

# NEW-AGE GLOBAL UNIVERSITY FOR LIBERAL EDUCATION



# Agile Software Engineering

## CS 2004

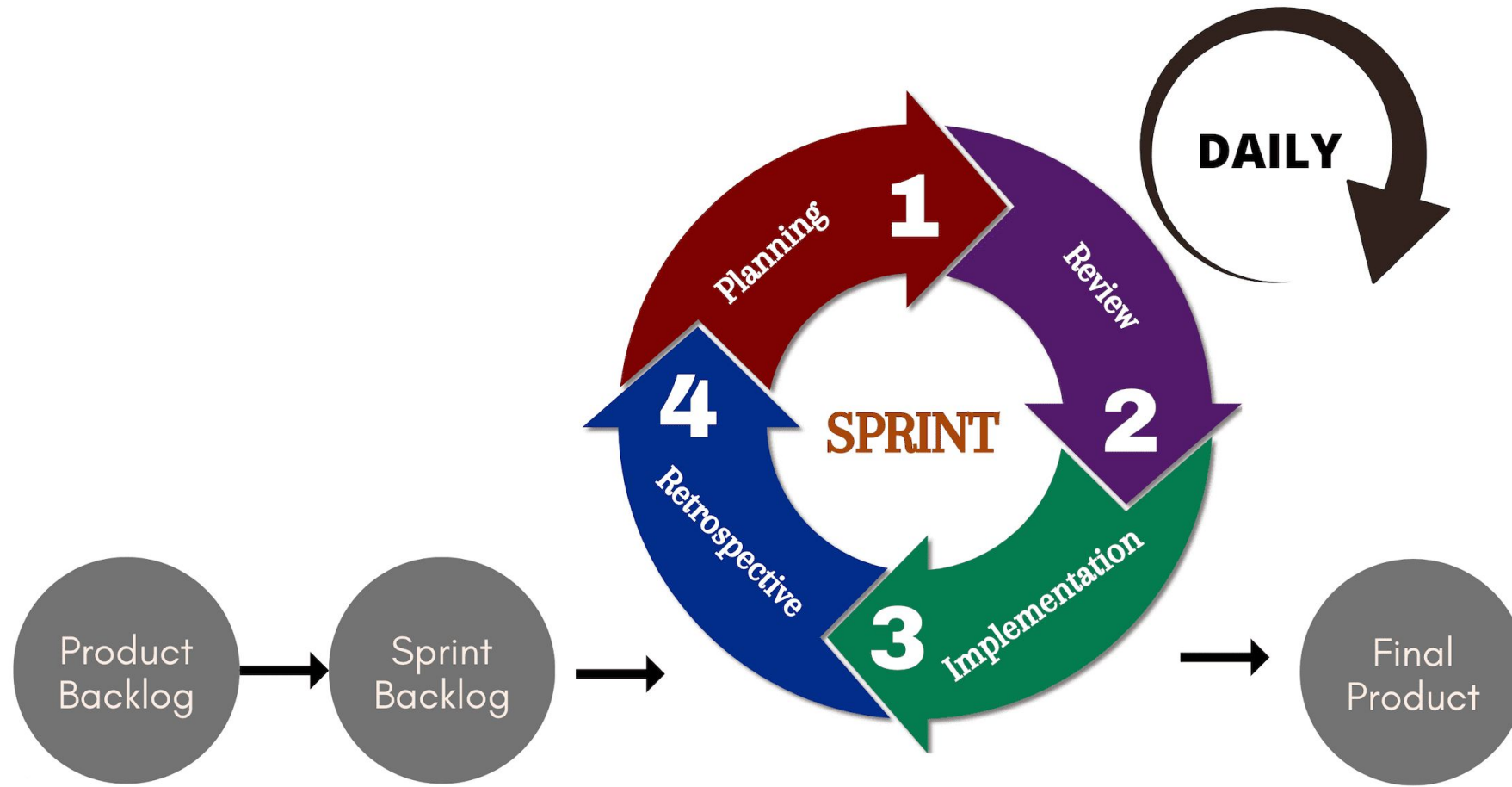
### Unit 3

#### Sprint Management & Planning Techniques

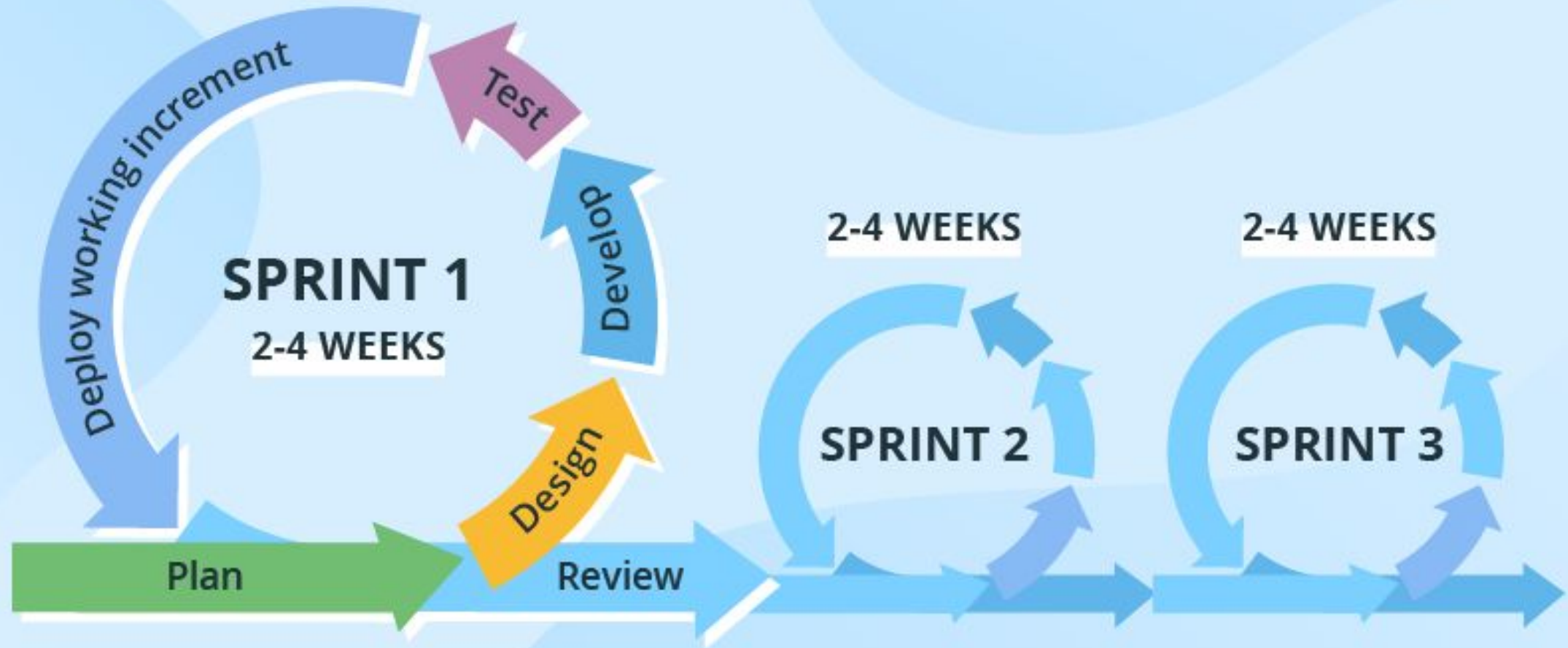
# Agenda

- Sprint Planning, Sprint Reviews, Sprint Retrospectives Sprint Planning - Agile release and iteration (sprint) planning,
- Develop Epics and Stories, Estimating Stories, Prioritizing Stories (WSJF technique from SAFe),
- Iterations/Sprints Overview. Velocity Determination, Iteration Planning Meeting, Iteration
- Planning Guidelines, Development, Testing, Daily Stand-up Meetings, Progress Tracking, Velocity Tracking, Monitoring and Controlling: Burn down Charts, Inspect & Adapt (Fishbone Model), Agile Release Train

# Scrum & Sprint – Recap.

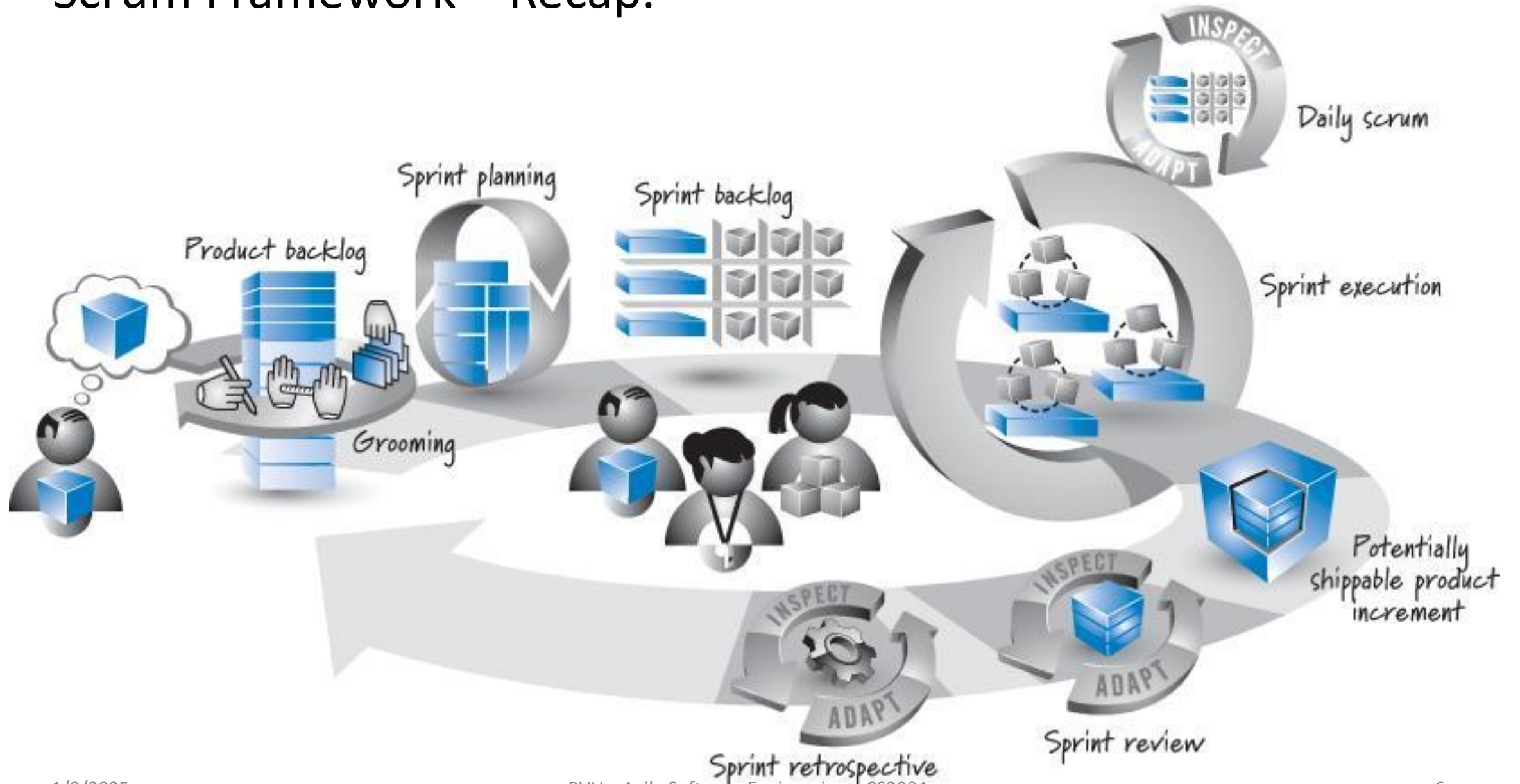


# SCRUM

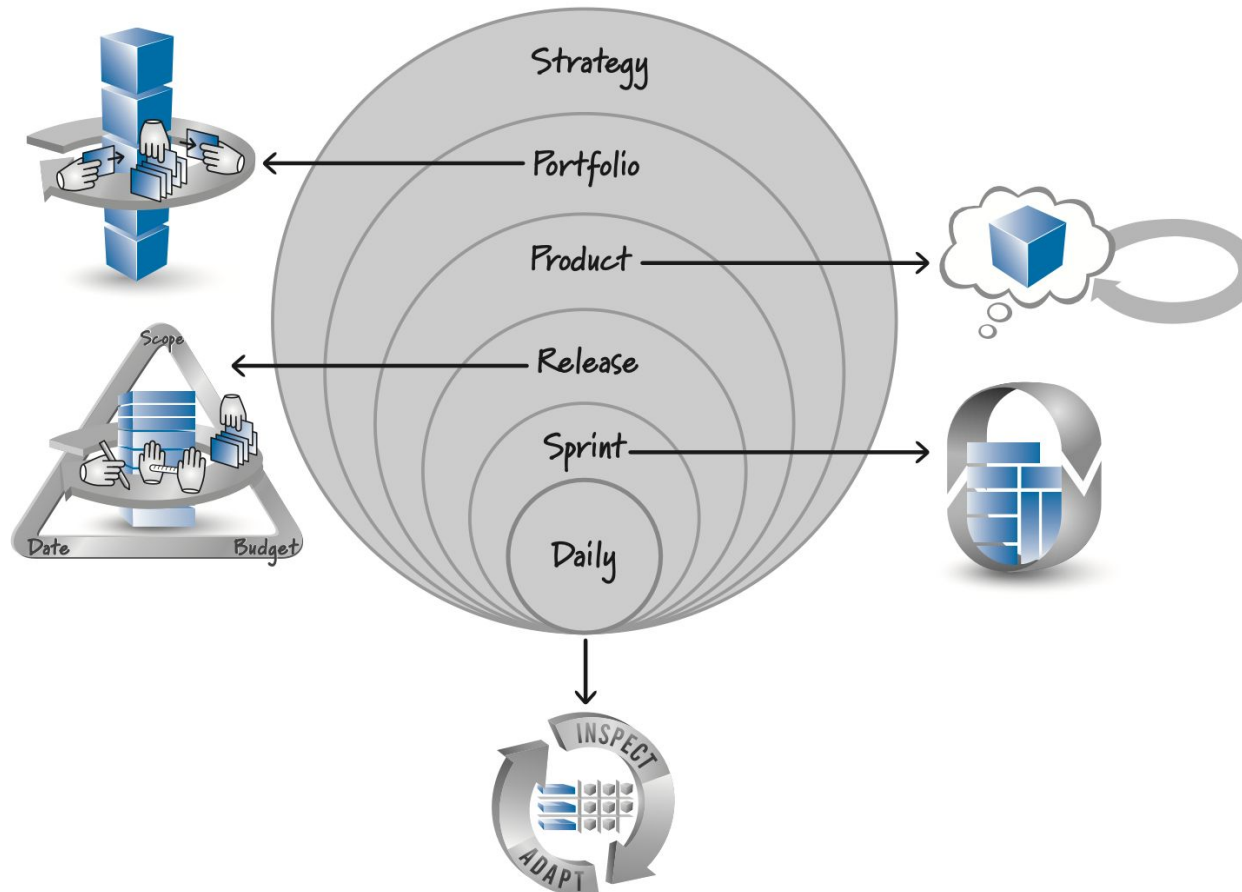




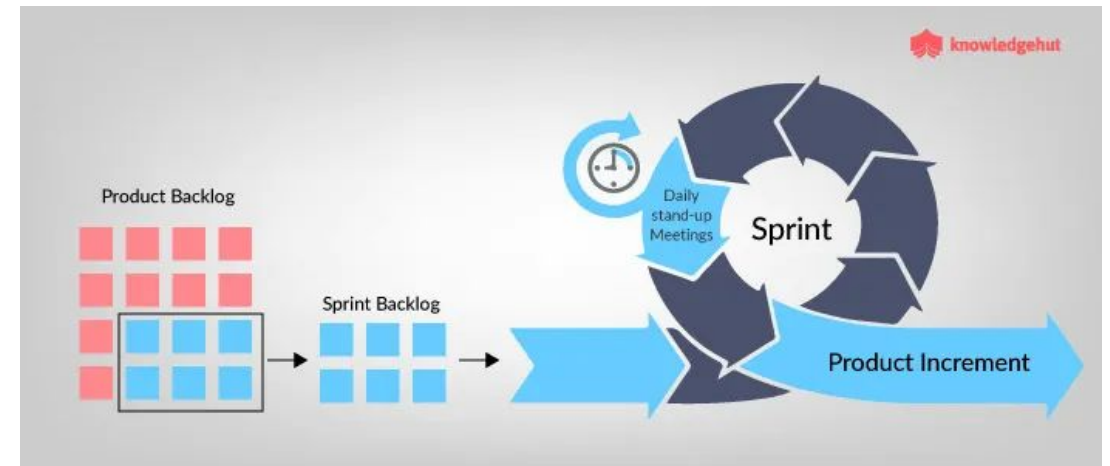
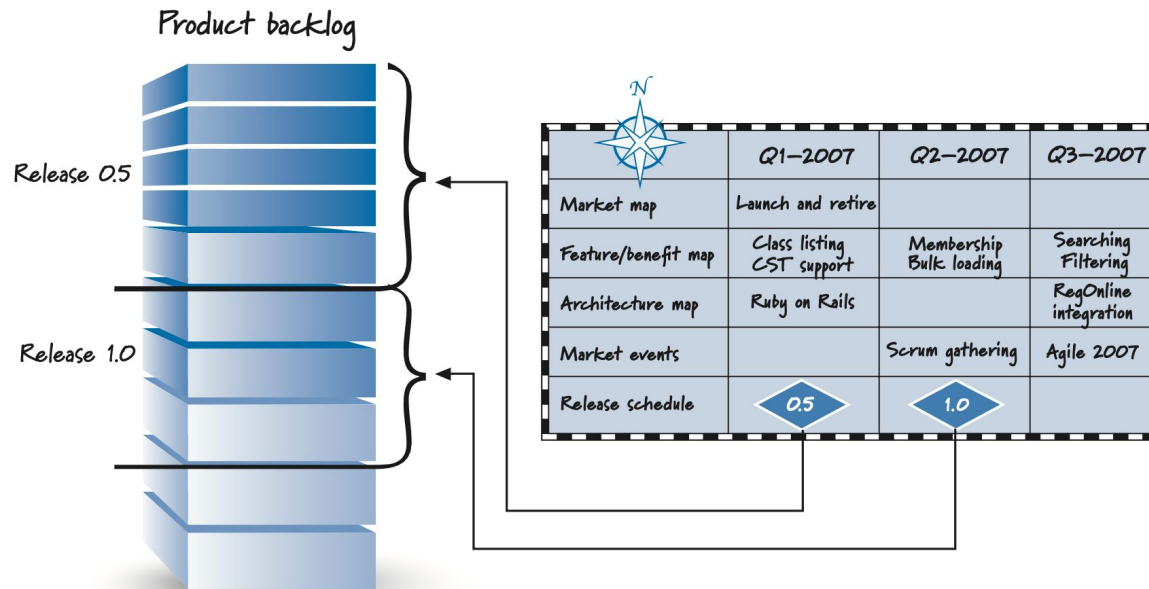
# Scrum Framework – Recap.



