values of Agile

Agile is built on the value statement shown in Figure 4. For the remainder of the description of Agile Values and Principles, as initially outlined in the Agile Manifesto, readers can expand each use of the term 'software' to include the working output of any Agile team, regardless of the domain.



© Scaled Agile, Inc.

Figure 4. Agile Values

### Individuals and Interactions over Processes and Tools

Deming notes, "If you can't describe what you are doing as a process, then you don't know what you are doing." [12] So, Agile processes in frameworks like Scrum, Kanban, and SAFe matter. However, a process is only a means to an end. When we're captive to a process that isn't working, it creates waste and delays. So, favor individuals and interactions, then modify processes accordingly. Tools are valuable but should supplement, rather than replace, face-to-face communication.

### Working 'Software' over Comprehensive Documentation

Documentation is important and has value. But creating documents to comply with potentially outdated corporate governance models has limited value. As part of a change program, governance, often captured by documentation standards, needs to be updated to reflect the Lean-Agile way of working. Rather than create detailed documentation too early—especially the wrong kind—it's more valuable to show customers working software, systems, and so on to get their feedback. Therefore, favor measuring progress by evaluating tangible work products. And document only what's truly needed.

## Customer Collaboration over Contract Negotiation

Customers are the ultimate deciders of value, so their close collaboration is essential in pursuing business agility. Contracts are often necessary to convey each party's rights, responsibilities, and economic concerns—but recognize that contracts can over-regulate what to do and how to do it. They don't replace regular communication, collaboration, and trust, no matter how well they're written. Instead, contracts should be win-win propositions. Win-lose contracts usually result in poor economic outcomes and distrust, creating contentious short-term relationships instead of long-term business partnerships. Instead, favor customer collaboration over contract negotiation.

## Responding to Change over Following a Plan

Change is a reality in the digital age and essential to achieving agility. The strength of Lean-Agile is in how it embraces change. As the system evolves, so does the understanding of the problem and the solution domain. Business stakeholder knowledge also improves over time, and customer needs evolve as well. Indeed, those changes add value to our system.

Of course, planning is an essential part of Agile. In fact, Agile teams and teams-of-teams plan more often and more continuously than their waterfall counterparts. However, plans must adapt as new learning occurs, new information becomes visible, and the situation changes. Worse, evaluating success by measuring conformance to a plan drives the wrong behaviors (for example, following a plan in the face of evidence that the plan is not working).

## Agile Principles

Agile has 12 self-explanatory principles that support its values, as shown in Figure 5. These principles take Agile Values a step further and specifically describe what it means to be Agile.

## Agile Principles

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity—the art of maximizing the amount of work not done—is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

[agilemanifesto.org](http://agilemanifesto.org)

© Scaled Agile, Inc.

Figure 5. Principles of the Agile Manifesto

The combination of these values and principles creates the essence of Agile. There is overwhelming evidence from success stories in all industries across every geography demonstrating the extraordinary business and personal benefits of this new way of thinking and working. We are grateful for it.

## Applying Lean Thinking and Agile in SAFe

Collectively, the values and principles of Lean Thinking and Agile form the DNA of everything contained within SAFe. All the roles, practices, events, and artifacts in SAFe are designed to provide practical guidance for adopting the combination of these two bodies of knowledge as the new way of working throughout the enterprise.

Thousands of implementations of SAFe over the last decade have shown that Lean Thinking and Agile principles and practices have unique implications when applied at scale. For example,

providing an uninterrupted flow of value within the context of a single Agile team will look different than when this same principle is applied to an entire portfolio. Yet the principle is equally important in both cases. The implications of Lean and Agile at scale have been captured in the SAFe Core Values and SAFe Principles articles.

