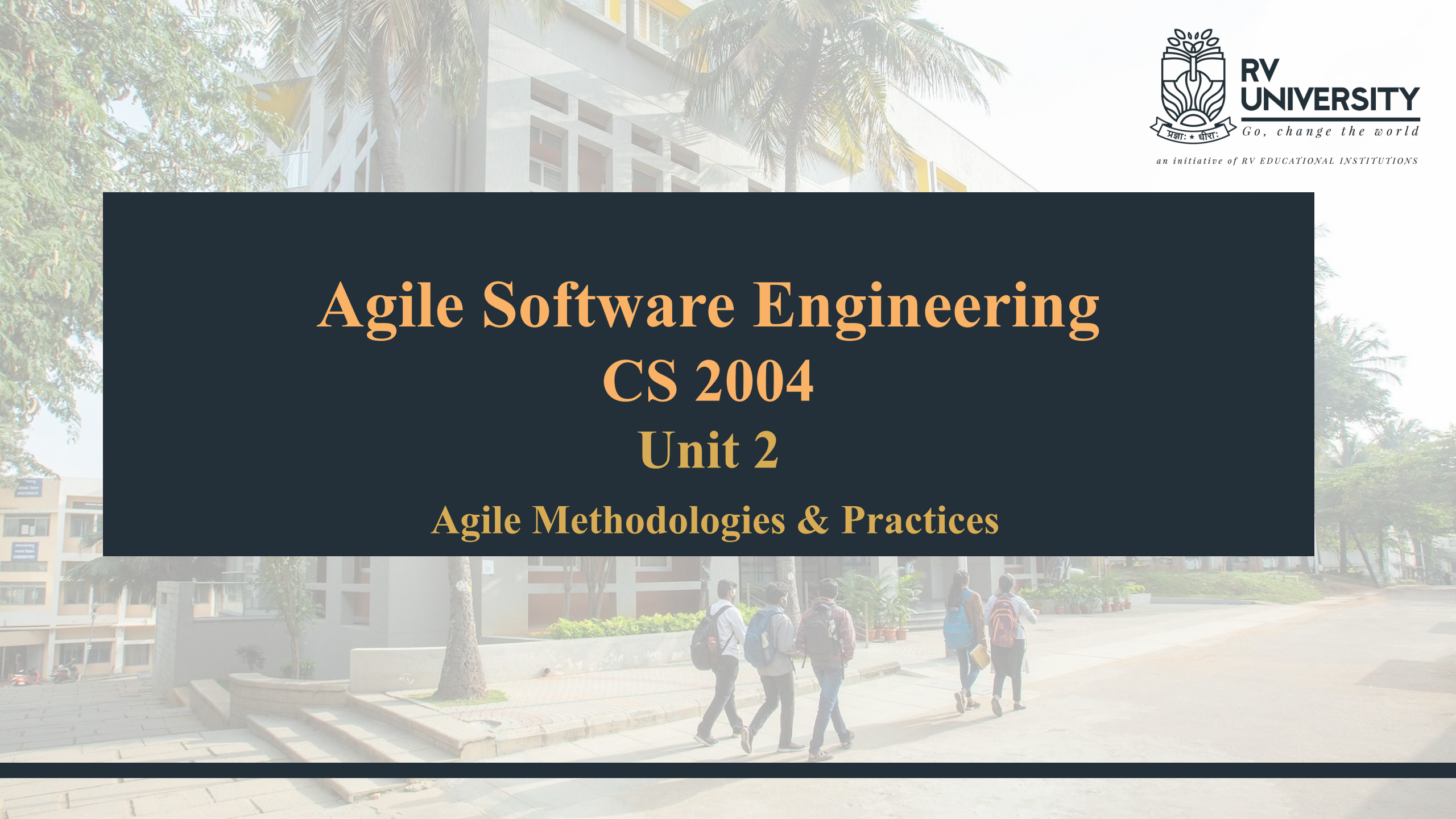
RV UNIVERSITY

Go, change the world

an initiative of RV EDUCATIONAL INSTITUTIONS

NEW-AGE GLOBAL UNIVERSITY FOR LIBERAL EDUCATION



RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

Agile Software Engineering
CS 2004
Unit 2
Agile Methodologies & Practices

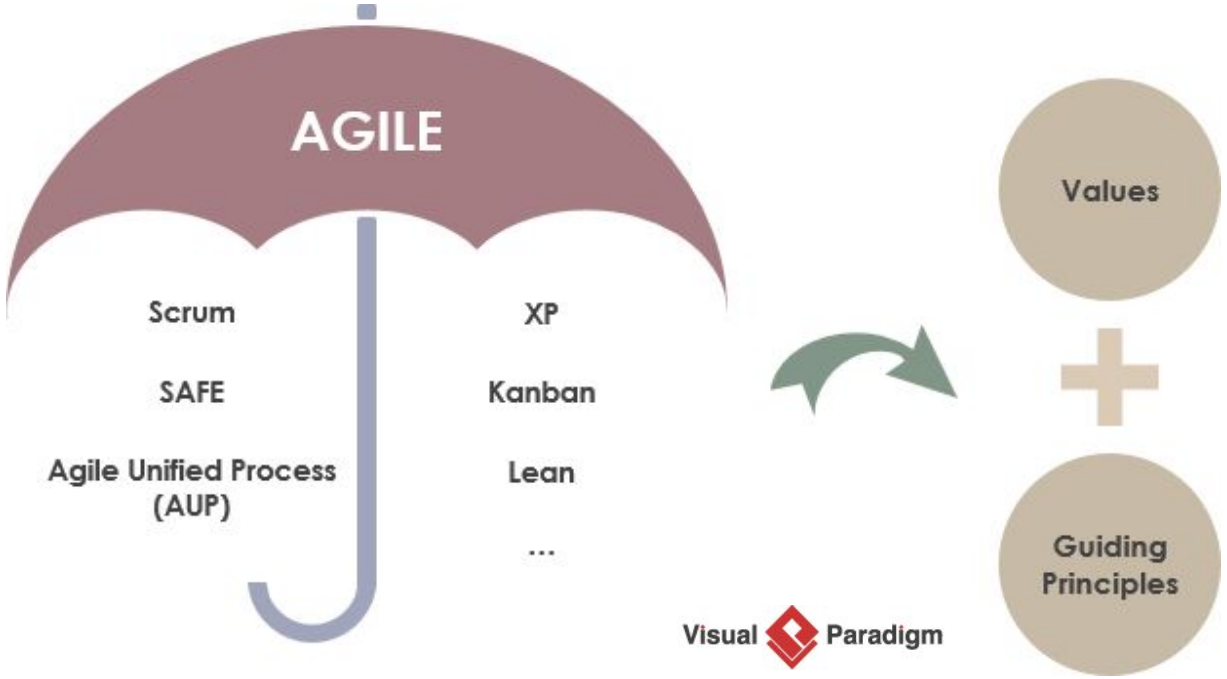
Agenda

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- Introduction to various Agile methodologies such as Scrum, XP, Lean, and Kanban
- Scrum: Scrum Roles – Product Owner, Scrum Master, Team, Release Manager, Project Manager, Product Manager, Architect, events, and artifacts
- Product Inception: Product vision, stakeholders, initial backlog creation;
- Agile Requirements – User personas, story mapping, user stories, 3Cs, INVEST Acceptance criteria, sprints, requirements, product backlog and backlog grooming
- Test First Development; Pair Programming and Code reviews; code coverage and tools