# Different Levels of Planning

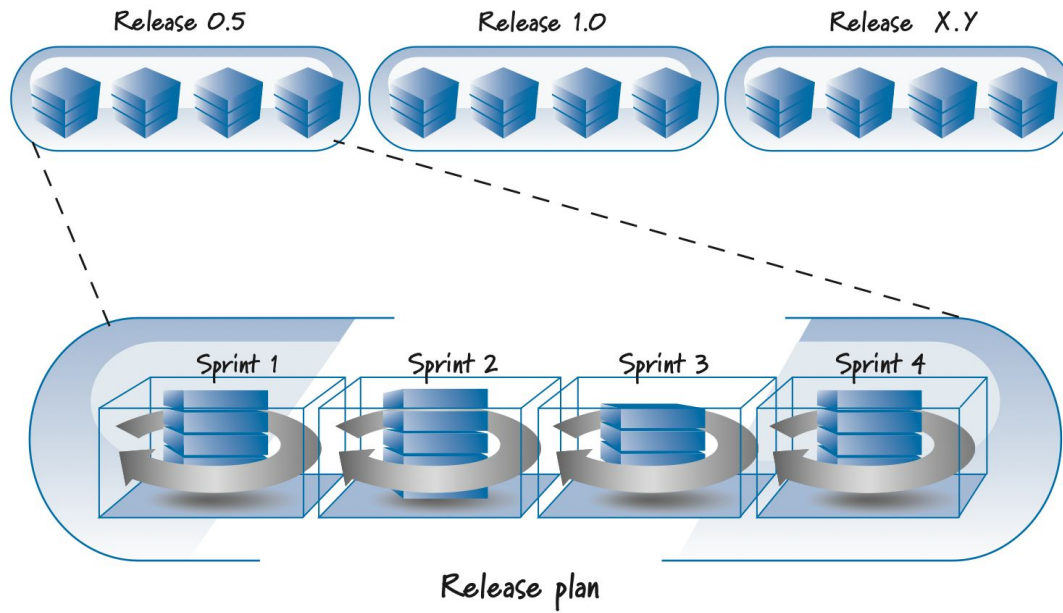


# Agile Release Planning





# Release Plan



## Sprint Calendar

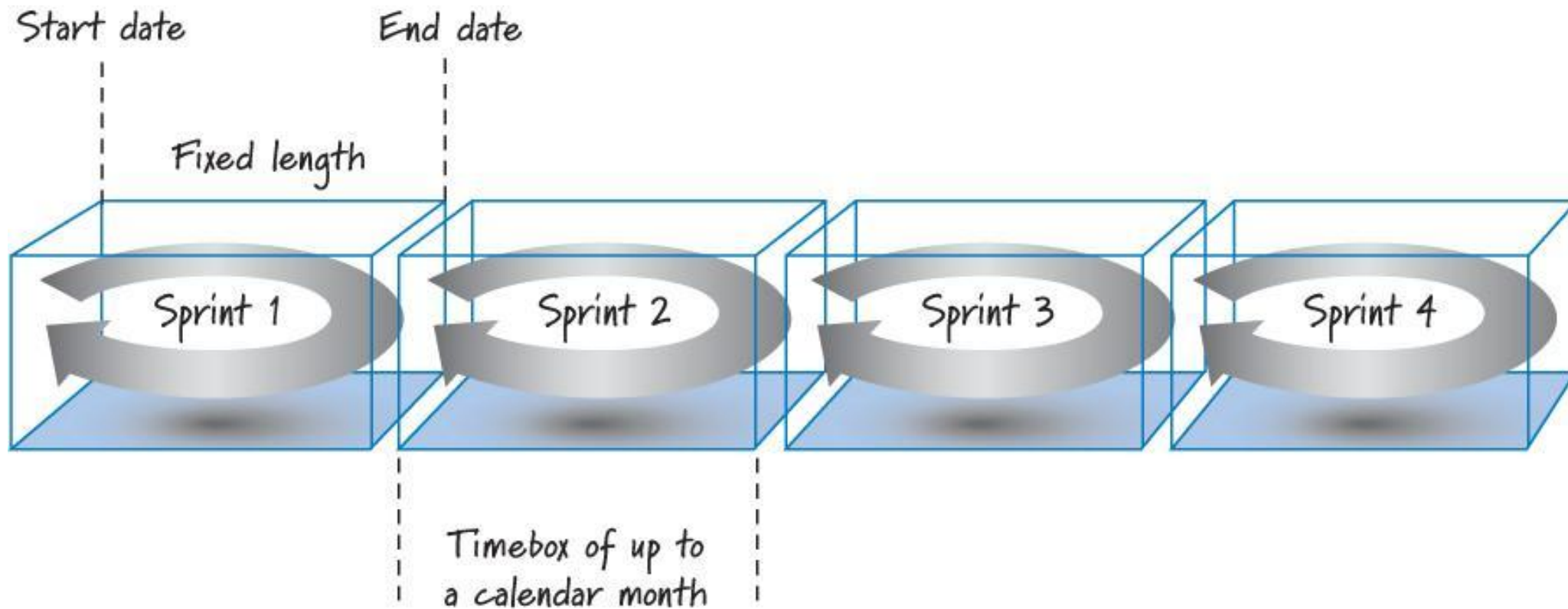
		July								
		S	M	T	W	TH	F	S		
							1	2		
Sprint 1		3	4	5	6	7	8	9		
		10	11	12	13	14	15	16		
		17	18	19	20	21	22	23		
Sprint 2		24	25	26	27	28	29	30		

		August								
		S	M	T	W	TH	F	S		
		31	1	2	3	4	5	6		
Sprint 3		7	8	9	10	11	12	13		
Sprint 4		14	15	16	17	18	19	20		
		21	22	23	24	25	26	27		
		28	29	30	31					

# Sprint

- "sprint" is a time-boxed iteration or development cycle during which a defined set of work is completed
- Sprints are a fundamental concept in Scrum and are used to structure the development process into manageable and predictable periods

# Sprint Characteristics

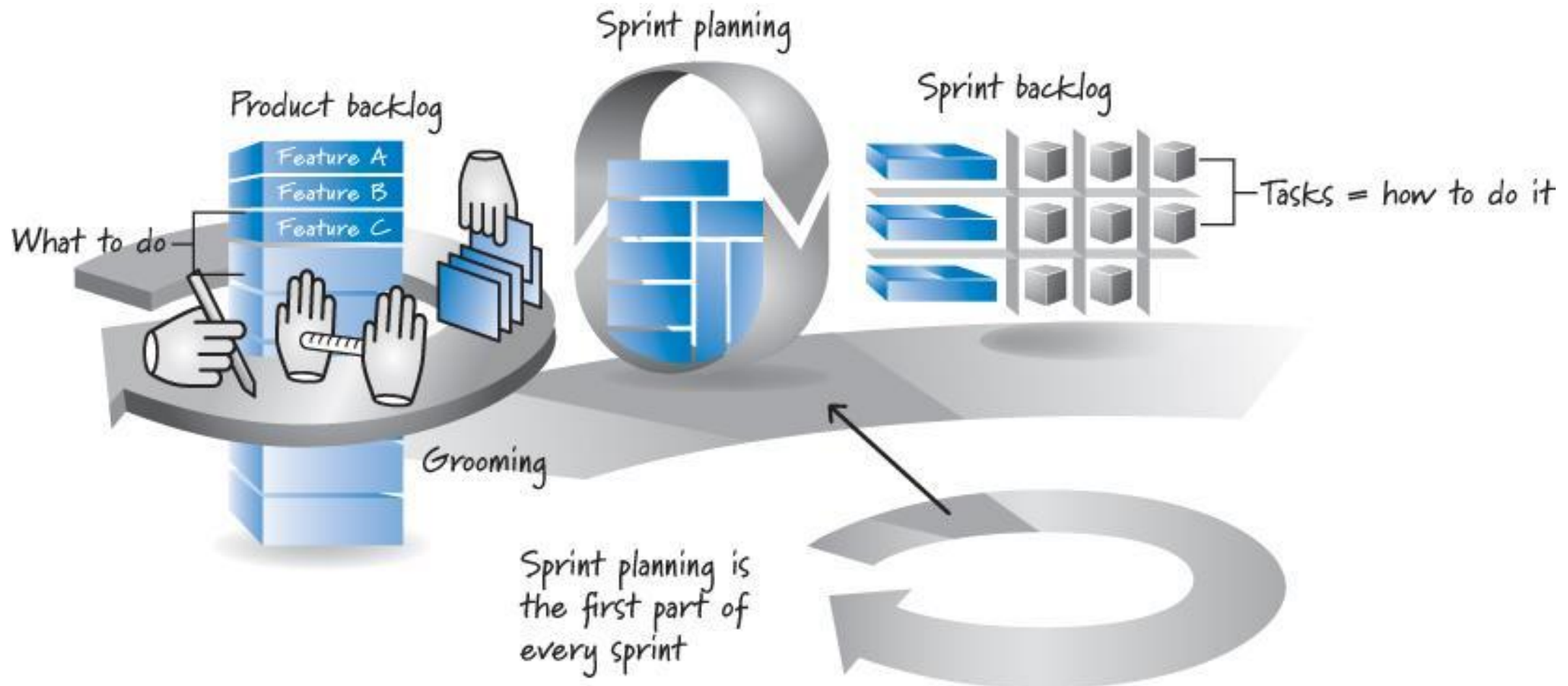




# Iterations

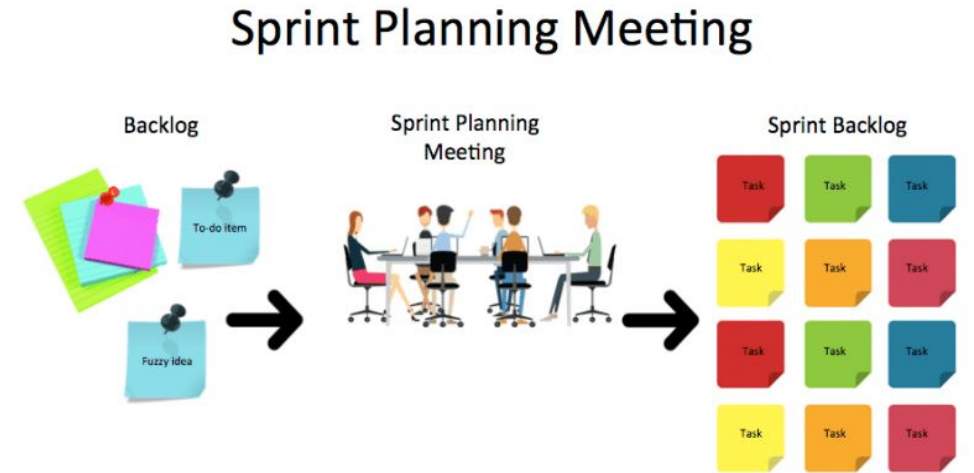
- **Iterations** (Sprints) are the heartbeat of Scrum and create a rhythm for work within the larger PI timebox
- All the work necessary to achieve the iteration—including planning, team sync, iteration review, and retrospective—occur within iterations
- Each iteration is a standard, fixed-length timebox, typically two weeks in length
- The team delivers high-quality increments of value during the iteration as working, tested software, solutions, or other valuable artifacts. Iterations are continuous and sequential, and a new iteration starts immediately after the previous one

# Sprint Planning

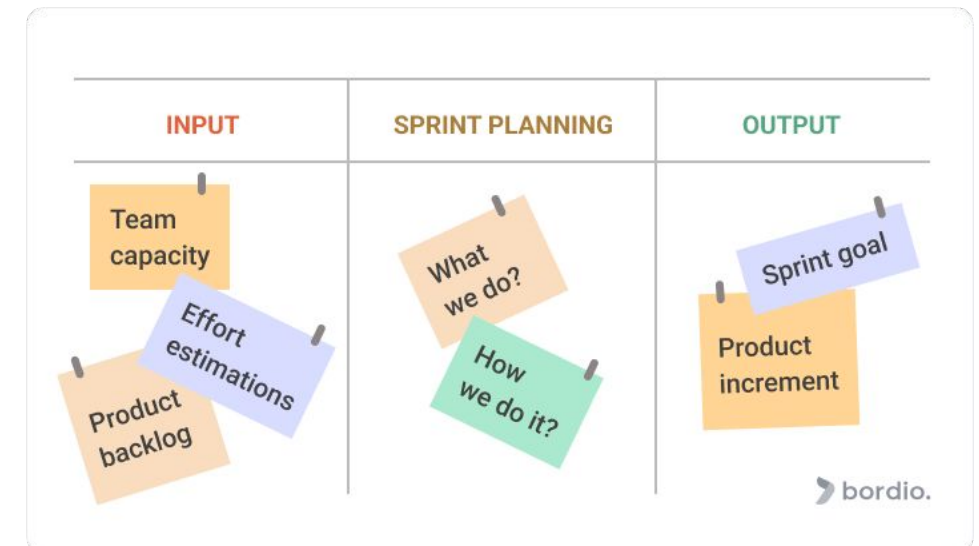


# Sprint Planning

- A key event in the Agile Scrum framework designed to set the direction and goals for an upcoming sprint.
- It is a collaborative meeting that occurs at the beginning of each sprint and involves the Scrum Team, which includes the Scrum Master, the Product Owner, and the development team.
- The primary objective of Sprint Planning is to determine what work will be performed during the sprint and how that work will be accomplished.



Tech Agilist





# Sprint Planning - Meeting structure



- **Timebox:** Sprint Planning is typically time-boxed to a maximum of 8 hours for a one-month sprint. For shorter sprints, the meeting duration is usually proportionately shorter.
- **Participants:** The entire Scrum Team participates, although the Product Owner plays a critical role in providing clarity on the backlog items.

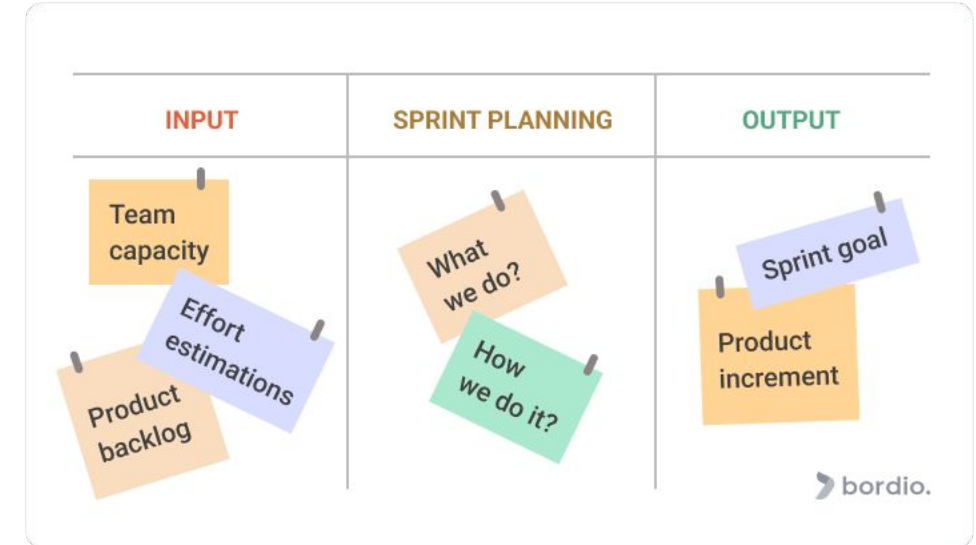
# Sprint Planning - Agenda

Sprint Planning usually involves two main parts:

- **What:** The team discusses and selects which Product Backlog items (PBIs) they will commit to completing during the sprint. This is based on the priority set by the Product Owner and the team's capacity.
- **How:** Once the items are selected, the team discusses how they will accomplish the work. This includes breaking down selected backlog items into smaller tasks, estimating the effort required, identifying dependencies, and ensuring team members agree on the approach.