---

## Learn More

- [1] Highsmith, Jim. *Agile Project Management: Creating Innovative Products*. Addison-Welsley Professional, 2009.
- [2] Womack, James P., and Daniel T. Jones. *Lean Thinking: Banish Waste and Create Wealth in Your Corporation*. Free Press, 2003.
- [3] *Manifesto for Agile Software Development*. <http://AgileManifesto.org/>
- [4] Dweck, Carol S. *Mindset: The New Psychology of Success*. Random House Publishing, 2007.
- [5] Womack, James P., Daniel T. Jones, and Daniel Roos. *The Machine That Changed the World: The Story of Lean Production—Toyota’s Secret Weapon in the Global Car Wars That Is Revolutionizing World Industry*. Free Press, 2007.
- [6] Ward, Allen C., and Durward K. Sobeck II. *Lean Product and Process Development*. Lean Enterprise Institute, 2014.
- [7] Reinertsen, Donald G. *The Principles of Product Development Flow: Second Generation Lean Product Development*. Celeritas Publishing, 2009.
- [8] Poppendieck, Mary, and Tom Poppendieck. *Implementing Lean Software Development: From Concept to Cash*. Addison-Wesley Professional, 2006.
- [9] Kersten, Mik. *Project to Product: How to Survive and Thrive in the Age of Digital Disruption with the Flow Framework*. IT Revolution Press, 2018.
- [10] Leffingwell, Dean. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*. Addison-Wesley Professional, 2010.
- [11] Cockburn, Alistair. *Agile Software Development: The Cooperative Game*. Addison-Wesley Professional, 2006.
- [12] Deming, W. Edwards. *Out of the Crisis*. The MIT Press, 2018.

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

## Scaled Agile, Inc

### Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

### Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

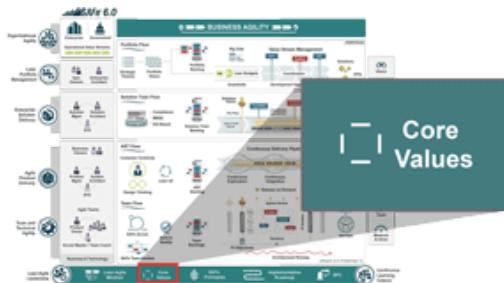
[Code of Conduct](#)

[Remote Training Policy](#)



+ Framework

ENGLISH (US) ▾



“Find people who share your values, and you'll conquer the world together.

—John Ratzenberger & Joel Engel, *We've Got it Made in America* [1]

## Core Values

The four Core Values of alignment, transparency, respect for people, and relentless improvement represent the foundational beliefs that are key to SAFe's effectiveness.

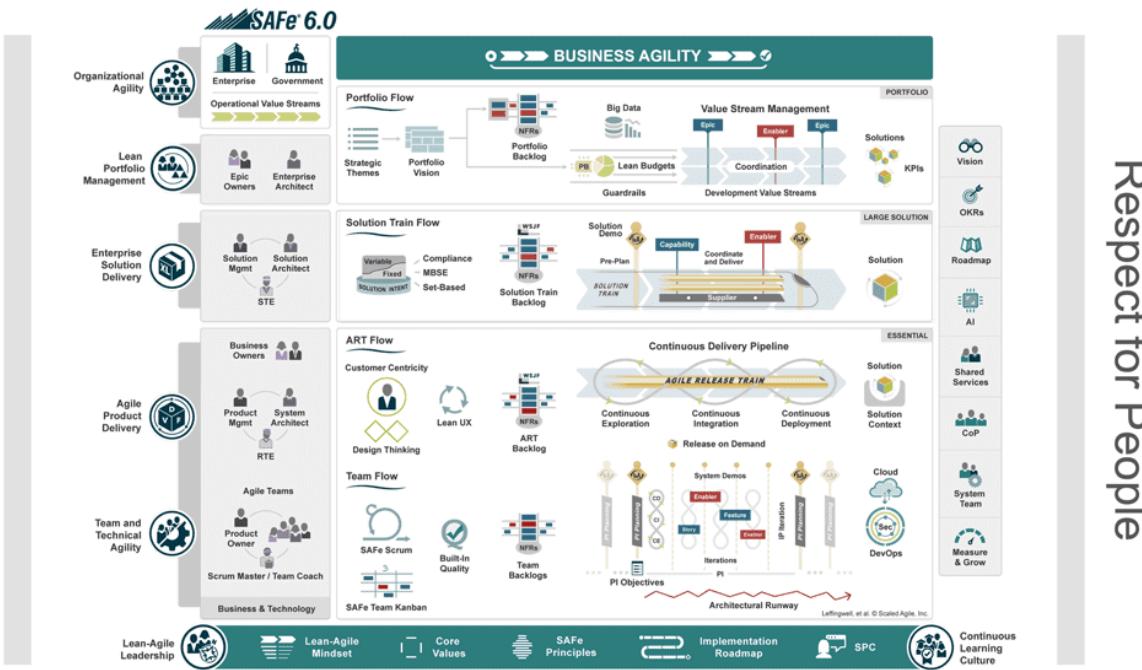
These tenets help guide the behaviors and actions of everyone participating in a SAFe portfolio. Those in positions of authority can help the rest of the organization embrace these ideals by exemplifying these values in their words and actions.