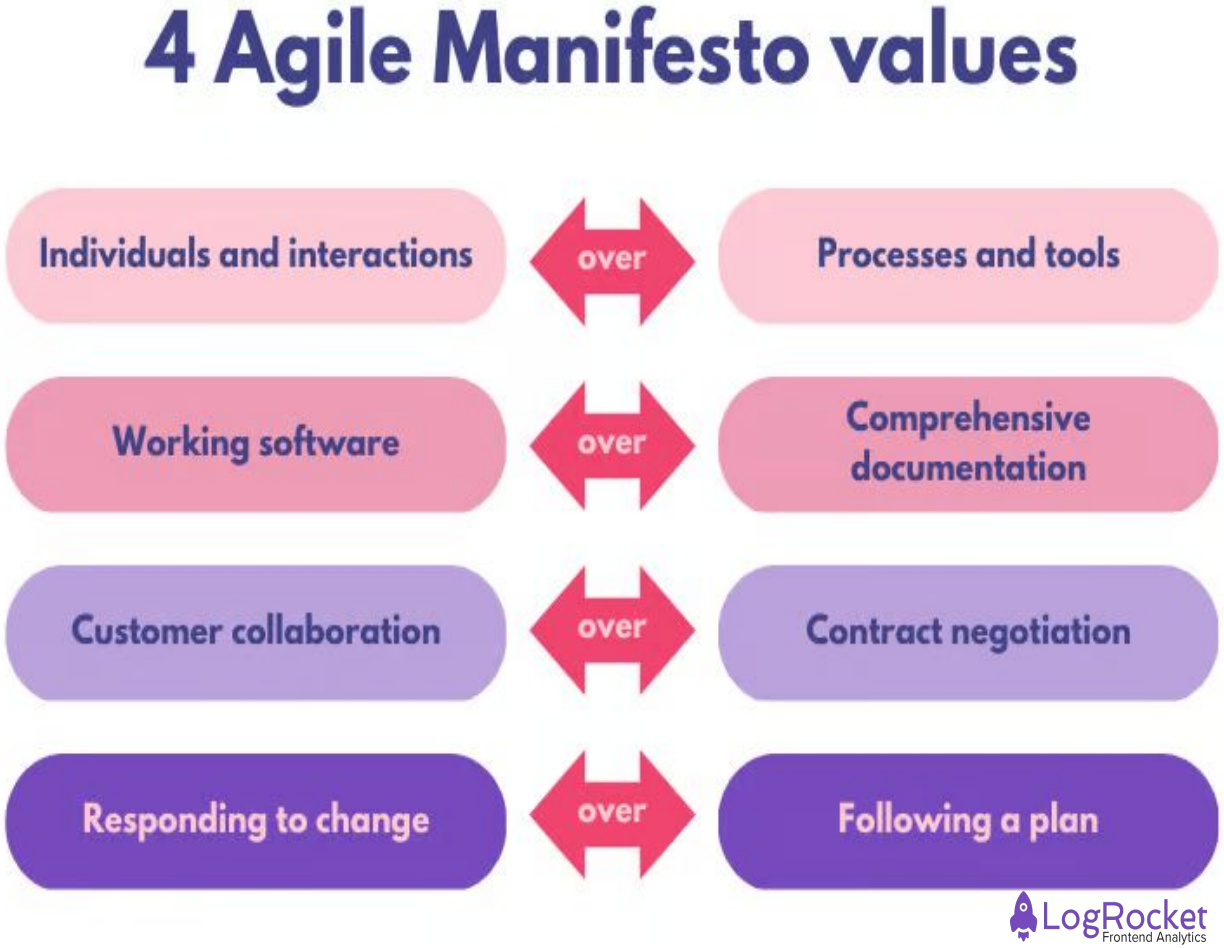
Various Agile Methodologies

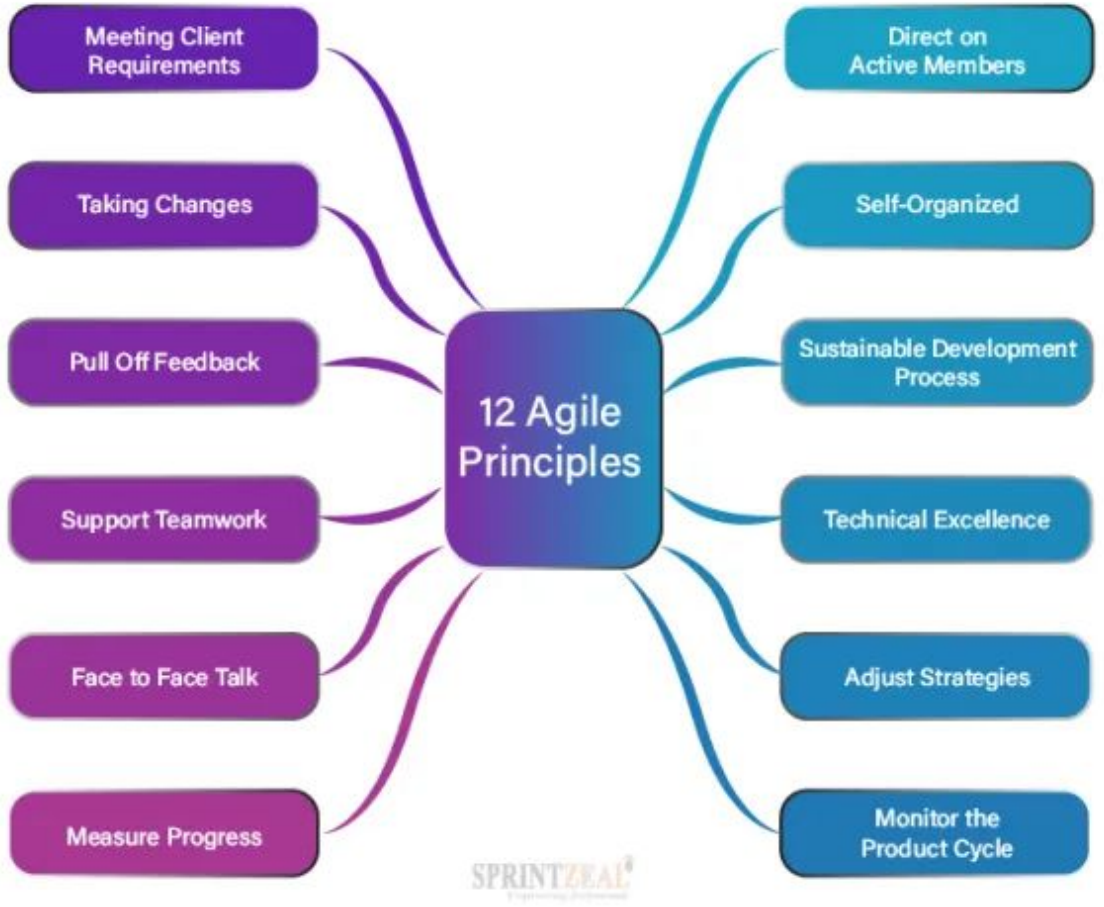
- Agile methodologies are frameworks and practices that embody the principles of the **Agile Manifesto**, focusing on iterative development, customer collaboration, and flexibility in managing changes.