# Sprint Planning - Inputs

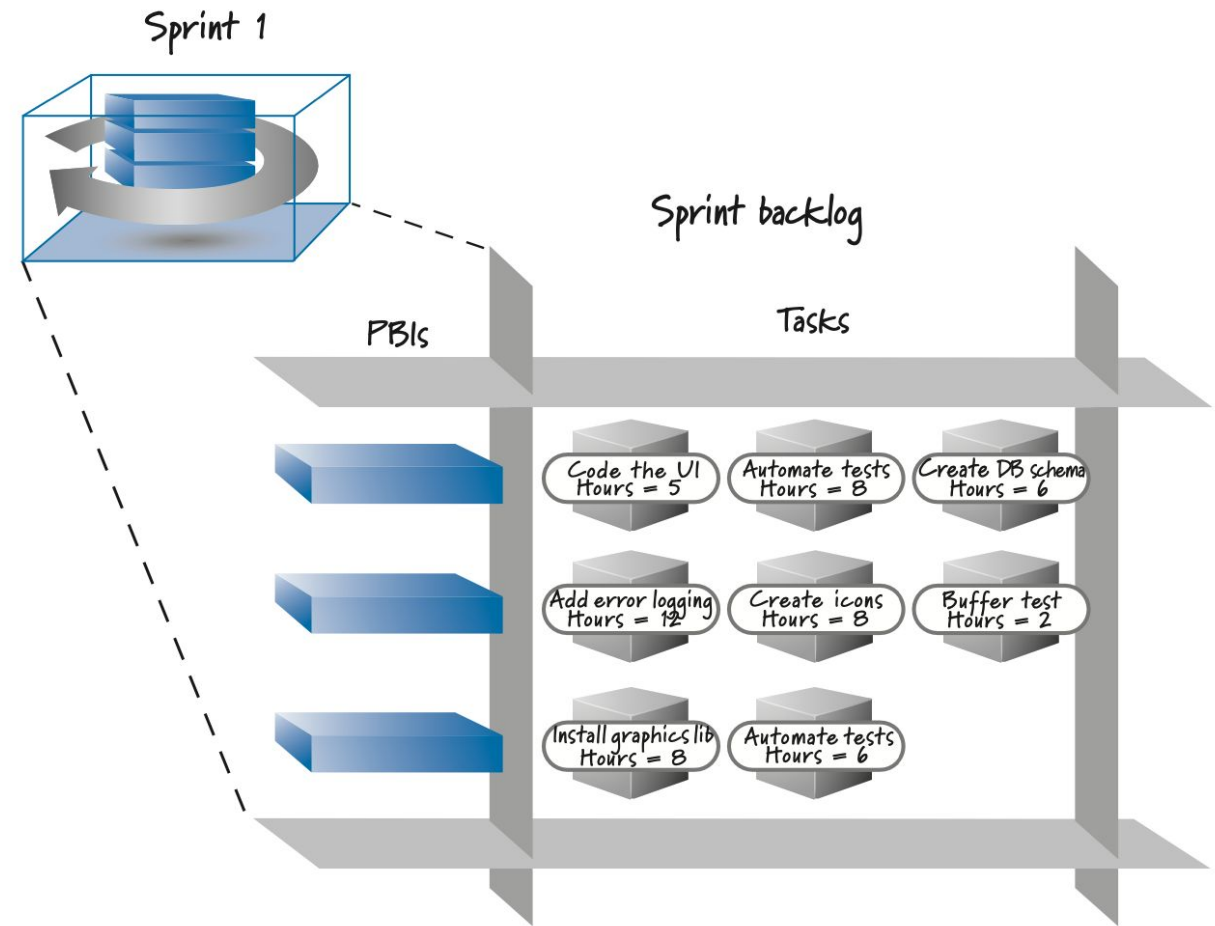
- **Product Backlog:** The ordered list of work items (stories, tasks, bugs, etc.) maintained by the Product Owner, which serves as the primary input for the planning.
- **Sprint Goal:** The team formulates a clear sprint goal that summarises the focus of the sprint and aligns with the selected work items
- Team Capacity & Effort Estimations





# Sprint Planning - Outputs

- **The Sprint Backlog:**
  - A list of accepted PBIs for the sprint, along with a plan for delivering the increment of product functionality.
  - The Sprint Backlog is a living artifact throughout the sprint
- **Sprint Goal:** The defined goal that provides a shared objective for the team and helps maintain focus throughout the sprint.

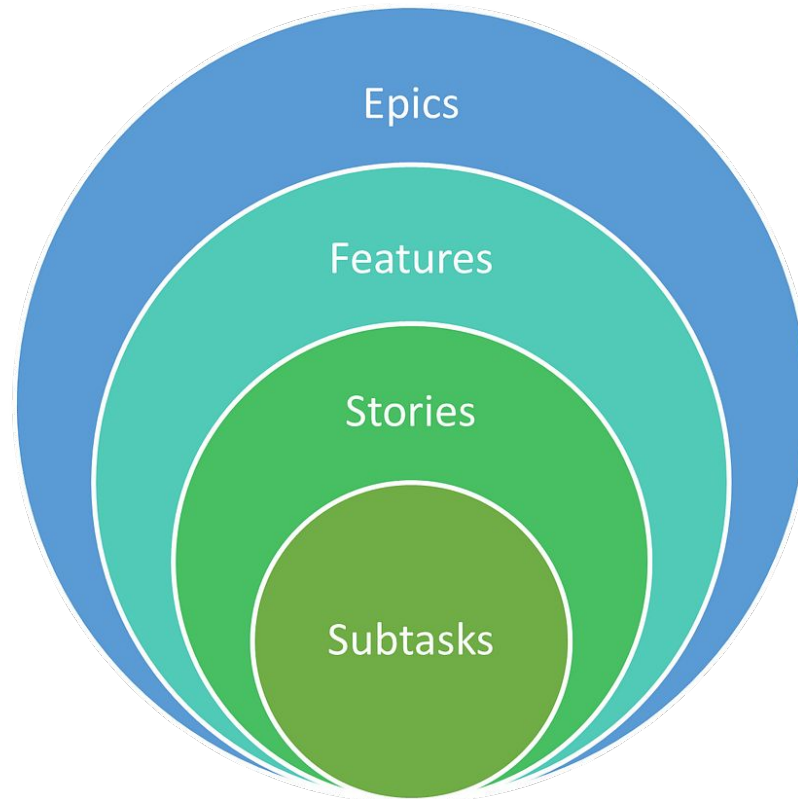


# Epics



- **Definition:** Epics are large bodies of work that can be broken down into smaller tasks or stories. They are broad, high-level features or objectives that span multiple sprints.
- **Purpose:** Provide a top-level overview of significant functionalities or goals, ensuring they align with business objectives and stakeholder needs.
- **Lifecycle:** Epics are typically broken down into smaller, more manageable pieces over time as more details are known and priorities are clarified.

# Epics, Features, Stories & Tasks





# Example Epic : I-phone UI

- **Epic: iPhone UI**
  - **Features** (High-level components of the Epic):
    - 1.Home Screen Design
    - 2.Notification System
    - 3.Settings Interface
    - 4.App Integration and Interaction

# Feature 1

## Feature 1: Home Screen Design

### *Story 1: Create Customizable Widget System*

- Subtask 1:** Research user preferences for widget customization.
- Subtask 2:** Design wireframes for widget layouts.
- Subtask 3:** Develop widget customization options.
- Subtask 4:** Test widget customization on various screen sizes.
- Subtask 5:** Document widget customization features.

### *Story 2: Implement New Icon Design Language*

- Subtask 1:** Create design mockups for new icon styles.
- Subtask 2:** Develop guidelines for new icon usage.
- Subtask 3:** Update existing app icons to match new design language.
- Subtask 4:** Conduct user testing for feedback on new icons.
- Subtask 5:** Launch updated icons in the iOS update.

# Feature 2

## Feature 2: Notification System

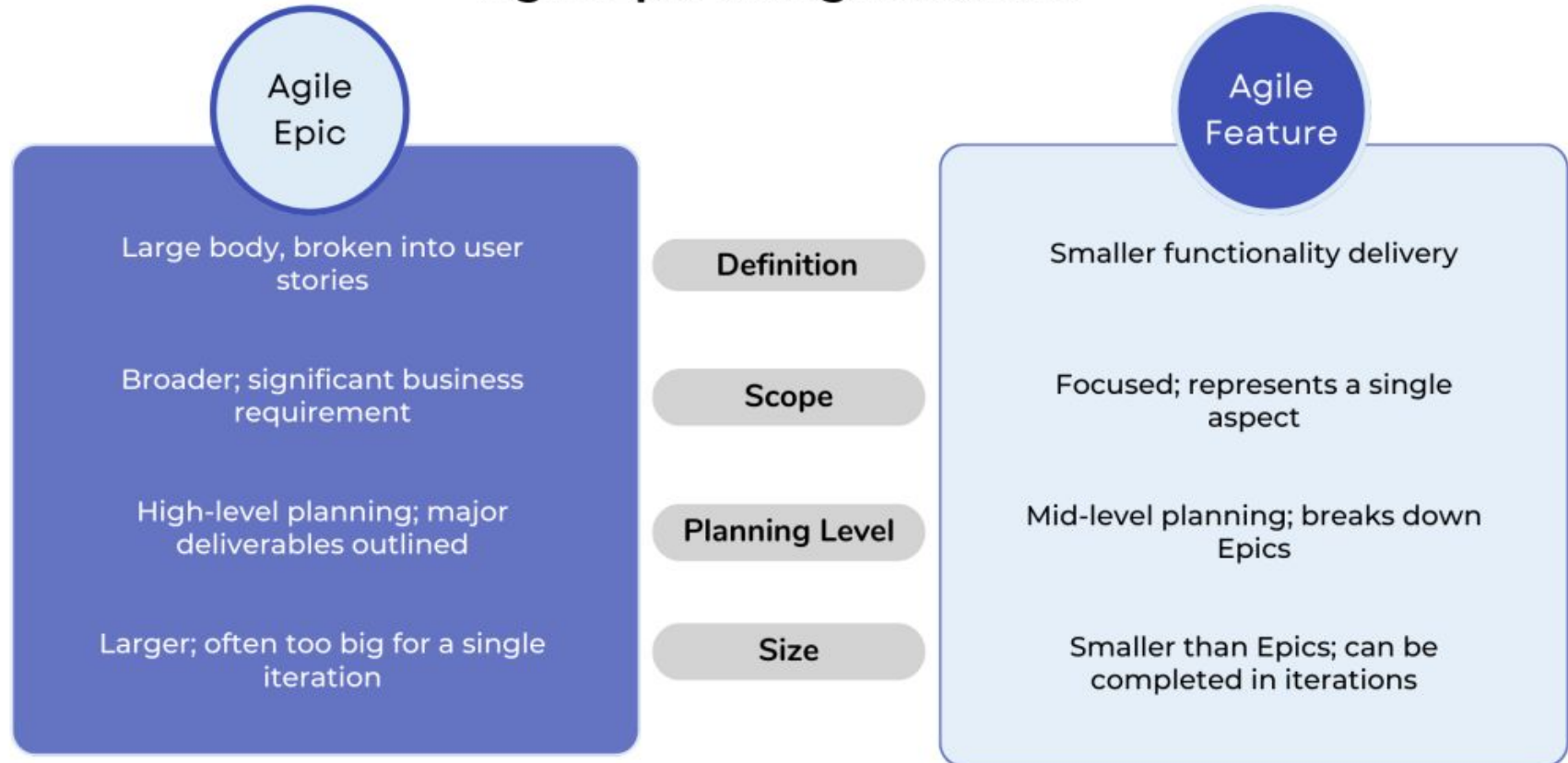
### *Story 1: Redesign Notification Center Layout*

- **Subtask 1:** Analyze current notification center user feedback.
- **Subtask 2:** Create new layout designs for the notification center.
- **Subtask 3:** Develop the redesigned notification center interface.
- **Subtask 4:** Test the new layout with real user data.
- **Subtask 5:** Implement improvements based on user testing results.

### *Story 2: Enhance Notification Interactivity*

- **Subtask 1:** Identify key interactive elements for notifications.
- **Subtask 2:** Design interactive notification prototypes.
- **Subtask 3:** Implement interactive features in notifications.
- **Subtask 4:** Perform user testing on interactive notifications.
- **Subtask 5:** Finalize and deploy interactive notification features.

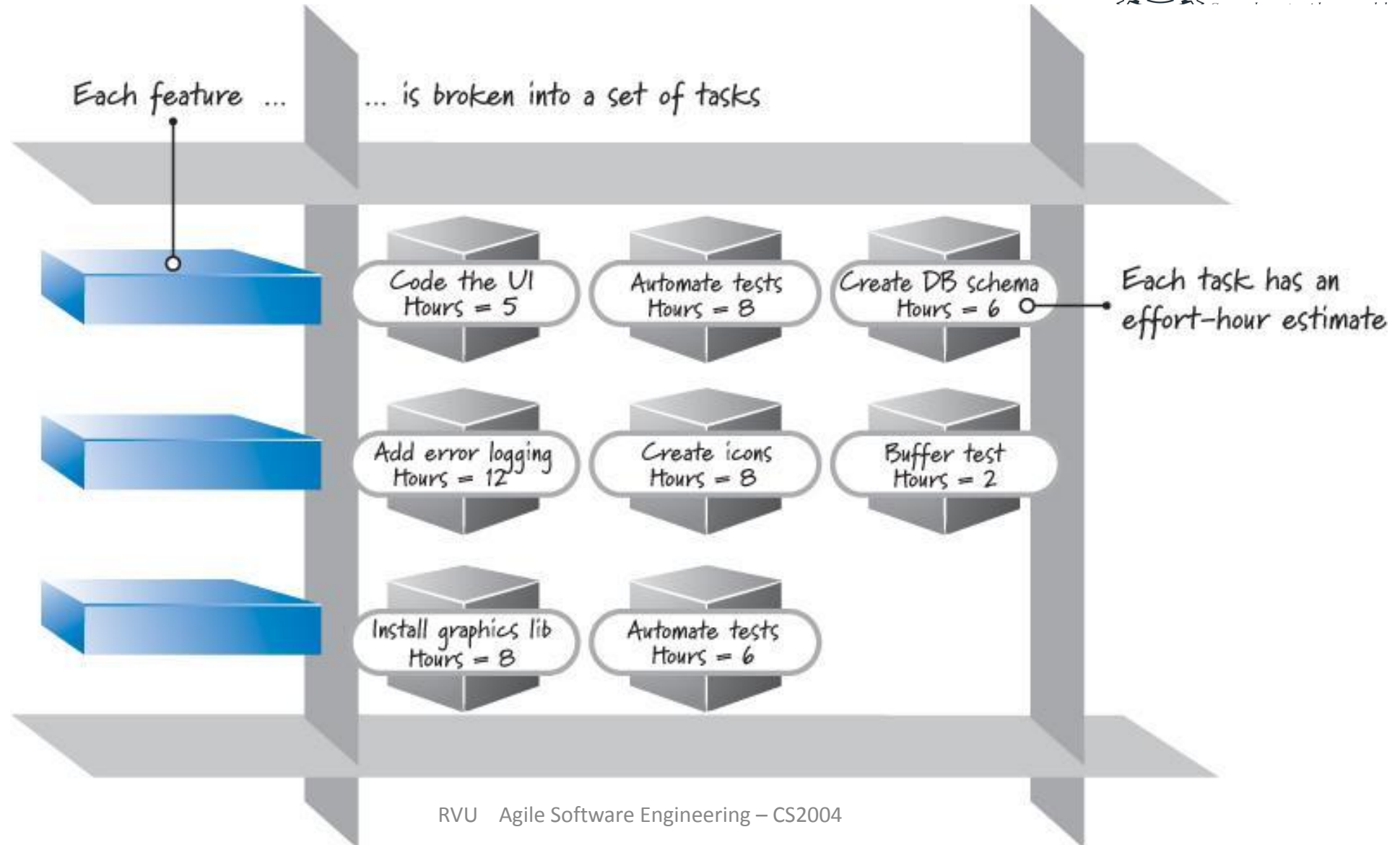
## Agile Epic Vs. Agile Feature



[www.invensislearning.com](http://www.invensislearning.com)