## Details

SAFe is a system for achieving business agility based on highly respected bodies of knowledge: Agile, Lean, systems thinking, DevOps, and value stream management, to name a few. Their relevance to business agility has been proven through the successful adoption of these practices by the world's largest organizations. These comprehensive success patterns make SAFe broad, deep, and scalable. But at its core, SAFe places the highest value on four deeply held beliefs: *alignment, transparency, respect for people, and relentless improvement*. These tenets are so fundamental to the practice of SAFe that without them, the practices in the Framework will inevitably fail to deliver the intended business results that prompted the decision to 'go SAFe.' These core values are illustrated in Figure 1 and described in the following sections.

# Transparency

Alignment



Respect for People

# Relentless Improvement

© Scaled Agile, Inc.

Figure 1. SAFe's four core values

## Alignment

Like cars out of alignment, misaligned companies can develop serious problems. They are hard to steer and don't respond well to changes in direction [2]. Even if it's clear where everyone thinks they're headed, the vehicle is unlikely to get them there.

The same is true for organizations adopting SAFe as their new way of working. In Lean-Agile, many decisions are decentralized to deliver value in the shortest sustainable lead time (see SAFe Principle #9). However, if decisions pull the organization in different directions, significant delays and quality concerns will result. The solution is to provide clear, consistent alignment from the top of the enterprise through every level of SAFe, all the way to each individual contributor. Value delivery with speed and quality can consistently be achieved when everyone is aligned.

Here are specific ways to create and maintain alignment in SAFe.

**Communicate the vision, mission, and strategy.** Alignment starts with keeping the enterprise's vision, mission, and strategy constantly present. For example, including these elements in the

Business Owner briefings during PI planning is one way to ensure the work of the ART is consistent with the higher aims of the enterprise.

**Connect strategy to execution.** The next step is to make sure everyone in the SAFe Portfolio aligns their work to the most important things to the enterprise. Strategic themes explicitly translate business strategies into tangible guidance that aligns the portfolio vision, lean budgets, and epics to enterprise priorities and subsequently inform the work of teams and trains in the portfolio.

**Speak with a common language.** It's difficult to achieve alignment if there's inconsistency in how the organization describes important roles, processes, events, and artifacts. SAFe provides a common language and promotes practices (backlogs, ART boards, solution intent, portfolio vision, and so on) that maintain a common view of the work and the resulting solutions.

**Constantly check for understanding.** Creating alignment requires regular reinforcement. SAFe events (iteration planning, backlog refinement, PI planning, ART syncs, portfolio syncs) and SAFe artifacts (backlogs, team boards, ART boards, portfolio canvas) are just some of the tools that help the SAFe organization stay aligned. Face-to-face conversations are also essential for checking for understanding.

**Understand your customer.** SAFe promotes continuous exploration with customer centricity and design thinking to gather inputs and perspectives from diverse stakeholders and information sources to ensure that the items in the backlogs are aligned with the most important voice of all... the customer.

## Transparency

Solution development is complex. Often, things go wrong or do not work out as planned. Without openness, facts are obscure, and decision-making is based on speculative assumptions and a lack of data. No one can fix a secret.

To ensure transparency—*trust* is needed. Trust exists when everyone can confidently rely on one another to act with integrity, particularly in times of difficulty. Without trust, it is impossible to build high-performing teams and trains or build (or rebuild) the confidence needed to make and meet reasonable commitments. Trust-based environments are also fun and motivating. Simply put, the new way of working promoted by SAFe will struggle to succeed without a culture of transparency and trust.

The following actions can help build a culture of transparency and trust in the SAFe enterprise.