- There are various Agile methodologies, each with its own approach to implementing these principles.
- Some of the key Agile methodologies are **Scrum**, **Extreme Programming (XP)**, **Lean**, and **Kanban**.



Agile : Recap

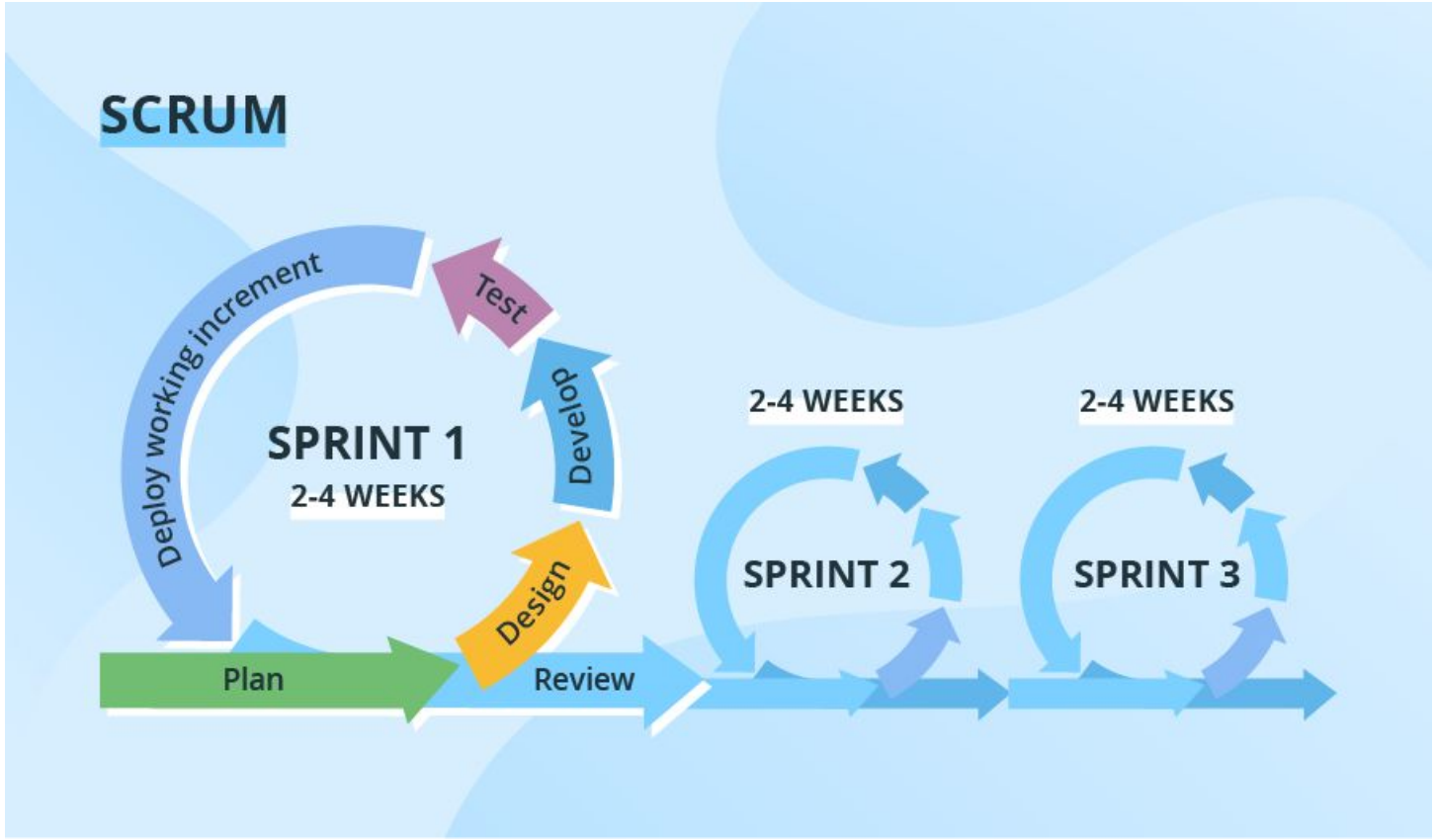
RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS