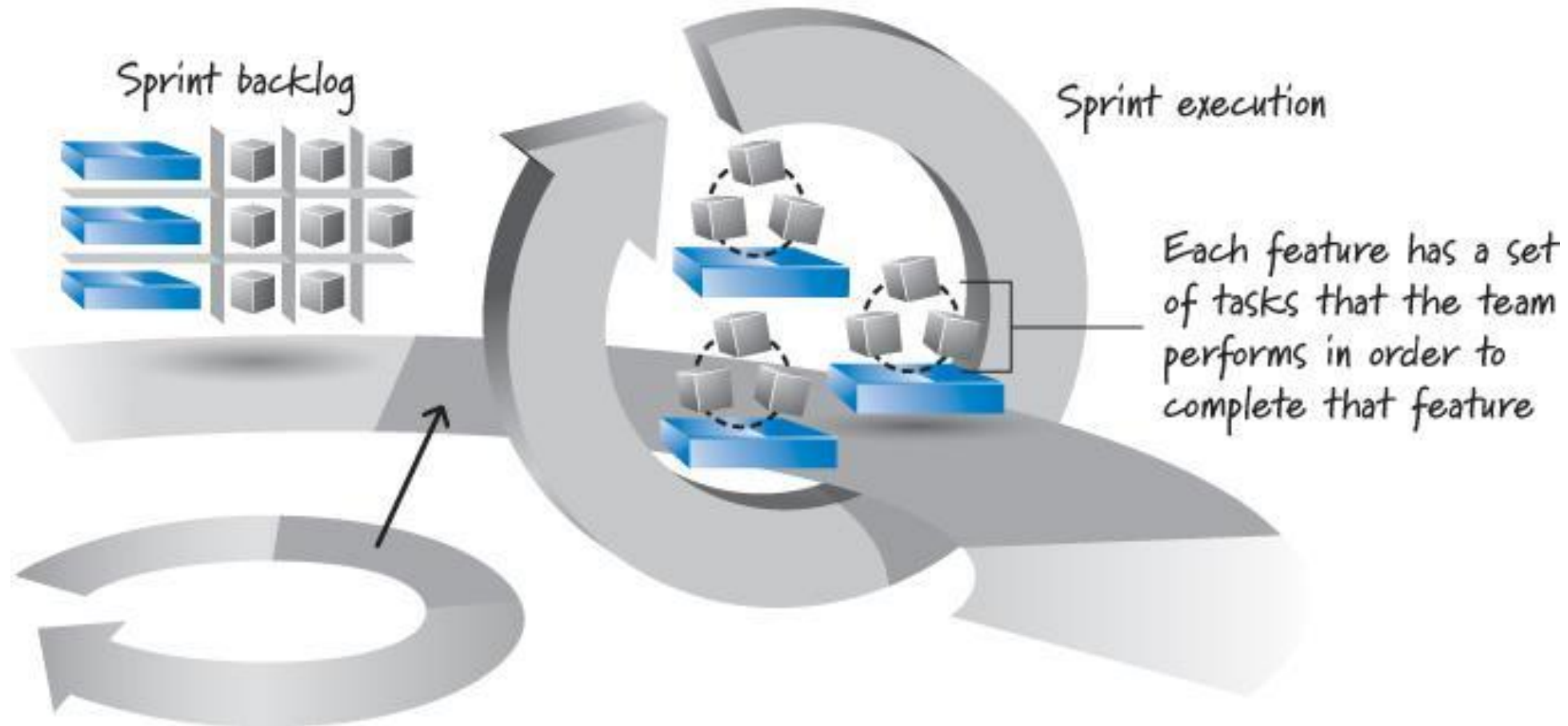


# Sprint Backlog

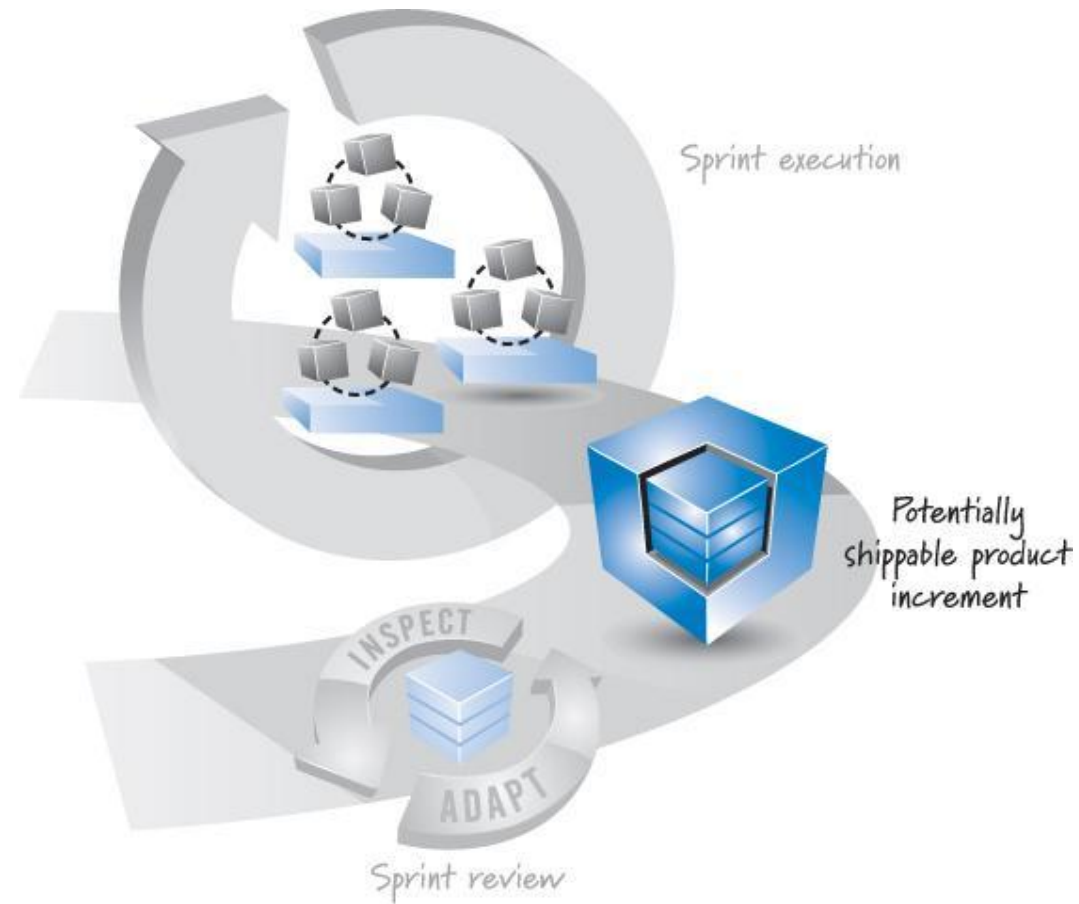


# Sprint Execution

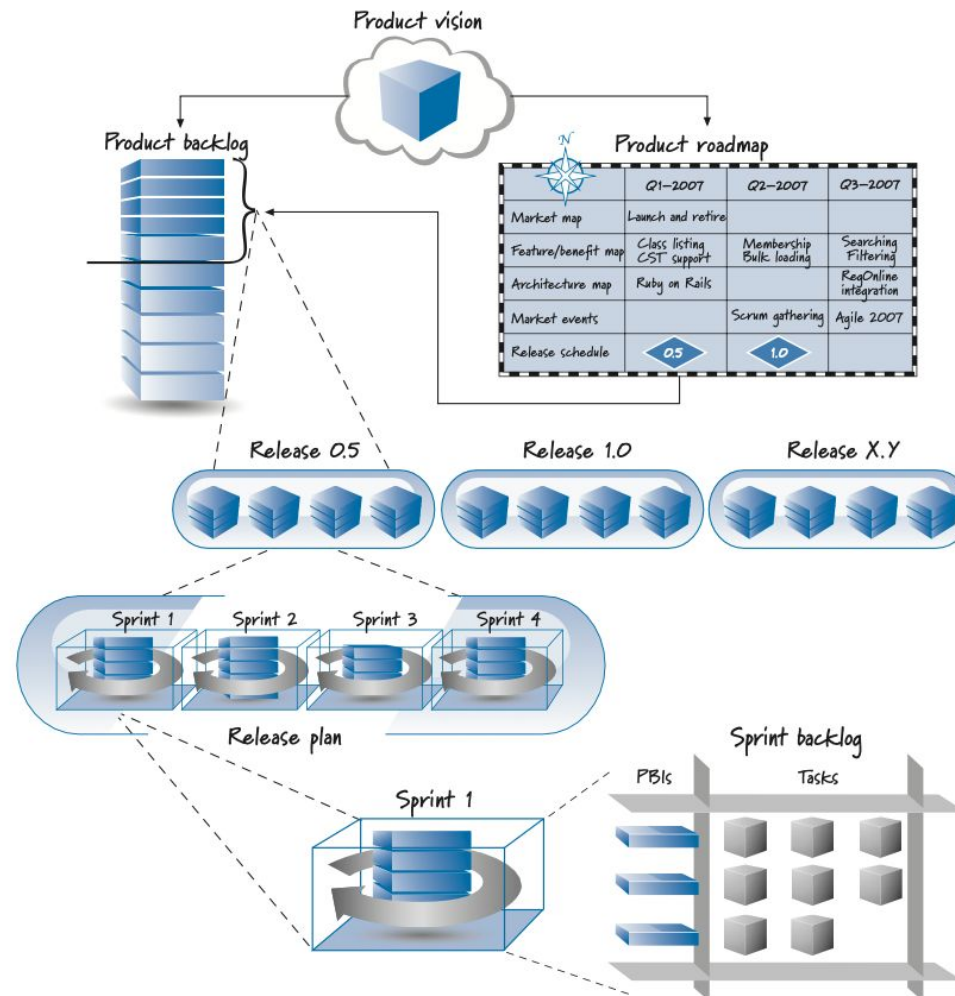
Sprint execution takes up the majority of time spent in each sprint



# Done Sprint Results



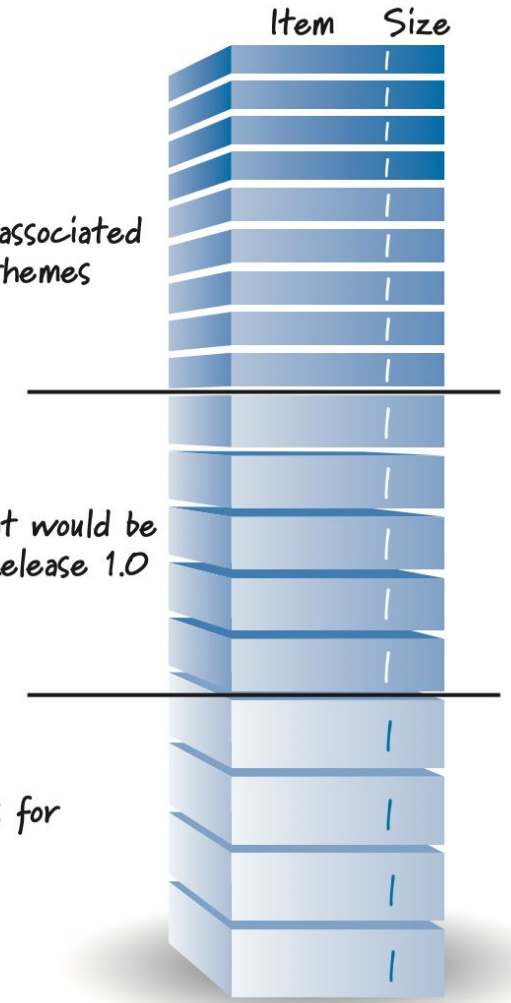
# Release - planning : Recap



Required stories associated with must-have themes

Other stories that would be nice to have in Release 1.0

Themes and epics for future releases



# Sprint Reviews

- A Sprint Review is an essential ceremony that occurs at the end of a Sprint, which is a time-boxed period (usually two to four weeks) during which a development team works to deliver a potentially releasable product increment. It serves the following purposes:
  - **Inspect Increment:** The primary goal is to review the work completed during the Sprint. The team demonstrates the increment of the product that has been developed, highlighting what has been accomplished.
  - **Gather Feedback:** Stakeholders, including customers, product owners, and team members, provide feedback on the presented work. This dialogue informs future development and can influence the product backlog.
  - **Adapt the Backlog:** Based on feedback and insights gained during the review, the Product Owner can adjust the product backlog as necessary. This ensures that the development effort remains aligned with the stakeholders' needs and priorities.



# Sprint Retrospectives

- Sprint retrospectives are a key component of the Agile/Scrum framework, serving as a dedicated meeting held at the end of each sprint (a time-boxed iteration typically lasting 1 to 4 weeks).
- The primary purpose of the sprint retrospective is to provide an opportunity for the Scrum team (which includes the Scrum Master, Product Owner, and Development Team) to reflect on the sprint that has just concluded. Here are the main objectives and components of a sprint retrospective:

# Sprint Retrospective - objectives



- **Reflection:** Team members reflect on what happened during the sprint, discussing both successes and challenges.
- **Improvement:** Identify and discuss areas for improvement in both processes and teamwork. This may include examining how the team collaborates, communicates, and completes the work.
- **Actionable Insights:** Generate actionable items or experiments the team can implement in the next sprint to enhance productivity and quality.
- **Fostering Openness:** Encourage open and honest communication among team members to build trust and foster a culture of continuous improvement.

# Sprint Retrospective - components



- **Set the stage:** The facilitator (usually the Scrum Master) creates a safe environment for discussion, encouraging everyone to share their thoughts and ensuring that the focus remains constructive.
- **Gather Data:** Collect insights from the team about what went well and what didn't. This can be done through various techniques such as silent brainstorming, using sticky notes, or surveys.
- **Generate Insights:** Analyze the data collected to identify patterns, root causes of issues, and areas of improvement. This might involve discussing specific incidents or broader themes based on the collected feedback
- **Decide What to do:** Prioritize the insights and agree on a set of actionable items the team would like to focus on in the next sprint. These action items should be realistic and clearly defined.
- **Close the retrospective:** Reflect on the retrospective itself, discussing what worked well in the meeting and what could be improved for future retrospectives. This ensures that the retrospective process evolves too.

# Sprint Retrospective - Best Practices



- **Timebox the meeting:** Keep the retrospective focused and concise, typically ranging from 1 hour to 1.5 hours, depending on the length of the sprint.
- **Use various Techniques:** Employ different formats and activities to keep retrospectives engaging and productive, such as the "Start, Stop, Continue" method or the "4Ls" (Liked, Learned, Lacked, Longed for).
- **Focus on Team Dynamics:** Encourage team members to express their thoughts on both technical aspects and interpersonal relations.
- **Follow up:** Ensure that the action items identified in the retrospective are tracked and revisited in future retrospectives to measure their effectiveness.

# Sprint Planning Vs Iteration Planning



- Iteration planning and sprint planning are similar concepts within Agile methodologies, particularly in Scrum, but they are not entirely the same.
- **Iteration Planning:**
  - Iteration planning is a term more commonly used in environments where Agile principles are applied but may not strictly follow the Scrum framework. It refers to the planning sessions for iterations (which can be similar to sprints)
  - Focus: Similar to sprint planning, iteration planning focuses on selecting work items to be completed in the upcoming iteration, as well as discussing how the team will approach the work. The specifics and techniques may vary based on the Agile framework used (e.g., Kanban, XP, SAFE).



# Sprint Planning Vs Iteration Planning - key differences

- **Framework Context** : Sprint planning is specific to the Scrum framework, whereas iteration planning can refer to planning in various Agile frameworks like SAFE.
- **Terminology**: The terminology may differ based on the methodology being used; for example, other Agile frameworks may use different terms or structures for planning their iterations.
- **Timeframes** : The specific timeframes and rituals associated with sprint planning (like daily scrums, sprint reviews, and retrospectives) are not necessarily part of iteration planning in other frameworks.

# Iteration planning Guidelines



- **Collaboration** : Encourage participation from the entire team, including Product Owners, Scrum Masters, and development team members. Collaboration helps ensure alignment on goals and understanding of tasks.
- **Time-Boxing**: Keep iteration planning sessions time-boxed (e.g., 2 hours for a 2-week sprint in Scrum) to maintain focus and efficiency.
- **Set a Clear Focus** : Begin by defining the iteration's goals. The team should understand what they aim to achieve and how it contributes to the overall project or product vision.
- **Prioritise Work**: Use the product backlog to prioritize features or user stories based on business value, urgency, and dependencies. This helps the team work on the most impactful tasks first.



# What is SAFe



# SAFe Scrum - Scaled Agile Framework

<https://scaledagileframework.com/safe-scrum/>



- SAFe, or the Scaled Agile Framework, is a set of organization and workflow patterns for implementing agile practices at enterprise scale
- The framework is a body of knowledge that includes structured guidance on roles and responsibilities, how to plan and manage the work, and values to uphold
- In essence, SAFe helps organizations deliver software and technology solutions faster, with higher quality, and with better alignment to overall business objectives
- It's the world's most trusted system for business agility because it works: it's trusted, customizable, and sustainable



# Key Points about SAFe



- **Scalability:** SAFe is designed to help businesses that need to scale their Agile practices beyond small teams of 10-15 individuals
- **Alignment:** It aligns collaboration across large numbers of Agile teams towards achieving enterprise-wide goals
- **Coordination:** It provides coordination and synchronization for multiple Agile teams in an enterprise
- **Built-In Quality:** It ensures that every increment produced by every Agile team in the enterprise meets required quality standards before it's released

# Iteration planning : SAFe specific Guidelines



- **Program Increment (PI) Planning** : In SAFe, iteration planning occurs within the context of a broader Program Increment planning cycle, where teams align on goals and dependencies across the ART (Agile Release Train).
- **Align with Business Objectives**: Ensure iteration goals are in sync with the objectives set out in the PI planning. This helps maintain alignment and focus on delivering business value.
- **Cross-Team Dependencies** : Identify and communicate dependencies with other teams that may impact planning and execution. This promotes collaboration and minimises blockers.
- **Capacity Planning**: Understand the team's capacity (taking into account holidays, team member availability, etc.) to ensure realistic commitments are made.
- **Risk Management**: Discuss potential risks during the planning session, and consider strategies to address risks proactively.

# Prioritising Stories (WSJF technique from SAFe)



- **Weighted Shortest Job First (WSJF)** is a prioritisation model used to sequence jobs (for example, Features, Capabilities, and Epics) to produce maximum economic benefit.
- In SAFe, WSJF is estimated as the Cost of Delay (CoD) divided by the job duration.
- The jobs with the highest WSJF deliver the best economic outcomes

$$\text{WSJF} = \frac{\text{Cost of Delay}}{\text{Job Duration}}$$

*If effort and CoDs are different,  
do the Weighted Shortest Job First!*




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

# Calculating the Cost of delay

- CoD is an estimate at best; it's hard for anyone to know the actual value of a new job (a new feature) that has yet to be delivered to market

$$\text{Cost of Delay} = \text{User - Business Value} + \text{Time Criticality} + \text{Risk Reduction and/or Opportunity Enablement}$$

© Scaled Agile, Inc.

User-Business Value	Time Criticality	Risk Reduction and/or Opportunity Enablement
 <p><b>What is the relative value to the Customer or business?</b></p> <ul style="list-style-type: none"> <li>• Do our users prefer this over that?</li> <li>• What is the revenue impact on our business?</li> <li>• Is there a potential penalty or other negative effects if we delay?</li> </ul>	 <p><b>How does user/business value decay over time?</b></p> <ul style="list-style-type: none"> <li>• Is there a fixed deadline?</li> <li>• Will they wait for us or move to another Solution?</li> <li>• What is the current effect on Customer satisfaction?</li> </ul>	 <p><b>What else does this do for our business?</b></p> <ul style="list-style-type: none"> <li>• Reduce the risk of this or future delivery?</li> <li>• Is there value in the information we will receive?</li> <li>• Enable new business opportunities?</li> </ul>

© Scaled Agile, Inc.

# Story Point in Scrum

- **Story point** is a unit of measure used to estimate the complexity, effort, and time required to complete a user story or a piece of work.
- **Relative Measurement:** Story points are not typically tied to hours or days; instead, they provide a way to assess effort relative to other user stories.
- For example, a story that is estimated at 5 story points is considered to be more complex or require more effort than one estimated at 3 story points.



# Story Point in Scrum (contd-)

- **Factors Influencing Estimation:** Various factors are considered when assigning story points, including:
  - **Complexity:** How complicated is the task?
  - **Effort:** How much work is needed to achieve the task?
  - **Risk:** How uncertain or unpredictable is the task?
- **Common Scale**
  - Many teams use the Fibonacci sequence (1, 2, 3, 5, 8, 13, etc.) to assign story points. This non-linear scale helps teams quickly differentiate between larger and smaller tasks and reflects the increasing uncertainty as tasks become larger.

# Story Point in Scrum (contd-)



- **Team Dynamics:**
  - The team collectively estimates story points, often using techniques like Planning Poker, where team members discuss and agree on the estimation collaboratively.
  - This encourages team ownership and understanding of the work involved.
- **Velocity:**
  - Once story points are assigned, teams can track their performance over time using "velocity", which is the number of story points completed in a given sprint.

# Agile Development : Recap

- Agile development is a software development methodology that emphasizes flexibility, collaboration, customer feedback, and iterative progress.
- It contrasts with traditional, rigid project management approaches and is designed to adapt to changing requirements throughout the development process.