**Create a trust-based environment.** Trust requires action, not just a feeling. People at every level of the organization must be willing to trust others and be trustworthy themselves. It means making and keeping commitments. It also means relinquishing control and trusting others to make and keep their commitments.

**Communicate directly, openly, and honestly.** A frequently used mantra in SAFe is ‘the facts are friendly.’ Problems cannot be solved if they are hidden. In a trust-based environment, information is shared without embellishment or blame to resolve issues as quickly and effectively as possible.

**Turn mistakes into learning moments.** People often learn more from their mistakes than from their successes, but this is effective only when those mistakes can be acknowledged without fear of retribution or punishment. Address mistakes as ‘learning moments’ [3] to create the psychological safety needed to quickly surface and resolve errors.

**Visualize work.** Making all work visible is essential to transparency. In SAFe, this begins with all work at every level being captured in a continuously refined backlog. Other tools such as Kanban boards, ART boards, PI objectives, solution intent, collaboration tools, and shared knowledge repositories support the aim of keeping work visible and accessible to all.

**Provide ready access to needed information.** Information that is difficult to find has the same impact as if that knowledge were intentionally hidden. True transparency requires that information is easily accessible to all who need it and that the location and means of access are well known. It requires a willingness to help each other find required information when the location is unclear and relentlessly improve systems to make accessing information as frictionless as possible.

## Respect for People

‘

*First we build people, then we build cars.*

—Widely attributed to Fujio Cho, former Chairman, Toyota

A Lean-Agile approach doesn’t implement itself or perform any real work. In reality, people do all the work, and people receive all the value from the work. Since people are the focal point of how enterprises create value with SAFe, respect for people must be considered in every aspect of the new way of working.

Respect is a basic human need. When treated with respect, people are unleashed to evolve their practices and contribute their creativity. Conversely, people cannot commit to another person, their teams, or their organizations if they feel a lack of respect. When disrespect is widespread and tolerated, it creates a toxic work environment, poor performance, and a high attrition rate [4].

The following suggestions provide just a few ways to cultivate a culture of respect for people in an organization.

**Hold precious what it is to be human.** This is the literal meaning of ‘respect for people’ described in Lean. Elaborated further, it means fostering a corporate culture that enhances

individual creativity and values teamwork while honoring mutual trust and respect [5].

**Value diversity of people and opinions.** Another way to show respect is to build organizations that include people with various personal and professional backgrounds. However, just hiring a diverse workforce is insufficient. Respect for people requires listening to and valuing perspectives and viewpoints different from our own.

**Grow people through coaching and mentoring.** Respect for people goes beyond a fundamental moral obligation. The practical business implication is that 'it is necessary to develop good people in order to make good products.' One way to help people grow is to facilitate connections with others inside and outside the organization who can contribute to each person's development journey.

**Embrace 'your customer is whoever consumes your work.'** Lean and Agile methods are customer-centric, as both recognize that customers are the ultimate beneficiaries of value. In addition, Lean explicitly addresses the fact that all who consume your work, including people inside the organization, are customers too. Treating customers with respect and empathy produces products and services that address their real problems.

**Build long-term partnerships based on mutual benefit.** Many suppliers are required when building the world's most complex systems. When applied to suppliers, respect means holding them in the same high regard as customers, challenging them, and helping them improve. This is done not by bullying or pressuring tactics but by creating long-term relationships defined by 'win-win' contracts based on mutual benefit and accountability.

## Relentless Improvement

The relentless pursuit of perfection has always been one of the core tenets of Lean. While unattainable, striving for perfection leads to continuous improvements to products and services. In the process, companies have created more and better products for less money and with happier customers, leading to higher revenues and greater profitability.

But improvement requires learning. Rarely are the causes and solutions for problems that organizations face clear and easily identified. Relentless improvement is built on a series of small iterative and incremental improvements and experiments that enable the organization to learn its way to the most promising answer to a problem.

The following moves can help build a culture of relentless improvement in a SAFe enterprise:

**Create a constant sense of urgency.** Improvement activities are essential to the survival of an organization and should be given priority, visibility, and resources. They require an intense focus on delivering value to customers by providing products and services that solve their problems in a preferred way over the organization's competitors. Organizations that become complacent and fail to relentlessly improve with urgency risk losing customers and may ultimately go out of business.