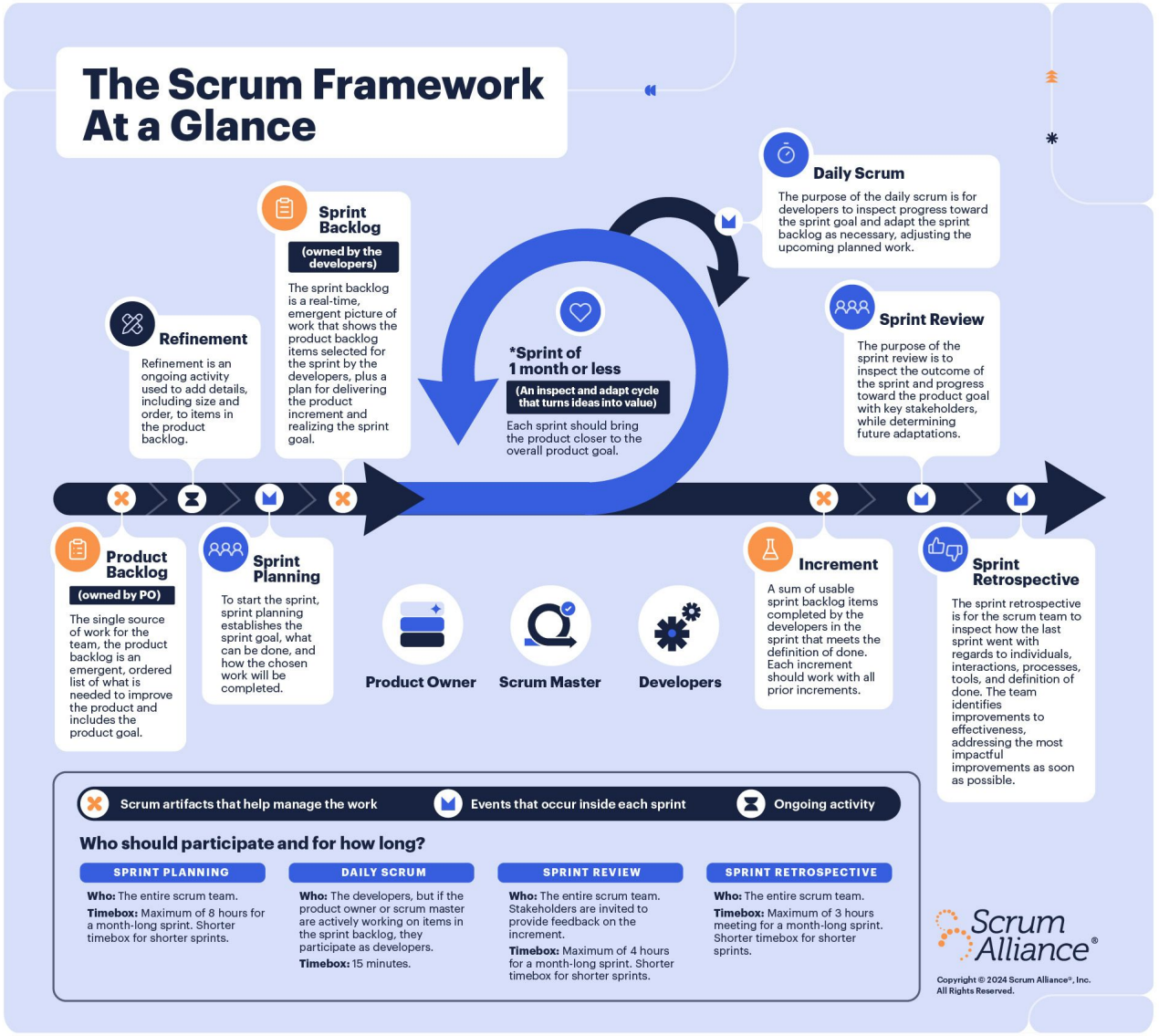


1. Scrum

- Scrum is a framework for organizing and managing work



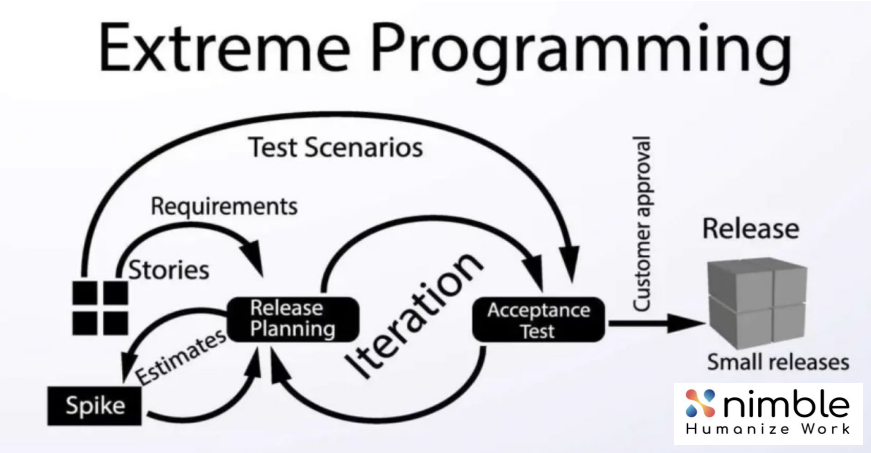
1. Scrum



RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

2. XP

- Another Agile methodology that focuses on improving software quality and responsiveness to changing customer requirements.
- XP emphasises technical excellence, collaboration, and frequent releases.