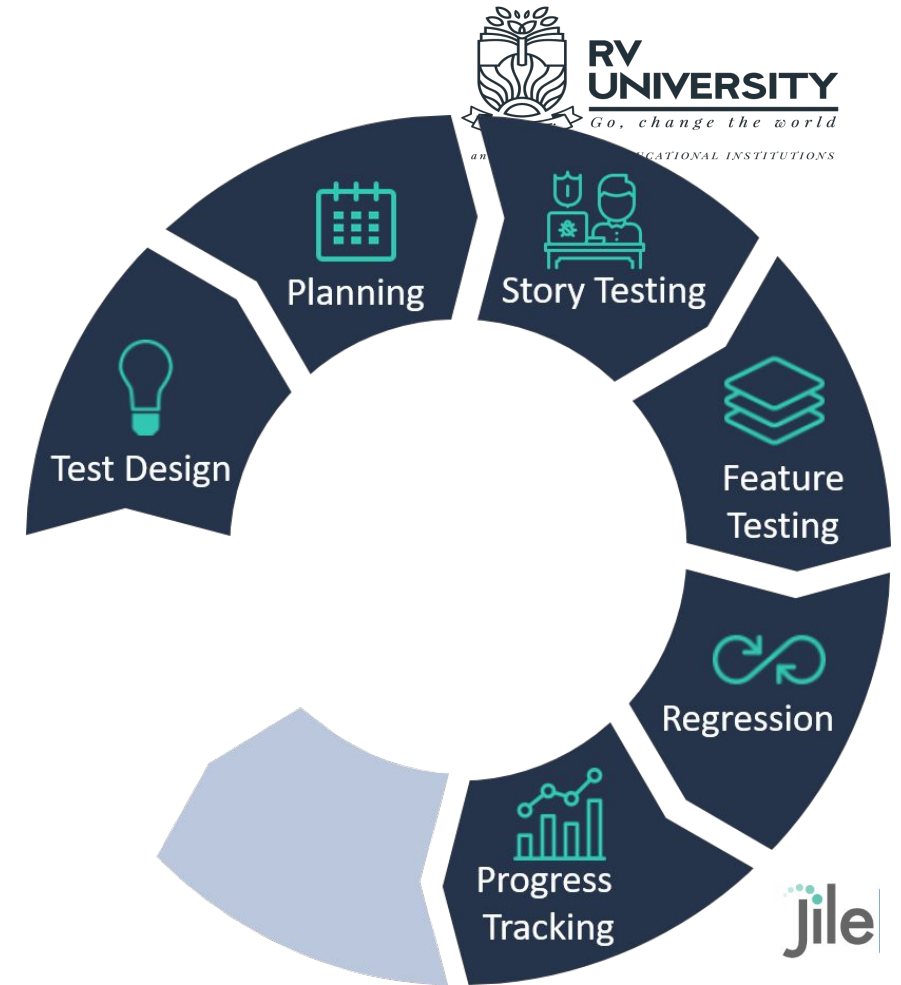


Javatpoint

**Fig. Agile Model**

# Agile Testing

- Agile testing is a software testing practice that follows the principles of Agile development.
- It emphasises
  - collaboration,
  - flexibility, and
  - customer satisfaction,
- focusing on delivering high-quality software products.



# Testing

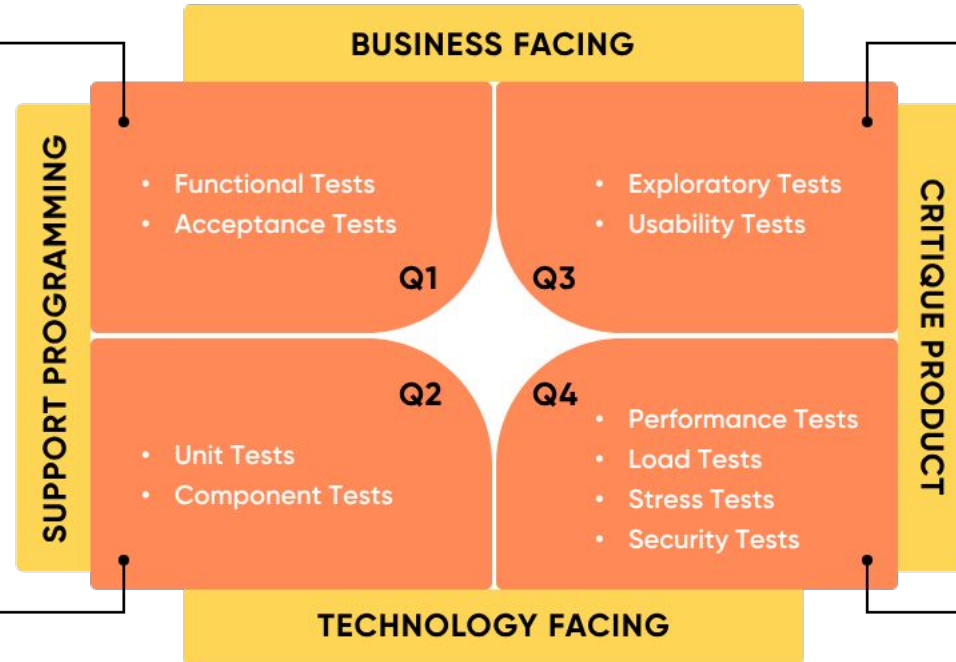
## types

### Automated & by Humans

- Selenium Webdriver
- Cypress Dtt
- Protractor
- TestProect
- Percy
- Applitools
- BackstopJS
- NightwatchJS

### Automated

- Jmeter
- Blazemeter
- ZAP
- SonarQube



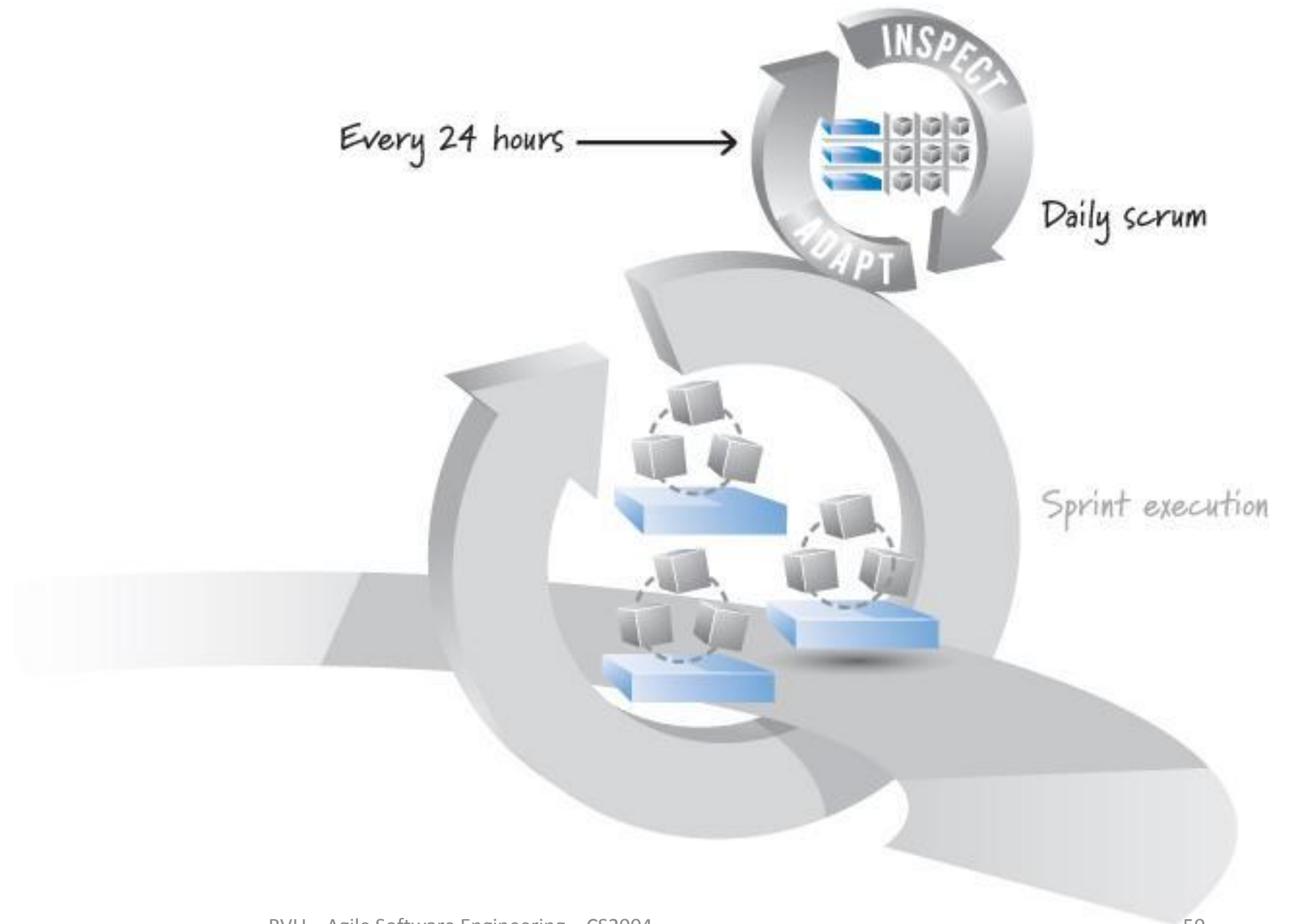
### Humans

- Formal Techniques
- Test Charters
- Xray
- Zephyr
- TestRail

### Tools

- Jmeter
- Blazemeter
- ZAP
- SonarQube

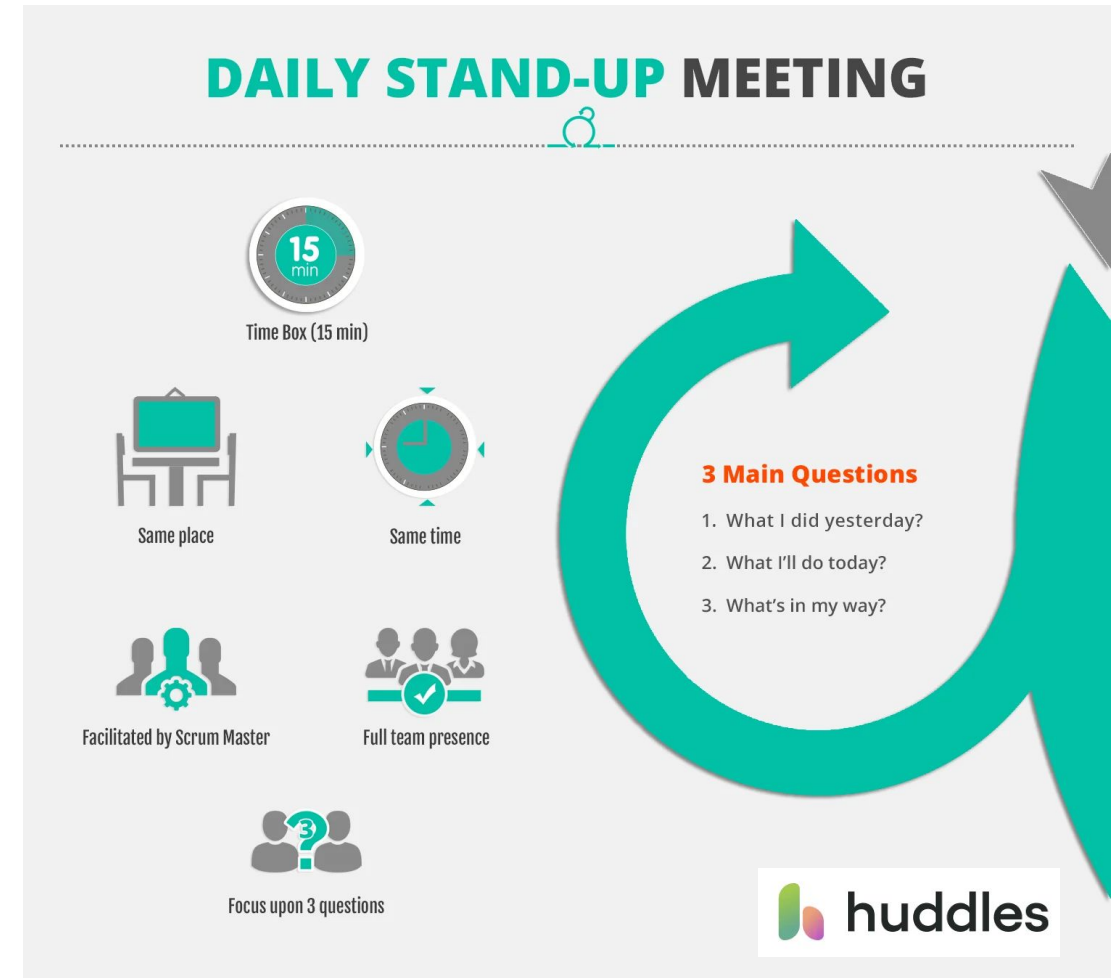
# Daily Scrum





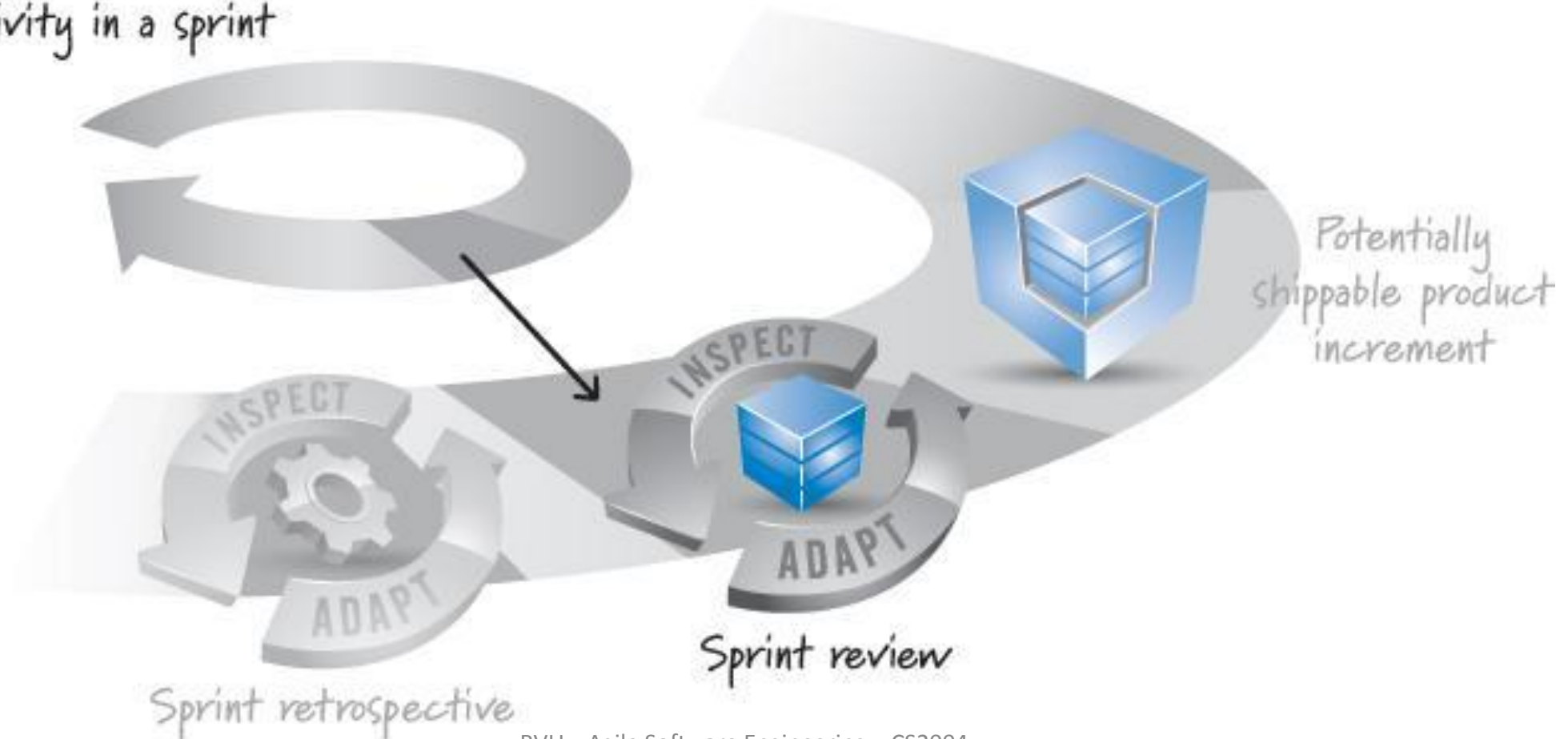
# Daily stand-up meeting

- A daily stand-up meeting, often referred to simply as a "stand-up," is a core component of Agile and Scrum methodologies.
- It is a short, time-boxed meeting typically held each day, and it usually lasts around 15 minutes.
- The primary purpose of the stand-up is to promote team communication and collaboration, to keep everyone on track, and to address any impediments or blockers that may hinder progress.