**Build a problem-solving culture.** Problem-solving is the driver for relentless improvement. It recognizes that a gap exists between the current and desired states, requiring an iterative process to achieve the target state. Iterative Plan-Do-Check-Adjust (PDCA) cycles provide the process for iterative problem-solving on small adjustments as well as breakthrough innovations. The goal is to have a culture of 'everyone improving all the time.'

**Reflect and adapt frequently.** It's vital in SAFe to periodically pause from the never-ending backlog of new work to openly identify and address shortcomings of the process at all levels. Improvements should be managed and prioritized just like any other story or feature because improvement requires real work and consumes real capacity of teams and trains, as well as for those who guide the portfolio processes.

**Let facts guide improvements.** Improvements based on opinion or conjecture will likely focus on symptoms instead of true root causes. Improvement results are objectively measured, focusing on empirical evidence. This helps an organization concentrate more on the work needed to solve problems and less on assigning blame or on pursuing improvements that are not solving the original problem.

**Provide time and space for innovation.** Improvements should be designed to increase the effectiveness of the entire system that produces the sustainable flow of value instead of optimizing individual teams, silos, or subsystems. Everyone at all levels should embrace improvement thinking, but improvements in one area, team, or domain should not be made to the detriment of the overall system. SAFe Principle #2 [Apply systems thinking](#) expands on this idea further.

## Leadership is Required

Consistently applying the SAFe core values requires the active support of [Lean-Agile Leadership](#) and a [Continuous Learning Culture](#). Leaders in a SAFe organization should exemplify the core values combined with the Lean-Agile Mindset, SAFe [Principles](#) and practices, and an orientation toward creating value for customers. In turn, modeling these mindsets and behaviors creates a persistent and meaningful values-based culture for teams and their stakeholders.

---

## Learn More

[1] Ratzenberger, John, and Joel Engel. *We've Got it Made in America: A Common Man's Salute to an Uncommon Country*. Center Street, 2006.

[2] Labovitz, George H., and Victor Rosansky. *The Power of Alignment: How Great Companies Stay Centered and Accomplish Extraordinary Things*. Wiley, 1997.

[3] Ridge, Garry. The Learning Moment. Retrieved October 10, 2023, from <https://thelearningmoment.net/welcome-to-learning-moment/>

[4] Hu-Chan, Maya. *The Basic Human Need Some Companies Fail to Meet*. Inc.com, October 30, 2020. Retrieved October 10, 2023, from <https://www.inc.com/maya-hu-chan/the-basic-human-need-some-companies-fail-to-meet.html>

[5] Miller, Jon. *Exploring the “Respect for People” Principle of the Toyota Way*. Gemba Academy, May 16, 2017. Retrieved October 10, 2023, from [https://blog.gembaacademy.com/2008/02/03/exploring\\_the\\_respect\\_for\\_people\\_principle\\_of\\_the/](https://blog.gembaacademy.com/2008/02/03/exploring_the_respect_for_people_principle_of_the/)  
Liker, Jeffery K. *Developing Lean Leaders at All Levels: A Practical Guide*. Lean Leadership Institute Publications, 2014.

Last update: 10 October 2023

The information on this page is © 2010-2024 Scaled Agile, Inc. and is protected by US and International copyright laws. Neither images nor text can be copied from this site without the express written permission of the copyright holder. Scaled Agile Framework and SAFe are registered trademarks of Scaled Agile, Inc. Please visit [Permissions FAQs](#) and [contact us](#) for permissions.

## Framework

[Download SAFe Posters & Graphics](#)

[Blog](#)

## Training

[Course Calendar](#)

[About Certification](#)

[Become a Trainer](#)

## Content & Trademarks

[FAQs on how to use SAFe content and trademarks](#)

[Permissions Form](#)

[Usage and Permissions](#)

# Scaled Agile, Inc

## Contact Us

5400 Airport Blvd., Suite 300  
Boulder, CO 80301 USA

## Business Hours

Weekdays: 9am to 5pm  
Weekends: CLOSED

---

© 2024 Scaled Agile, Inc.

[Get News](#)

[Terms of Use](#)

[Privacy Policy](#)

[Code of Conduct](#)

[Remote Training Policy](#)