	Extreme Programming (XP)	Scrum
Method	Continuous development	Iterative development
Team	Equal roles, no hierarchy, everyone has shared ownership	Three key roles: Product Owner, Scrum Master, Development Team
Dev Team	Entire team is responsible for all aspects of the project	Team is responsible for delivering a potentially shippable product increment at the end of each sprint
Meetings	No formal structure	Daily stand-up meetings, sprint planning, sprint review, sprint retrospective, backlog refinement
Sprint	No fixed length	Fixed-length sprints, usually 2-4 weeks
Time-Box	No fixed time-boxing	Uses time-boxing to ensure that sprints are completed on time

Medium

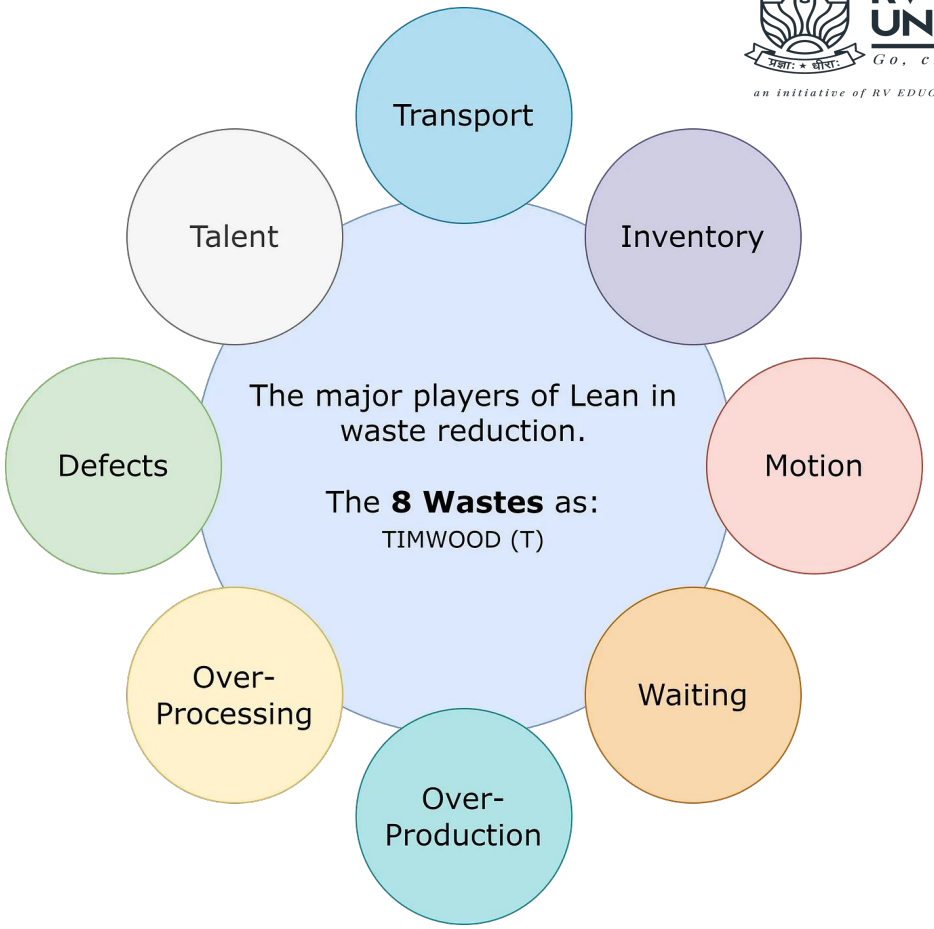
Key Elements of XP

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **Core practices**
  - **Test-Driven Development (TDD):** Writing automated unit tests before writing code to ensure correctness.
  - **Pair Programming:** Two developers work together at one workstation, with one writing code and the other reviewing it simultaneously.
  - **Continuous Integration:** Code is frequently integrated into the main branch, and automated tests are run to detect errors early.
  - **Refactoring:** Continuously improving and simplifying the codebase without altering its functionality.
  - **Simple Design:** Keeping the design as simple as possible to solve the current problem.
  - **Collective Code Ownership:** Everyone on the team is responsible for the entire codebase, reducing bottlenecks.
  - **Small Releases:** Delivering small, frequent releases of working software to get quick feedback from users
- **Communication with the customer:** XP encourages frequent collaboration with the customer to ensure the product meets their evolving needs.