# Sprint Review

Sprint review is the next-to-last activity in a sprint



# Sprint Review



## 1. Purpose:

- **Focus on Deliverables:** The primary purpose of the Sprint Review is to inspect the work that was done during the sprint and to assess whether the product increment (the potentially shippable product) meets the Sprint Goal and is ready for release
- **Feedback and Collaboration:** It serves as a platform for stakeholders to provide feedback on the increment and to collaborate on future product development

## 2.Participants:

- **Scrum Team:** This includes the Development Team (those who built the product increment) and the Scrum Master
- **Product Owner:** The Product Owner plays a key role in reviewing the increment and ensuring that it aligns with the product vision and user needs
- **Stakeholders:** Anyone with an interest in the product, such as customers, end-users, management, and other relevant parties, can attend to provide feedback

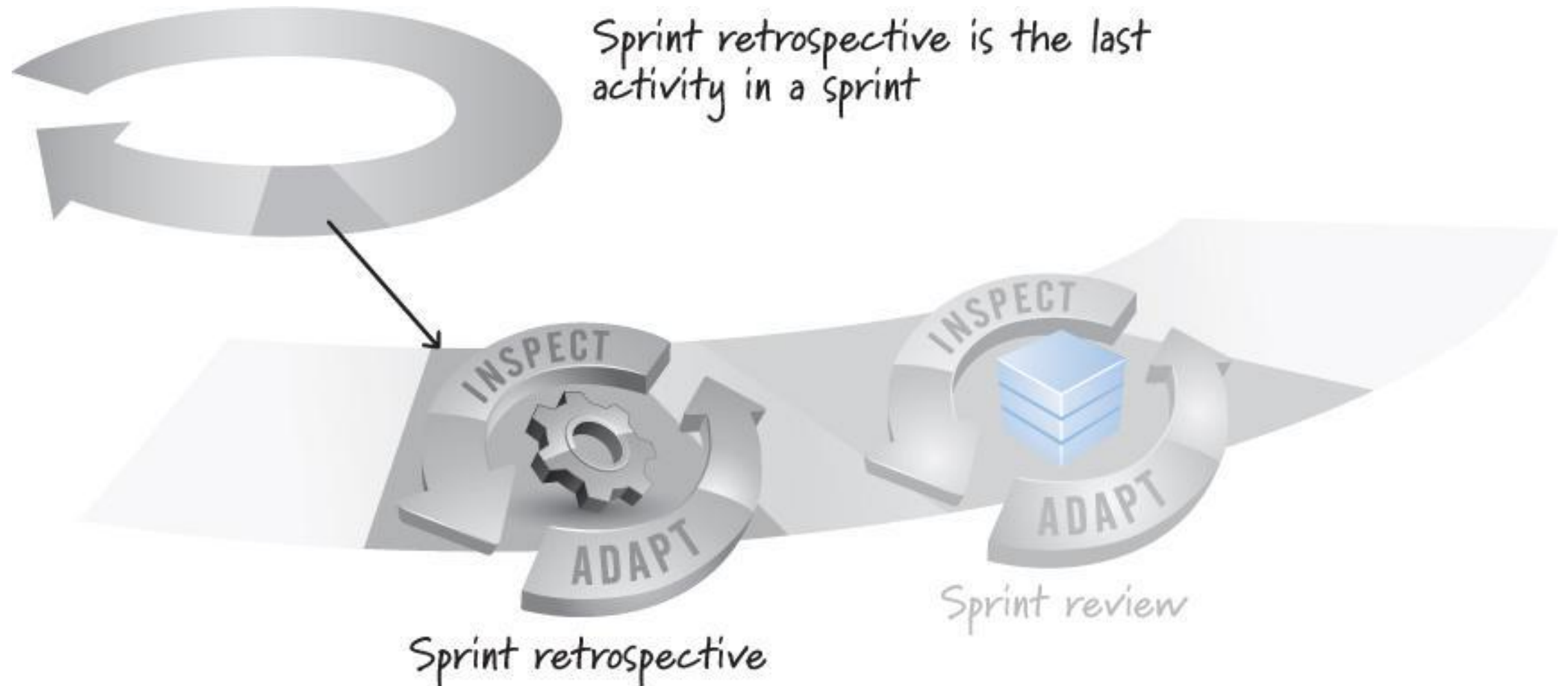
## 3.Activities:

- **Demo:** The Development Team demonstrates the product increment, showcasing the completed work during the sprint
- **Feedback:** Stakeholders provide feedback and raise questions or concerns about the product
- **Review of the Sprint Goal:** The Product Owner verifies if the Sprint Goal has been achieved
- **Adaptation:** Based on the feedback received, the Product Backlog is updated, and the Product Owner decides what to do next

## 4. Output:

- The output of the Sprint Review may include updated Product Backlog items, new user stories, and a better understanding of the product's direction

# Sprint Retrospective



# Sprint Retrospective

## 1.Purpose:

- **Process Improvement:** The Sprint Retrospective focuses on the Scrum process and team collaboration rather than the product itself.
- **Continuous Improvement:** Its primary purpose is to inspect the team's performance during the sprint and identify areas for improvement in processes, communication, and teamwork

## 2.Participants:

- **Scrum Team:** This includes the Development Team, Scrum Master, and Product Owner (though the Product Owner's participation is typically optional)

## 3.Activities:

- **Reflection:** The team reflects on the previous sprint, discussing what went well, what didn't go well, and what could be improved
- **Identification of Action Items:** The team identifies specific action items or changes that can be made to improve processes, communication, or collaboration
- **Commitment to Improvement:** The team commits to implementing these changes in the next sprint

## 4.Output:

- The Sprint Retrospective results in a set of action items that the team agrees to implement in the upcoming sprint. These action items are intended to drive continuous improvement in the team's processes and practices

# Sprint Review vs Sprint Retrospective



- In summary, while both the Sprint Review and Sprint Retrospective occur at the end of a sprint in Scrum, they serve distinct purposes
- The Sprint Review focuses on the product and its readiness for release, involving stakeholders for feedback
- The Sprint Retrospective, on the other hand, centers on the team's performance and aims to identify opportunities for process improvement, with the team itself driving the changes
- Both ceremonies are essential for fostering transparency, collaboration, and continuous improvement in agile development projects.

# Definition of Ready

A checklist of conditions that must be true before a product backlog item is considered ready to pull into a sprint during sprint planning

Definition of Ready	
<input type="checkbox"/>	Business value is clearly articulated.
<input type="checkbox"/>	Details are sufficiently understood by the development team so it can make an informed decision as to whether it can complete the PBI.
<input type="checkbox"/>	Dependencies are identified and no external dependencies would block the PBI from being completed.
<input type="checkbox"/>	Team is staffed appropriately to complete the PBI.
<input type="checkbox"/>	The PBI is estimated and small enough to comfortably be completed in one sprint.
<input type="checkbox"/>	Acceptance criteria are clear and testable.
<input type="checkbox"/>	Performance criteria, if any, are defined and testable.
<input type="checkbox"/>	Scrum team understands how to demonstrate the PBI at the sprint review.



# Product Backlog Item (Examples)

PBI Type	Example
Feature	As a customer service representative I want to create a ticket for a customer support issue so that I can record and manage a customer's request for support.
Change	As a customer service representative I want the default ordering of search results to be by last name instead of ticket number so that it's easier to find a support ticket.
Defect	Fix defect #256 in the defect-tracking system so that special characters in search terms won't make customer searches crash.
Technical improvement	Move to the latest version of the Oracle DBMS.
Knowledge acquisition	Create a prototype or proof of concept of two architectures and run three tests to determine which would be a better approach for our product.

# Sprint Backlog

- The Sprint Backlog is a subset of the Product Backlog that the Development Team commits to delivering during a Sprint
- It is created during Sprint Planning and is a plan for the Sprint. The Sprint Backlog is a living document that can be updated daily during the Daily Scrum as the team works towards completing the Sprint Goal

# What is Program Increment

- The term “Program Increment” refers to the sum of all the Product Backlog items (PBIs) that have been completed during a Sprint, plus the work from all previous Sprints.
- The Increment must be in a potentially releasable state, meaning it meets the team's Definition of Done and is ready for review

# Key points about the Increment (1 of 2)



- **Continuous Integration:**

- The Increment is built incrementally, Sprint by Sprint. Each Sprint, the Development Team adds new functionality or improvements to the existing Increment. This is achieved through continuous integration, where the work of individual team members is integrated into a central codebase frequently

- **Definition of Done:**

- The Increment is subject to the team's Definition of Done (DoD). The Definition of Done is a set of criteria that the Increment must meet to be considered complete and potentially releasable. The DoD ensures a consistent and high level of quality for each Increment

- **Ready for Review:**

- At the end of each Sprint, the Increment should be in a state where it can be reviewed by stakeholders. This means that it is not only developed but also tested, documented, and meets the acceptance criteria defined for the user stories in the Sprint Backlog

# Key points about the Increment (2 of 2)



- **Accumulation of Value:**

- Over time, the Increment accumulates value as more features, improvements, and fixes are added. It represents the evolving and potentially shippable product that can be released to users or customers at any time

- **Visibility and Transparency:**

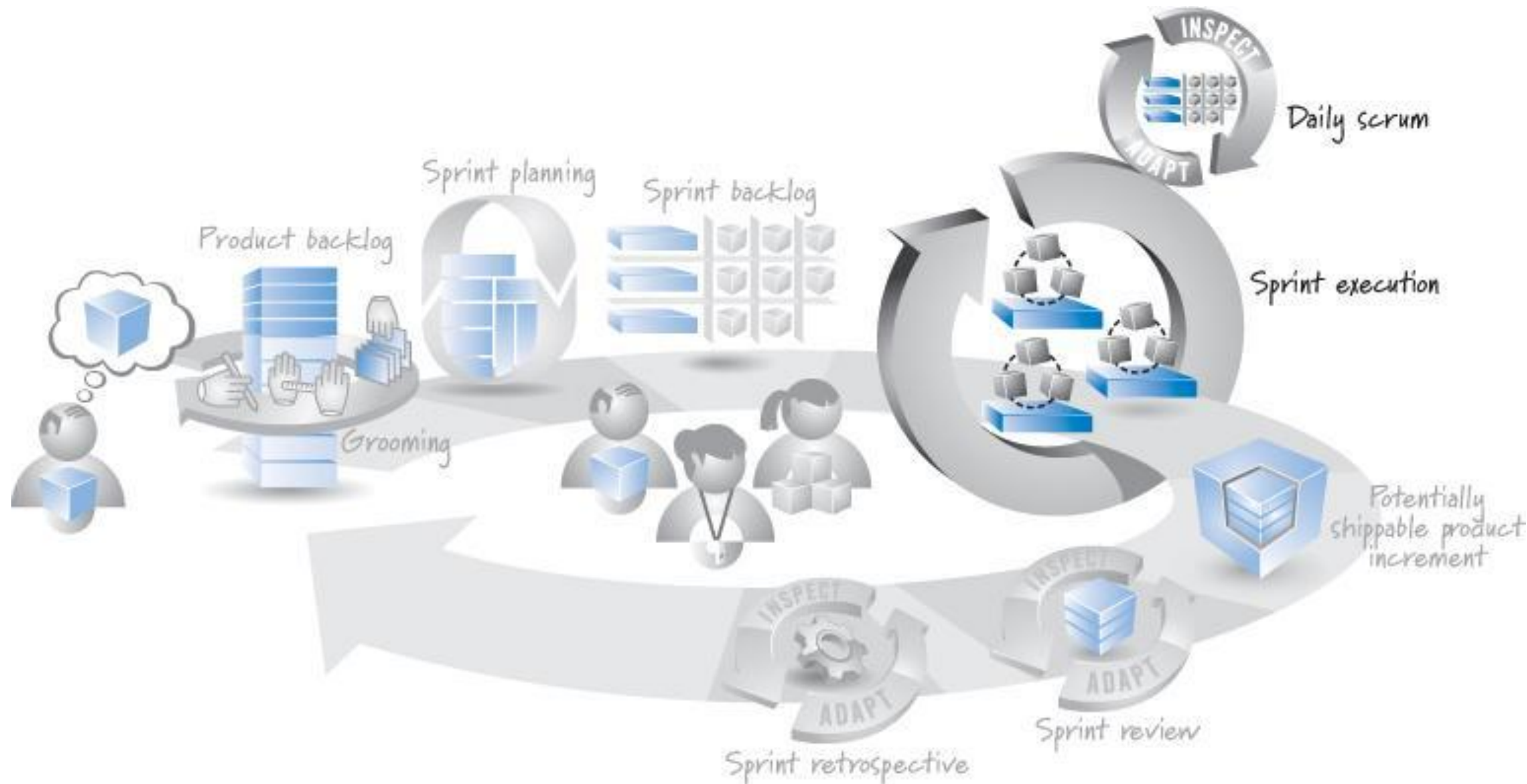
- The Increment serves as a tangible and visible result of the team's work. It provides transparency into the progress made in each Sprint and allows stakeholders to assess the state of the product at the end of every Sprint Review

- **Inspect and Adapt:**

- The Increment is inspected during the Sprint Review, where stakeholders provide feedback. This feedback is used to adapt the Product Backlog, adjust priorities, and plan the next steps for the product

Definition of Done	
<input type="checkbox"/>	Design reviewed
<input type="checkbox"/>	Code completed
<input type="checkbox"/>	Code refactored
<input type="checkbox"/>	Code in standard format
<input type="checkbox"/>	Code is commented
<input type="checkbox"/>	Code checked in
<input type="checkbox"/>	Code inspected
<input type="checkbox"/>	End-user documentation updated
<input type="checkbox"/>	Tested
<input type="checkbox"/>	Unit tested
<input type="checkbox"/>	Integration tested
<input type="checkbox"/>	Regression tested
<input type="checkbox"/>	Platform tested
<input type="checkbox"/>	Language tested
<input type="checkbox"/>	Zero known defects
<input type="checkbox"/>	Acceptance tested
<input type="checkbox"/>	Live on production servers











# Sprint Backlog with Estimated Effort Remaining Each Day

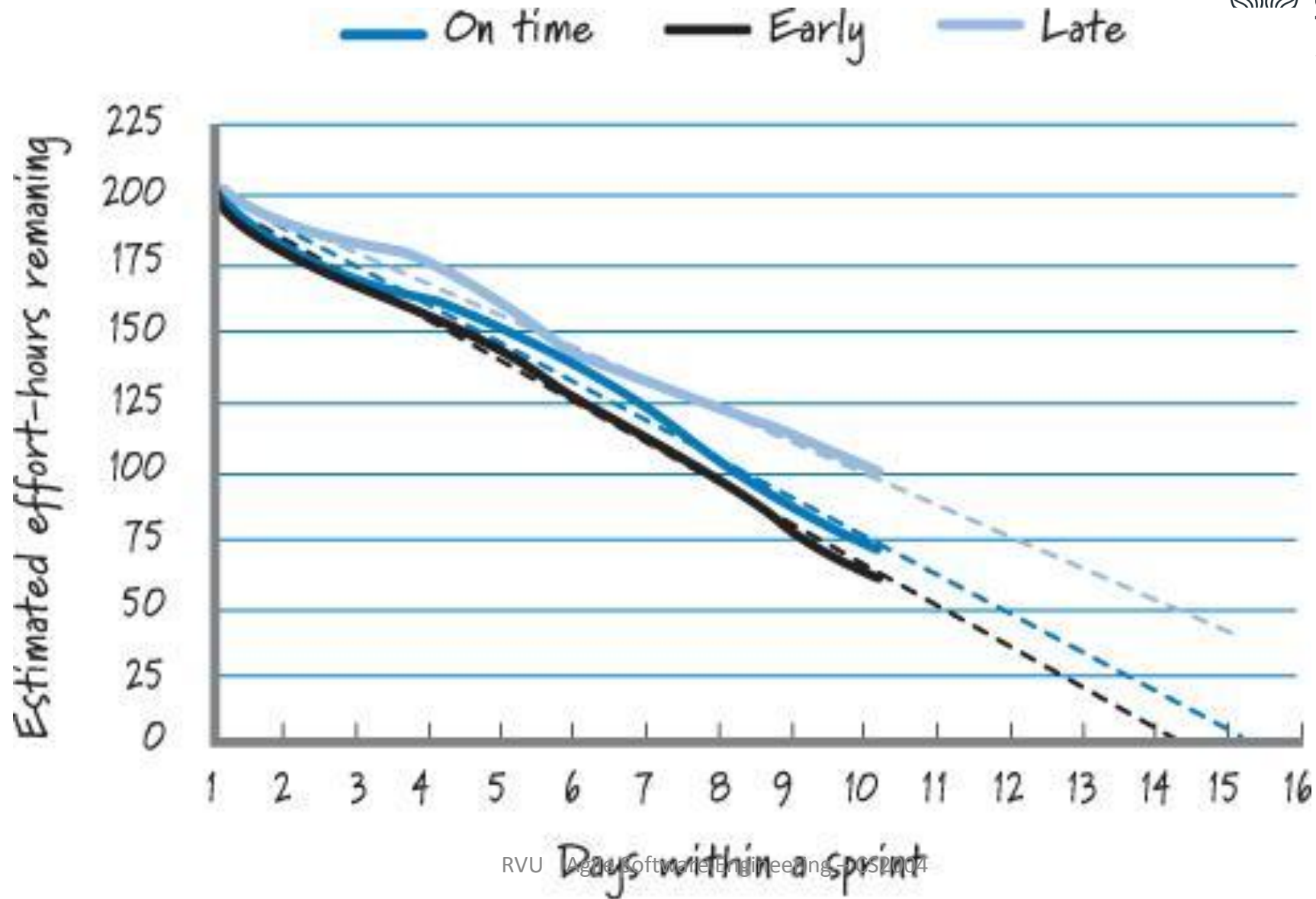
Tasks	D1	D2	D3	D4	D5	D6	D7	D8	D9	...	D15
Task 1	8	4	4	2							
Task 2	12	8	16	14	9	6	2				
Task 3	5	5	3	3	1						
Task 4	7	7	7	5	10	6	3	1			
Task 5	3	3	3	3	3	3	3				
Task 6	14	14	14	14	14	14	14	8	4		
Task 7						8	6	4	2		
Tasks 8–30	151	139	143	134	118	99	89	101	84		0
Total	200	180	190	175	155	130	115	113	90		0

# Sprint burndown chart

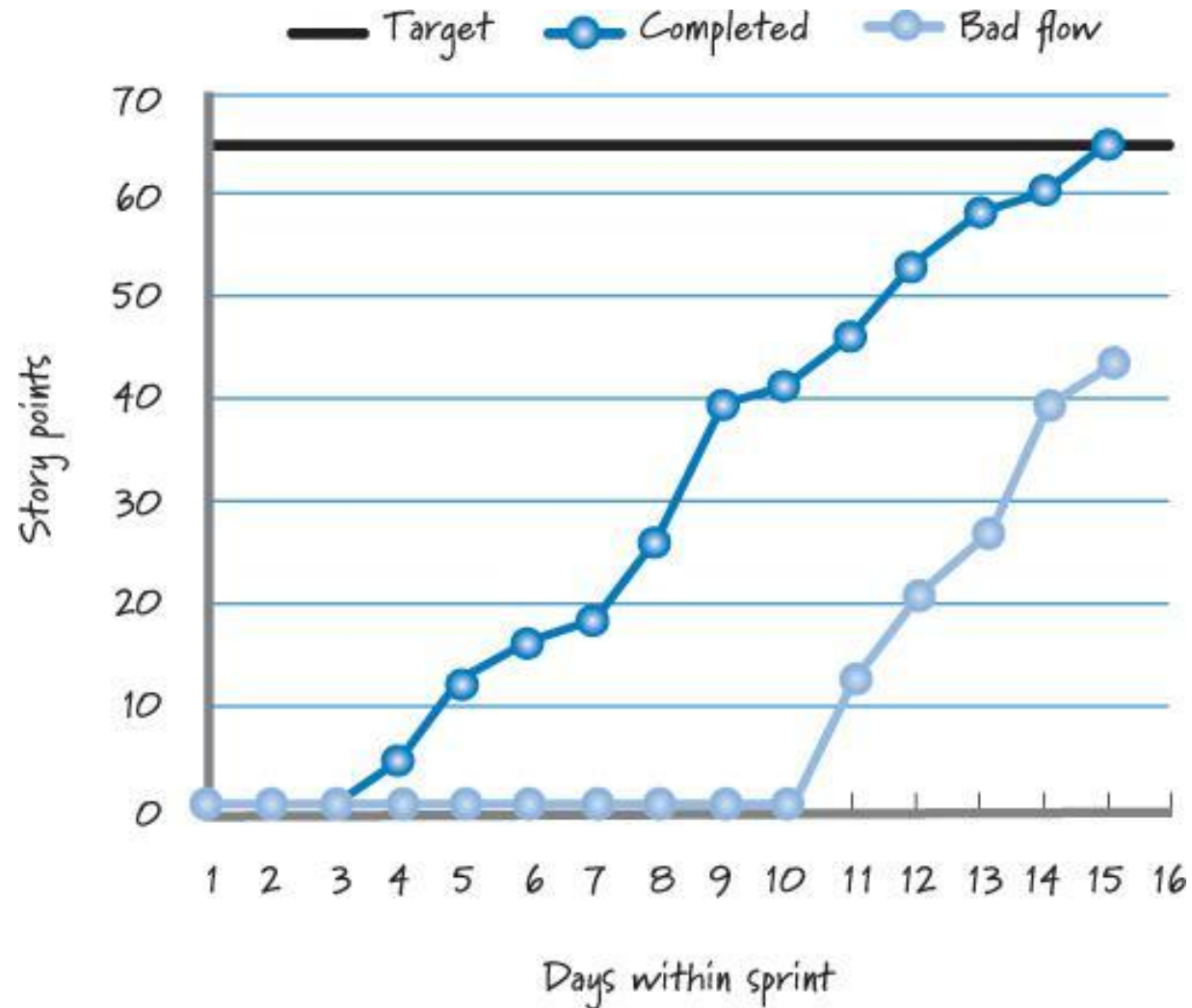




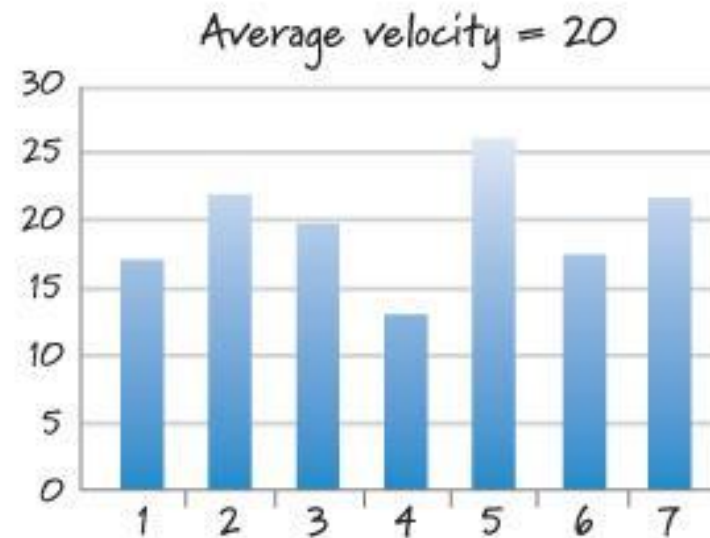
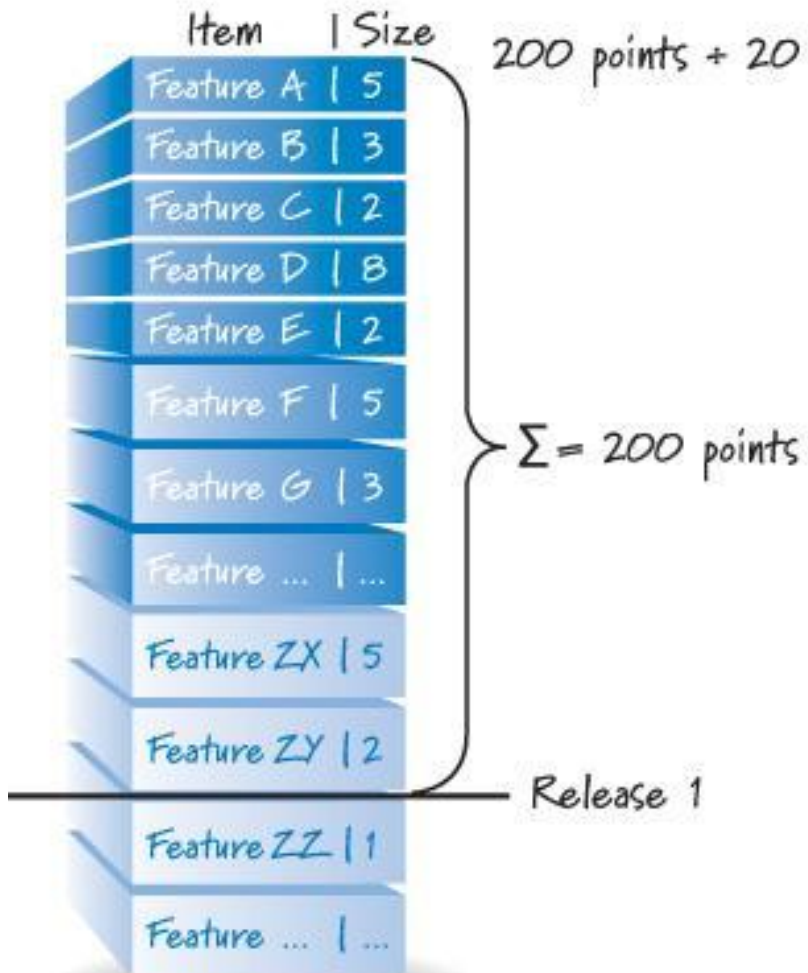
# Sprint burndown chart



# Sprint burnup chart



Estimated size + measured velocity = (number of sprints)





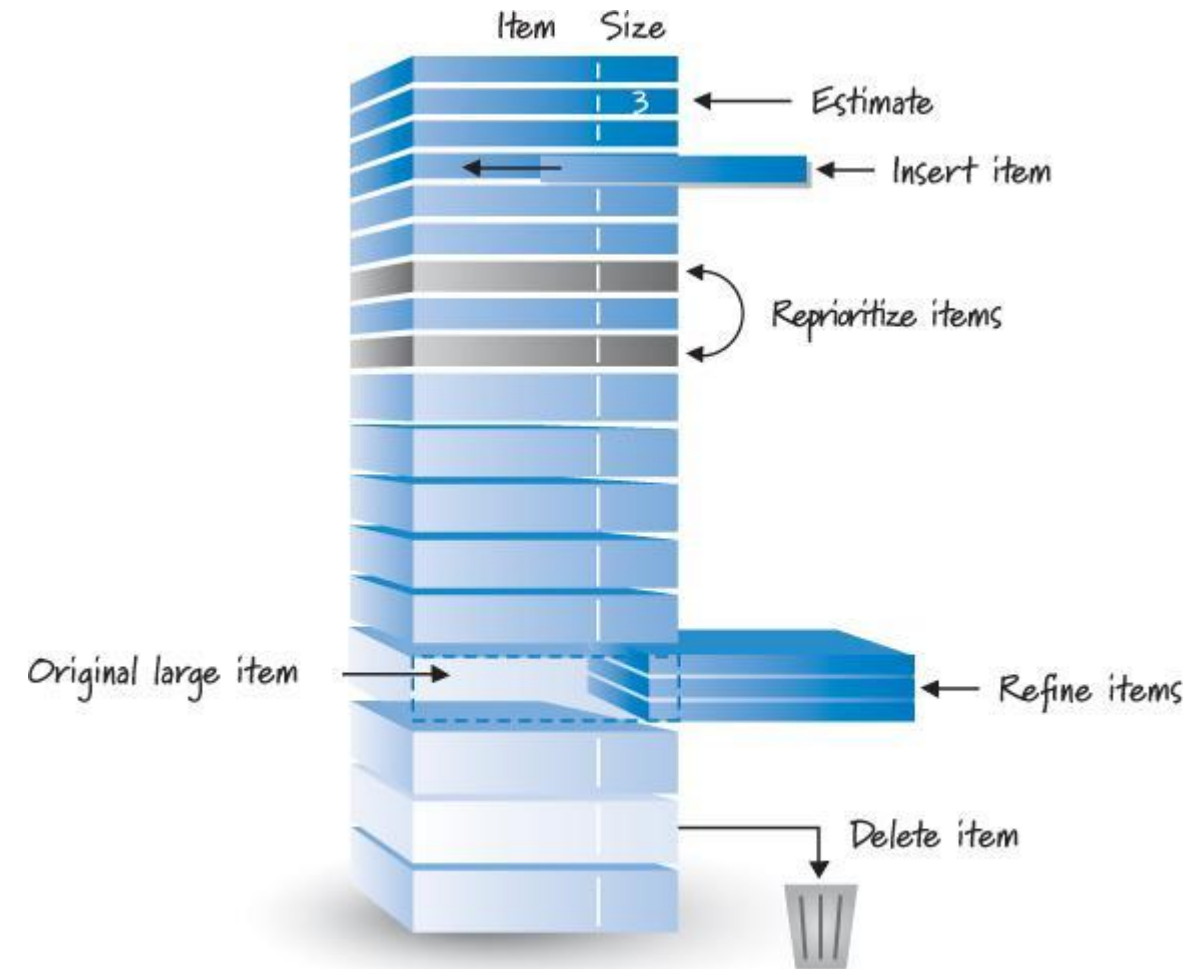
- When planning and managing the development of a product, we need to answer important questions such as
  - “How many features will be completed?”
  - “When will we be done?” and
  - “How much will this cost?”
- To answer these questions using Scrum, we need to estimate the size of what we are building and measure the velocity or rate at which we can get work done.
- With that information, we can derive the likely product development duration (and the corresponding cost) by dividing the estimated size of a set of features by the team’s velocity

# Velocity



- A measure of the amount of work a team can complete in a given sprint
- typically expressed in terms of story points, hours, or any other unit of measure that the team uses for estimating work.
- It helps teams gauge their productivity over time and is a key metric in Agile project management.
- **Measurement:** Velocity is calculated at the end of a sprint by summing the story points (or equivalent measure) for all the user stories that have been completed (i.e., which meet the Definition of Done) during that sprint.
- Many teams use the Fibonacci sequence (1, 2, 3, 5, 8, 13, etc.) to assign story points. This non-linear scale helps teams quickly differentiate between larger and smaller tasks and reflects the increasing uncertainty as tasks become larger.

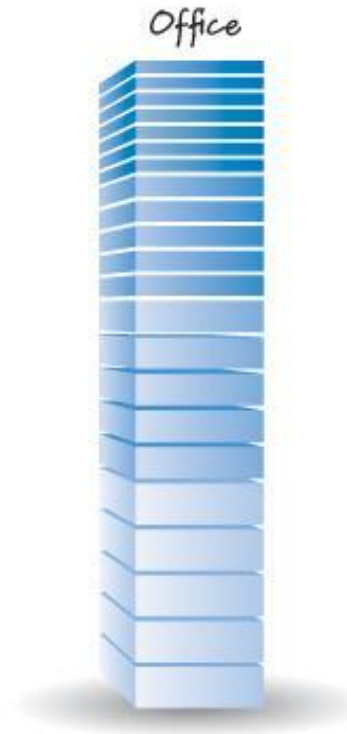
Grooming refers to a set of three principal activities: creating and refining (adding details to) PBIs, estimating PBIs, and prioritizing PBIs.

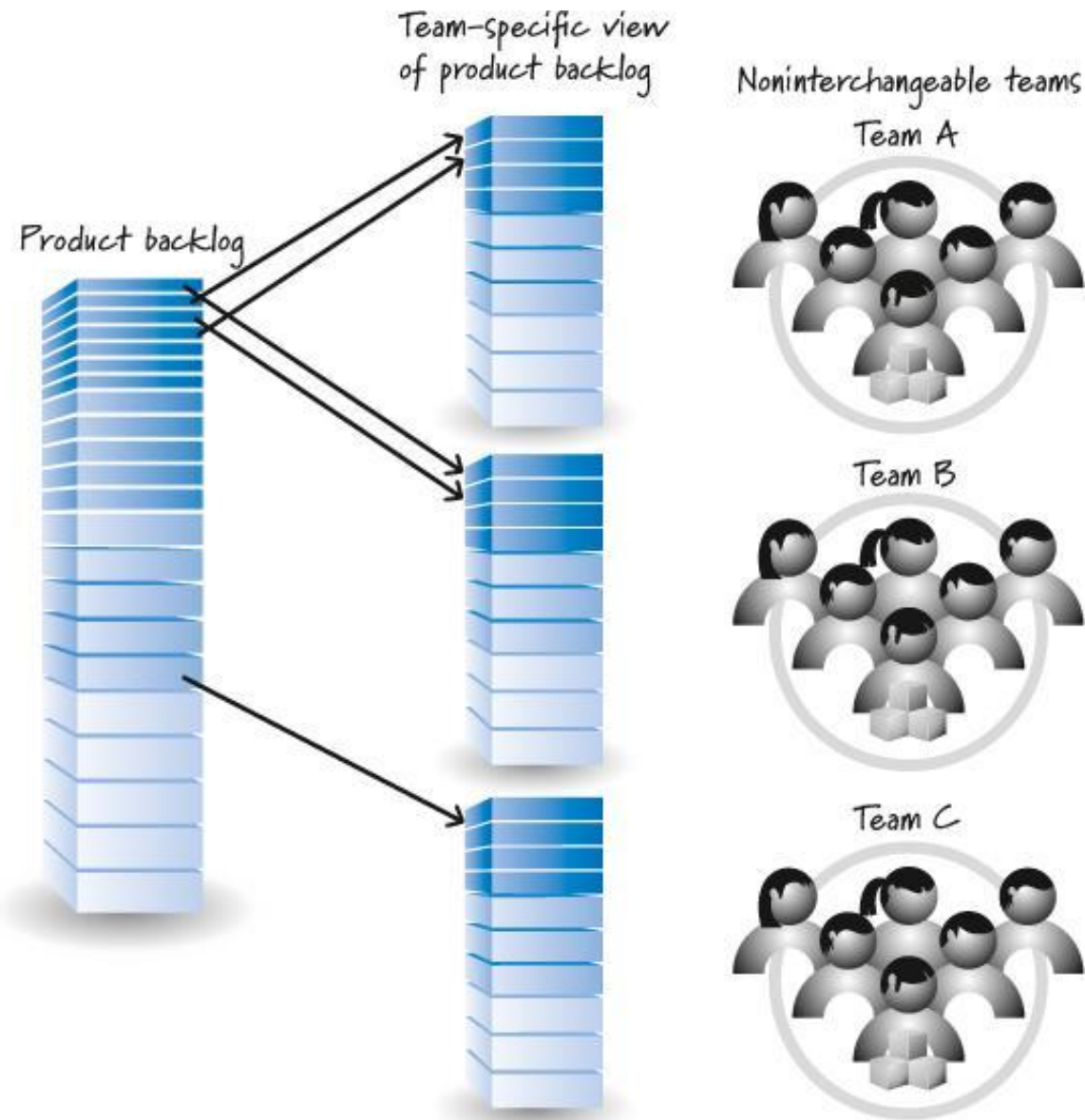


Product backlog per application



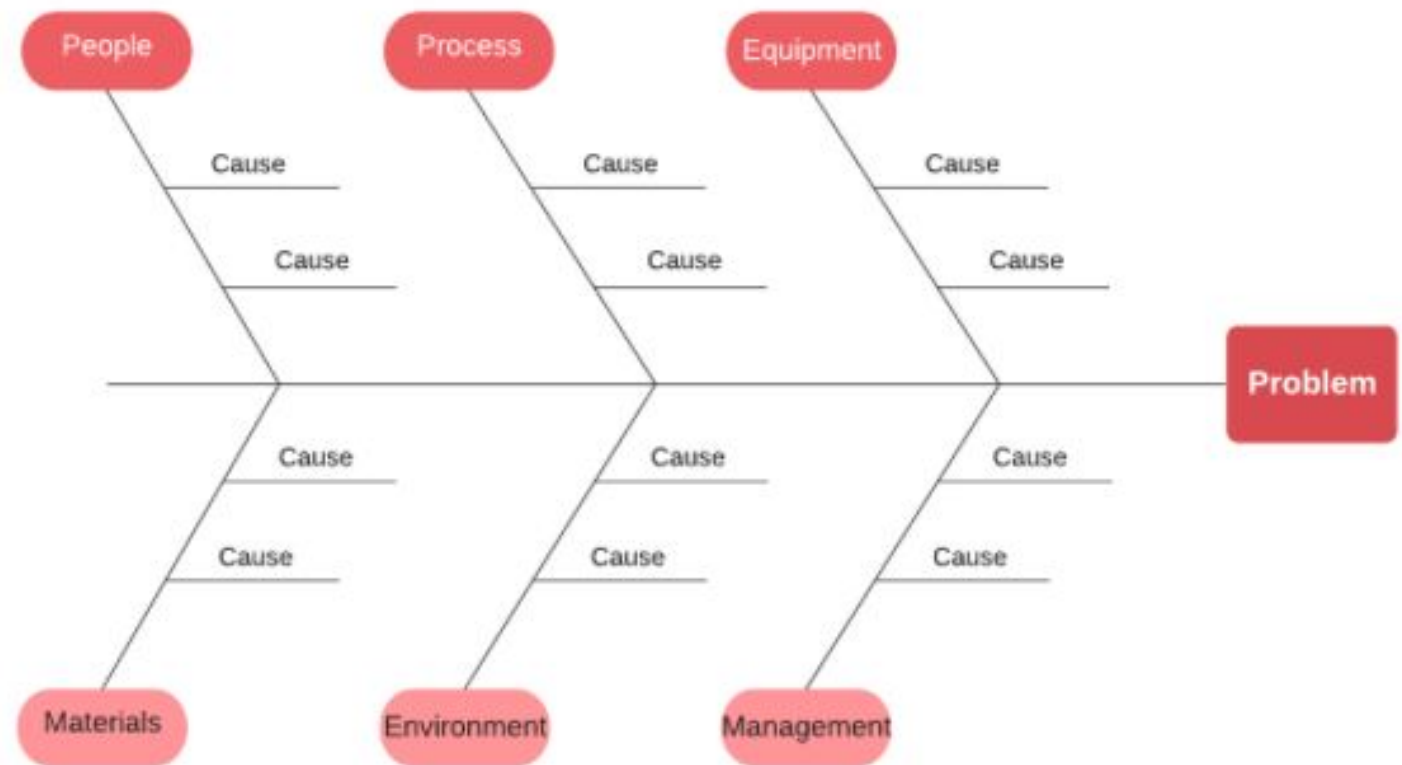
Product backlog for the suite





# Inspect & Adapt (Fishbone model)

- The **Inspect & Adapt (I&A) Fishbone Diagram** is a tool used in Agile methodologies, particularly within the Scaled Agile Framework (SAFe), to identify and analyze the root causes of problems or inefficiencies.
- It combines the principles of the Inspect & Adapt process with the structure of a Fishbone (Ishikawa) diagram.