3. Lean

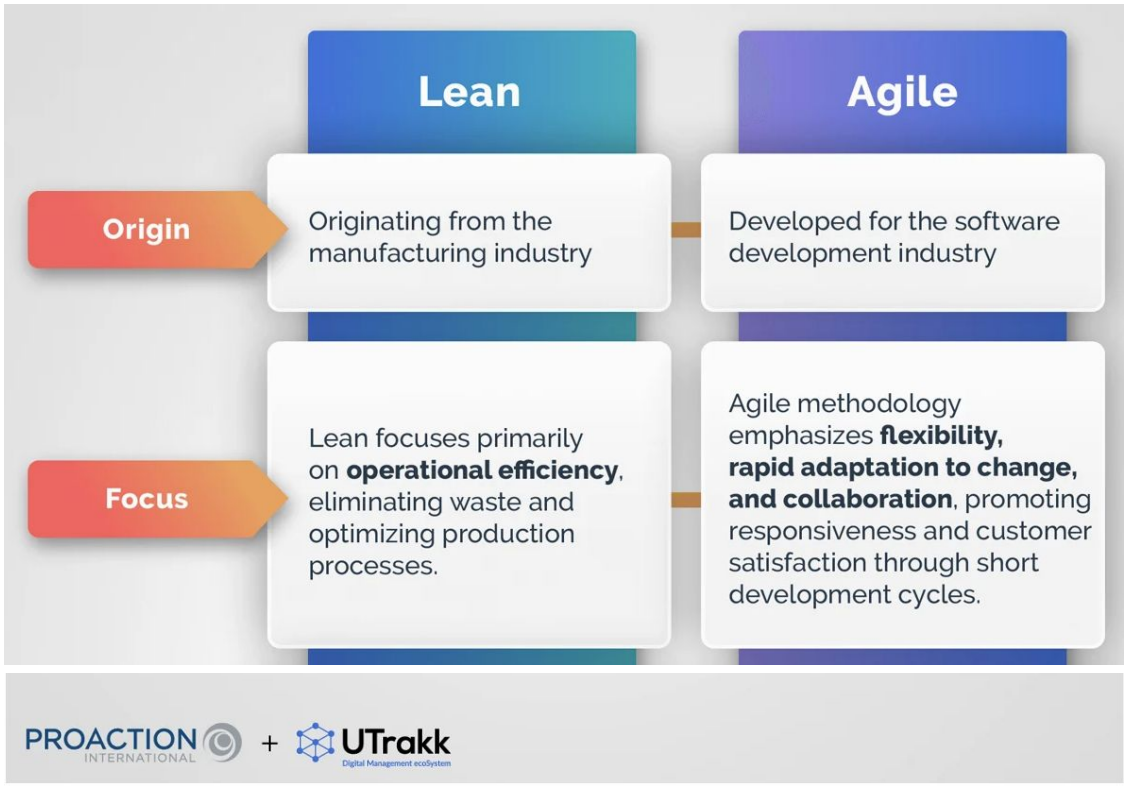
- Lean Software Development is an Agile methodology that derives its principles from **Lean manufacturing** (originally developed by Toyota).
- It focuses on **eliminating waste**, improving flow, and delivering value as quickly as possible.

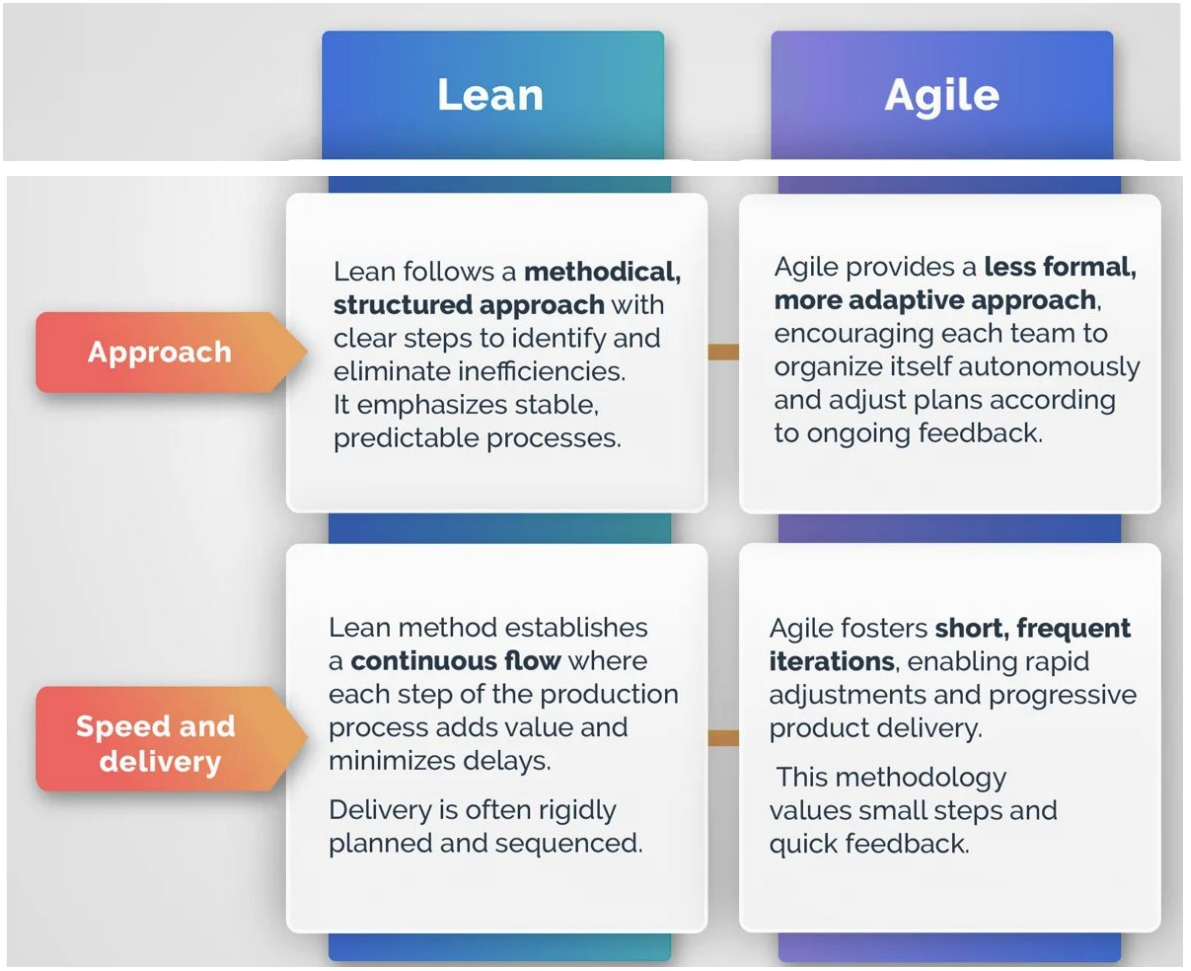


The Eight Wastes, abbreviated here as TIMWOOD(T). The last T is often described as the “hidden” waste. — Kevin Cooper on kevincooper.dev