# Inspect & Adapt (I&A)



**Purpose:** This is a significant event held at the end of each Program Increment (PI) in SAFe. It involves demonstrating the current state of the solution, evaluating progress, and identifying areas for improvement.

**Structure:** The I&A event includes a PI System Demo, a quantitative and qualitative measurement review, and a retrospective and problem-solving workshop.



# Fishbone Diagram

**Purpose:** Also known as the Ishikawa diagram, it helps teams categorize potential causes of a problem into specific branches, facilitating a structured approach to root cause analysis.

**Structure:** The diagram typically includes a central "spine" with branches representing different categories of potential causes, such as People, Processes, Tools, and Environment.

# What is Agile Release Train (ART)

# Agile Release Train (ART)



**ART** stands for **Agile Release Train**. It's a concept from the Scaled Agile Framework (SAFe) that represents a long-lived team of Agile teams, which, along with other stakeholders, incrementally develops, delivers, and where applicable operates, one or more solutions in a value stream. Here are some key characteristics of an ART:

- Long-lived:** An ART is a long-term team of Agile teams that incrementally develops, delivers, and often operates one or more solutions in a value stream
- Cross-functional:** ARTs are cross-functional and have all the capabilities needed to define, build, validate, release, and operate solutions

# Agile Release Train (ART) –contd.



- Aligned to a mission:** ARTs align to a shared business and technology mission
- Organized around value:** ARTs are organized around the enterprise's significant Development Value Streams
- Fixed schedule:** The train departs the station on a known, reliable schedule
- Continuous flow of value:** An ART delivers a continuous flow of value  
In essence, an Agile Release Train helps to align teams to a shared business and technology mission. It's like a virtual organization (typically 50 – 125 people) that plans, commits, develops, and deploys together. This facilitates the flow of value from ideation through deployment and release into operations.

**Agile Release Train (ART):** In SAFe, an ART is essentially a “long-term team of Agile teams”. It’s a **virtual** organization (typically 50 – 125 people) that plans, commits, develops, and deploys together. The ART aligns these teams to a shared business and technology mission.

The phrase “**long-term team of Agile teams**” refers to a group of Agile teams that work together over an extended period of time

An Agile Release Train (ART) typically consists of 50 to 125 people. This usually translates to about **5 to 12 Scrum teams**, assuming each team has 7 to 9 members, which is a common team size in Scrum. However, the exact number can vary depending on the specific needs and structure of the organization.

# A long-lived Agile Release Train

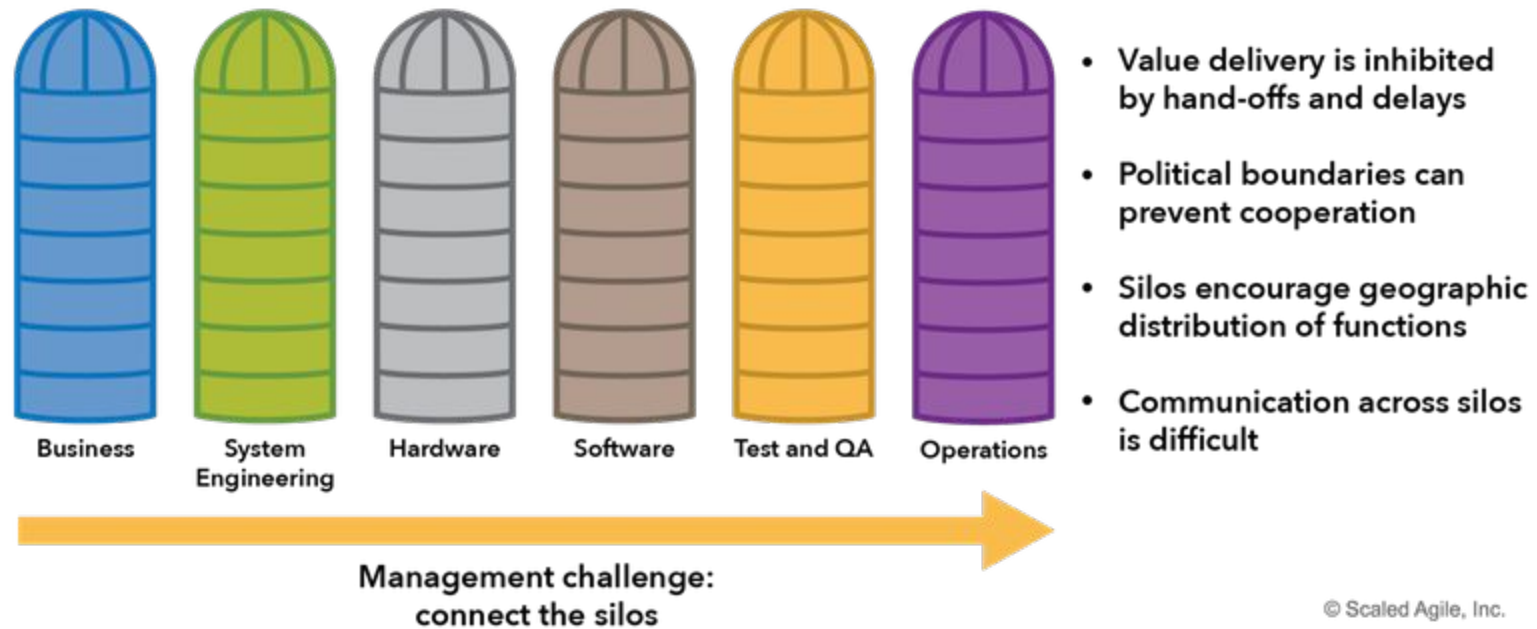
ARTs are cross-functional and have all the capabilities  
—software, hardware, firmware, and other  
—needed to define, implement, test, deploy, release, and where applicable, operate solutions.

- An ART delivers a continuous flow of value

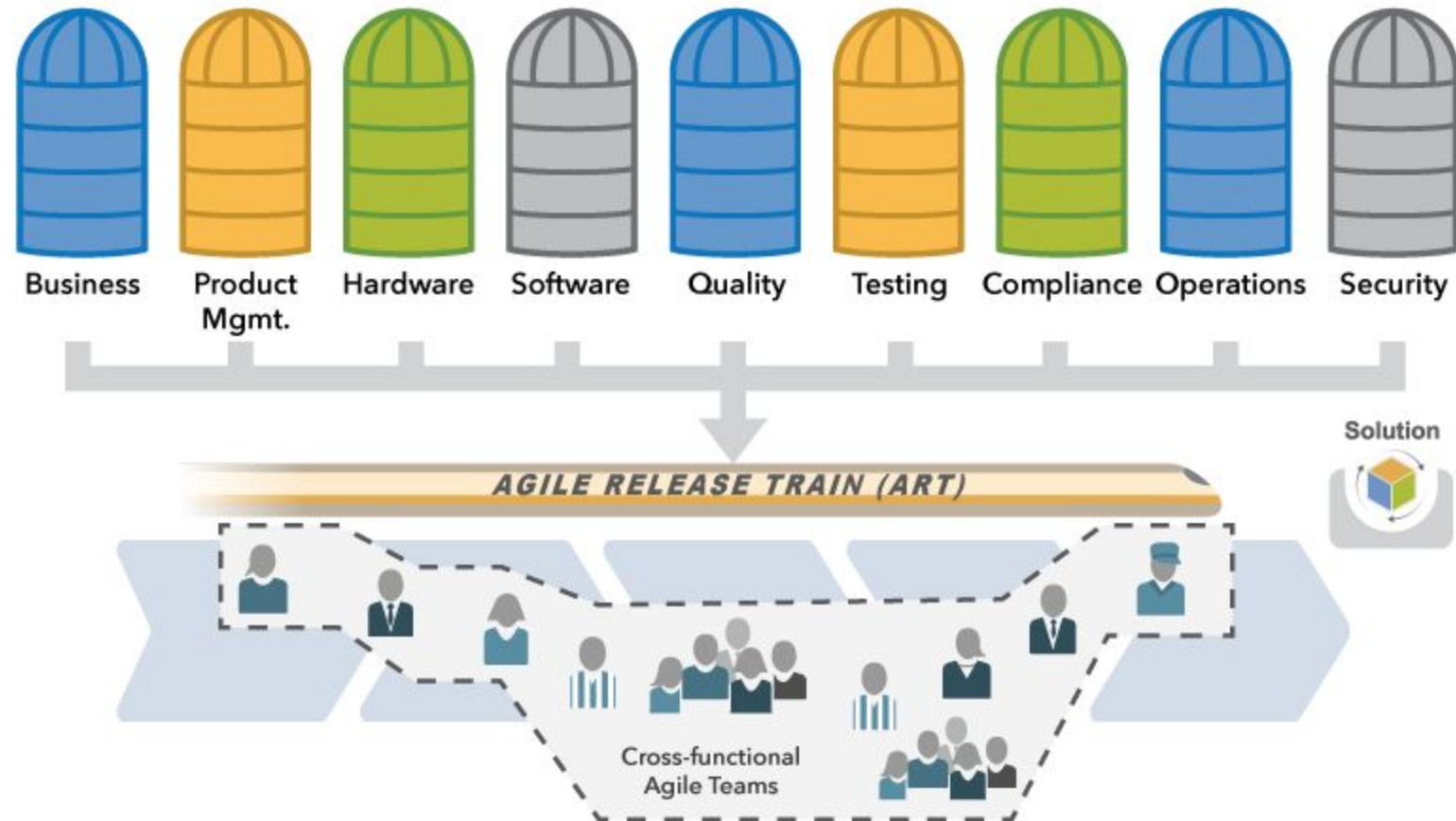




# Traditional functional organization

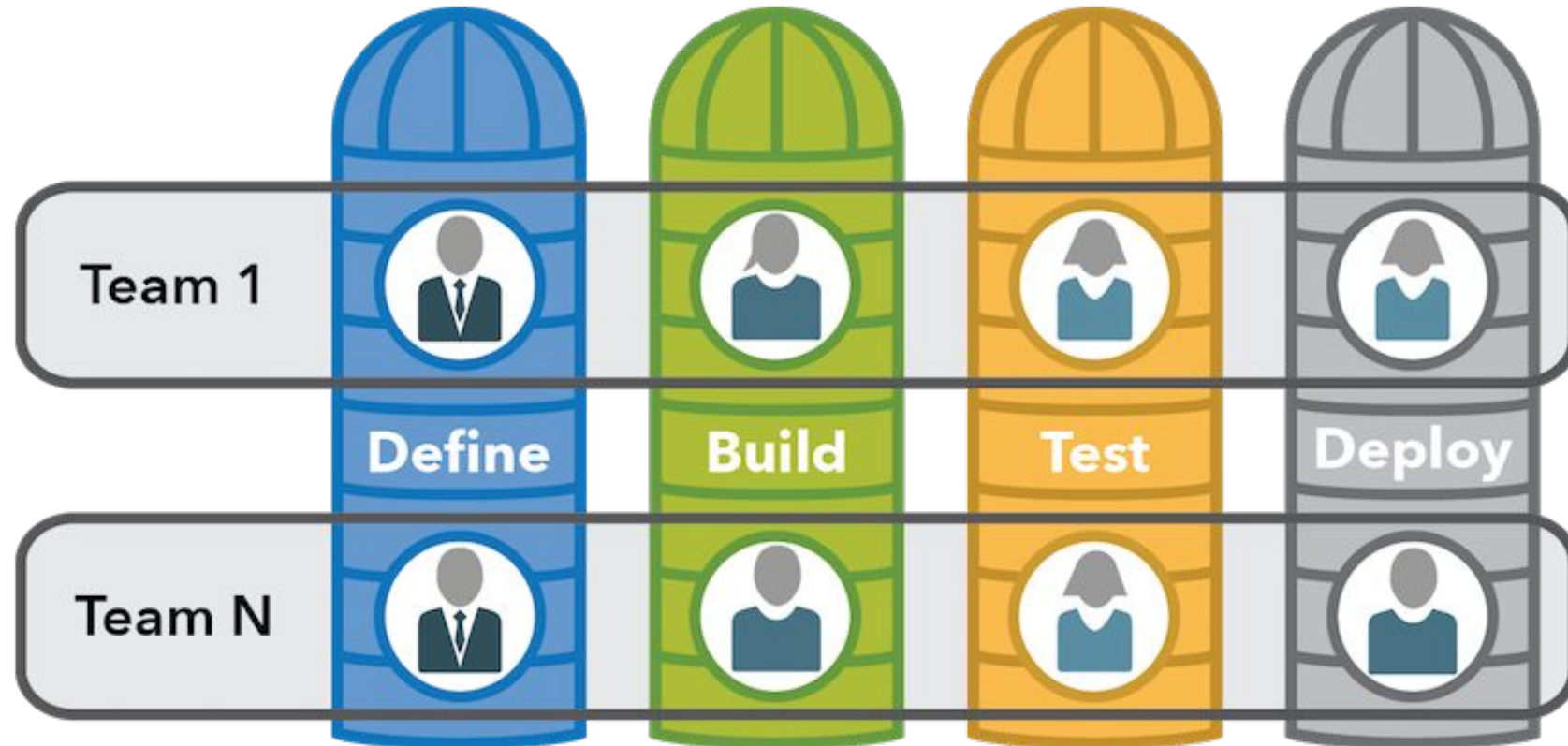


# Agile Release Trains are fully cross-functional



© Scaled Agile, Inc.

# Agile teams are cross-functional



© Scaled Agile, Inc.

# Agile Release Train Roles

# ART - Roles



In an Agile Release Train (ART), there are several key roles

- Agile Team Members:** These are the individuals who carry out the work of delivering value in the form of a product or service<sup>4</sup>
- System Architects / Engineers:** They are responsible for defining and communicating the technical vision for the solution being developed
- Scrum Master:** The Scrum Master helps the team to work together effectively and to continuously improve
- Business Owner:** The Business Owner is a key stakeholder in the ART who has the ultimate responsibility for the business outcome

# ART – Roles –contd.



- Product Owner:** The Product Owner is responsible for defining and prioritizing the team's backlog
  - Product Manager:** The Product Manager works with Product Owners and other stakeholders to define the program backlog
  - Release Train Engineer (RTE):** The RTE is a servant leader who facilitates program execution, removes impediments to flow, and manages risk and dependencies
- These roles work together to ensure that the ART can effectively deliver value to customers

# What is PI?



- In the context of SAFe (Scaled Agile Framework), PI stands for **Program Increment**
- A Program Increment (PI) is a time box during which an Agile Release Train (ART) delivers incremental value in the form of working, tested software and systems
- PIs are typically 8 – 12 weeks long, with most organizations using a default length of 10 weeks
- The PI includes both the development of new features and capabilities, as well as Innovation and Planning (IP) iteration
- The PI provides a development timeline for Agile teams within an ART

# Project Management – the Agile way

	Agile/Scrum/SAFe Term	Project Management Term
1	<b>Scrum Master</b>	Project Manager
2	<b>Product Owner</b>	Stakeholder/Customer Representative
3	<b>Development Team</b>	Project Team
4	<b>Product Backlog</b>	Project Requirements/Task List
5	<b>Sprint Backlog</b>	Work Breakdown Structure (WBS)
6	<b>Increment</b>	Deliverable/Milestone

# Project Management – the Agile way –Contd.

	Agile/Scrum/SAFe Term	Project Management Term
7	<b>Sprint Planning</b>	Planning Phase/Meeting
8	<b>Daily Scrum</b>	Daily Stand-up/Status Meeting
9	<b>Sprint Review</b>	Project Review/Checkpoint
10	<b>Sprint Retrospective</b>	Post-Project Review/Lessons Learned
11	<b>Time-Boxed Iterations</b>	Phased Deliverables/Milestones
12	<b>Continuous Improvement</b>	Quality Assurance/Continuous Improvement (Kaizen)

# Project Management – the Agile way –Contd.

	Agile/Scrum/SAFe Term	Project Management Term
13	<b>Release Train Engineer (RTE)</b>	Program Manager
14	<b>Product Management</b>	Program Management
15	<b>System Architect/Engineering</b>	Technical Lead/System Architect
16	<b>Program Increment (PI) Planning</b>	Strategic Planning/Program Planning
17	<b>Inspect and Adapt (I&amp;A)</b>	Continuous Improvement/Quality Review
18	<b>System Demo</b>	Project Demonstration/Client Review