Lean Vs Agile

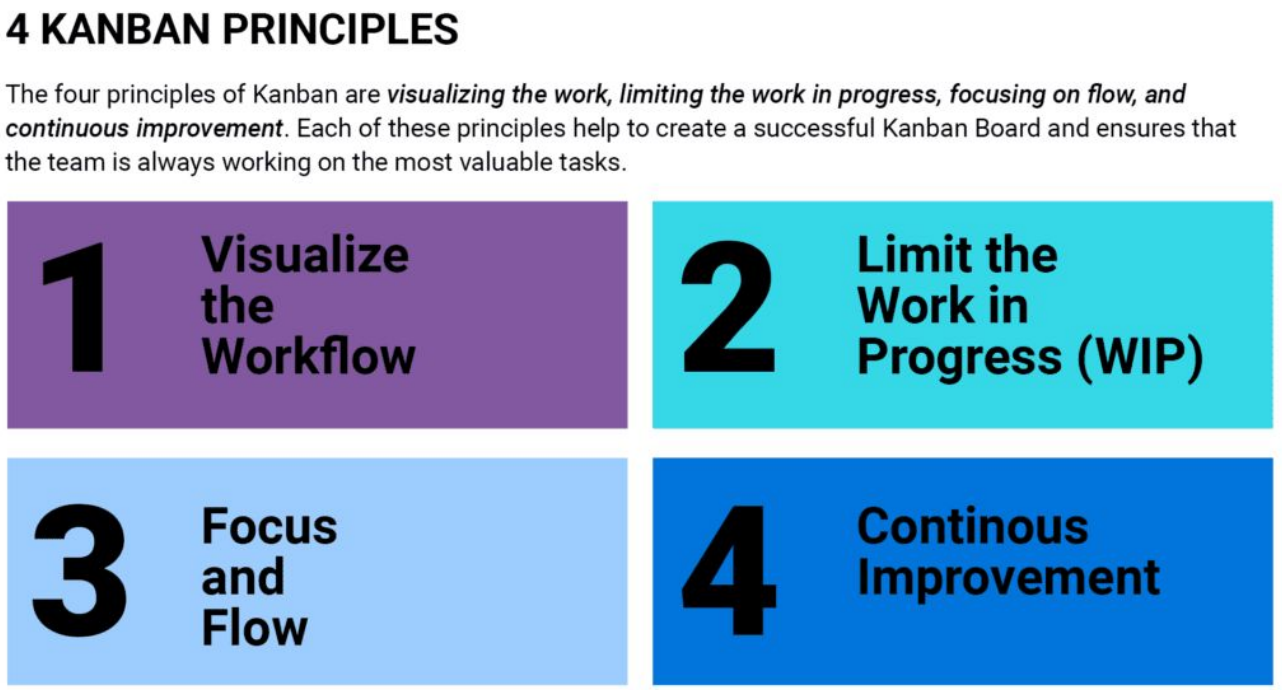
RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS





4. Kanban

- Kanban is an Agile methodology that focuses on **visualising work, limiting work in progress (WIP),** and continuously optimising the flow of tasks.
- It is especially popular for managing continuous flow projects, such as maintenance or support tasks, but can also be adapted for development teams



Kanban Vs Scrum (Contd.)

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

bmc

Scrum and Kanban

A comparison of Agile methodologies

	Scrum	Kanban
Origin	Software development	Lean manufacturing
Ideology	Solve complex problems while delivering valuable products	Use visuals to improve work flows and processes
Practices	Sprint planning Sprint Daily scrum Sprint review Sprint retrospective	Visualize the flow of work Limit work in progress Manage flow Make process policies explicit Implement feedback loops Improve, experiment
Roles	Product Owner Scrum Master Development Team	No formal roles
Metrics	Velocity	Cycle time Throughput

Kanban Vs Scrum (Contd.)

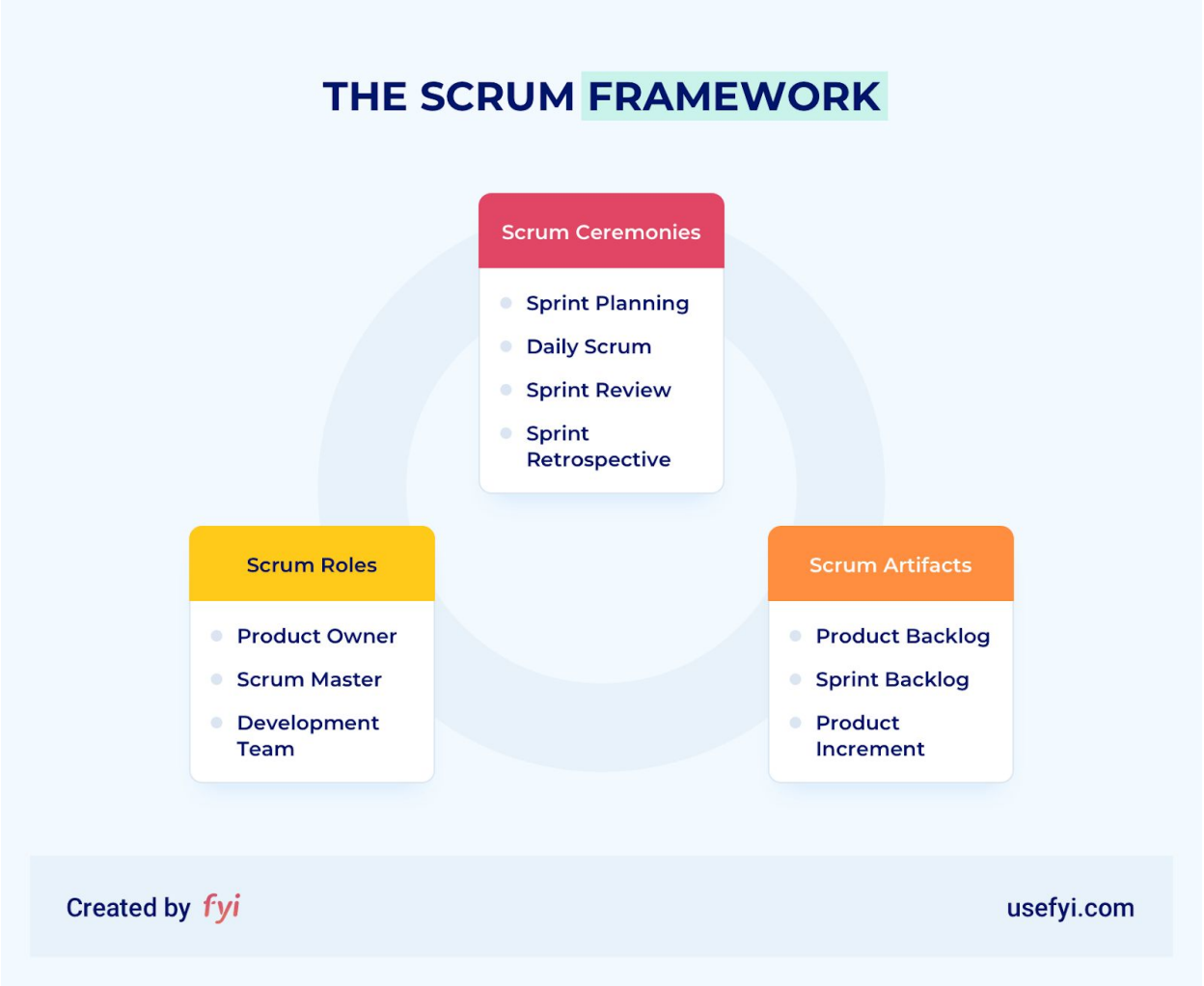
RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

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

K&C

1. Scrum

- Scrum is one of the most widely used Agile methodologies, especially for managing software development projects.
- It focuses on delivering small increments of the product through well-defined roles, events, and artifacts.



Scrum framework

Roles

- Product owner
- ScrumMaster
- Team

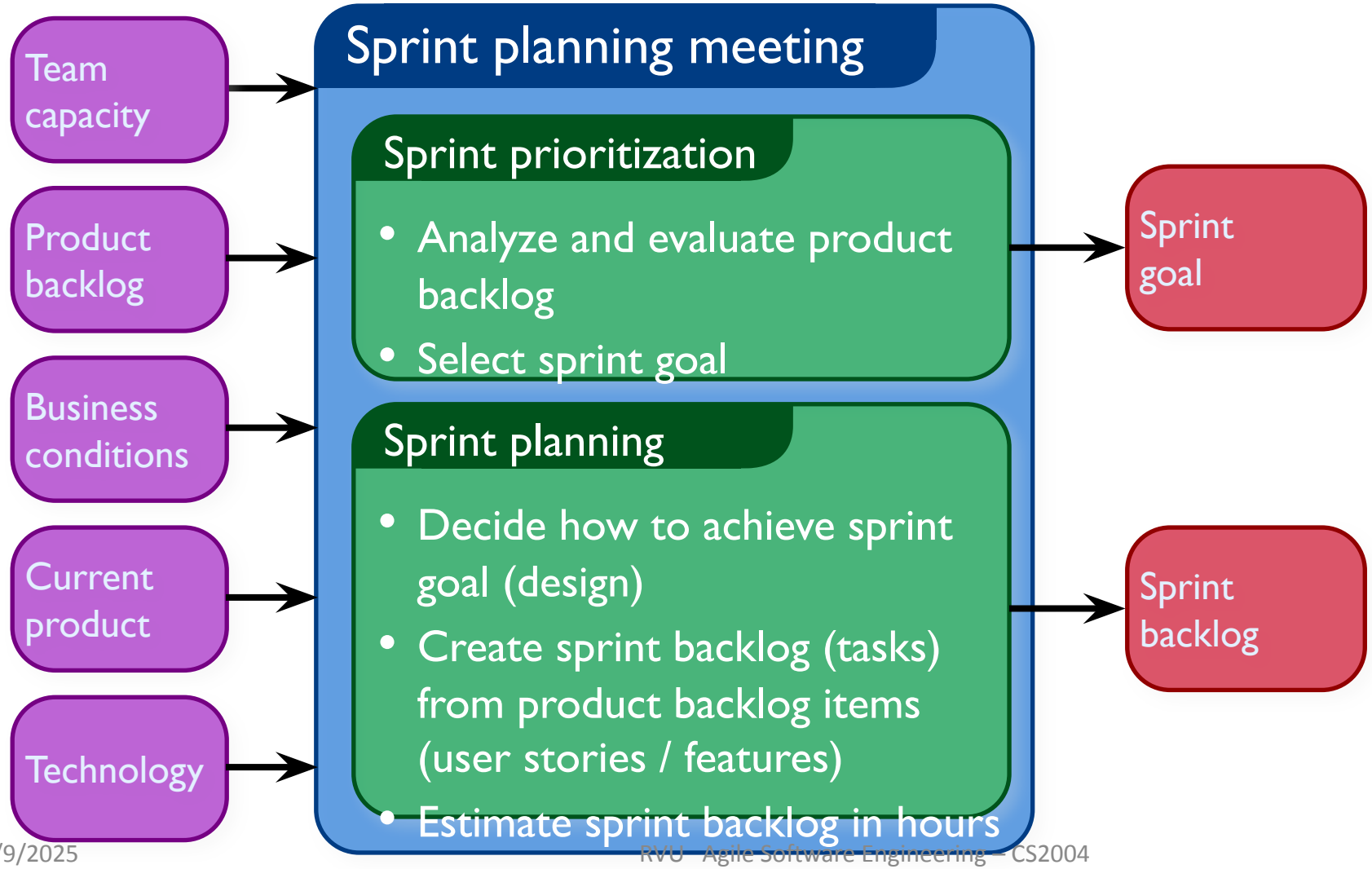
Ceremonies

- Sprint planning
- Sprint review
- Sprint retrospective

Artifacts

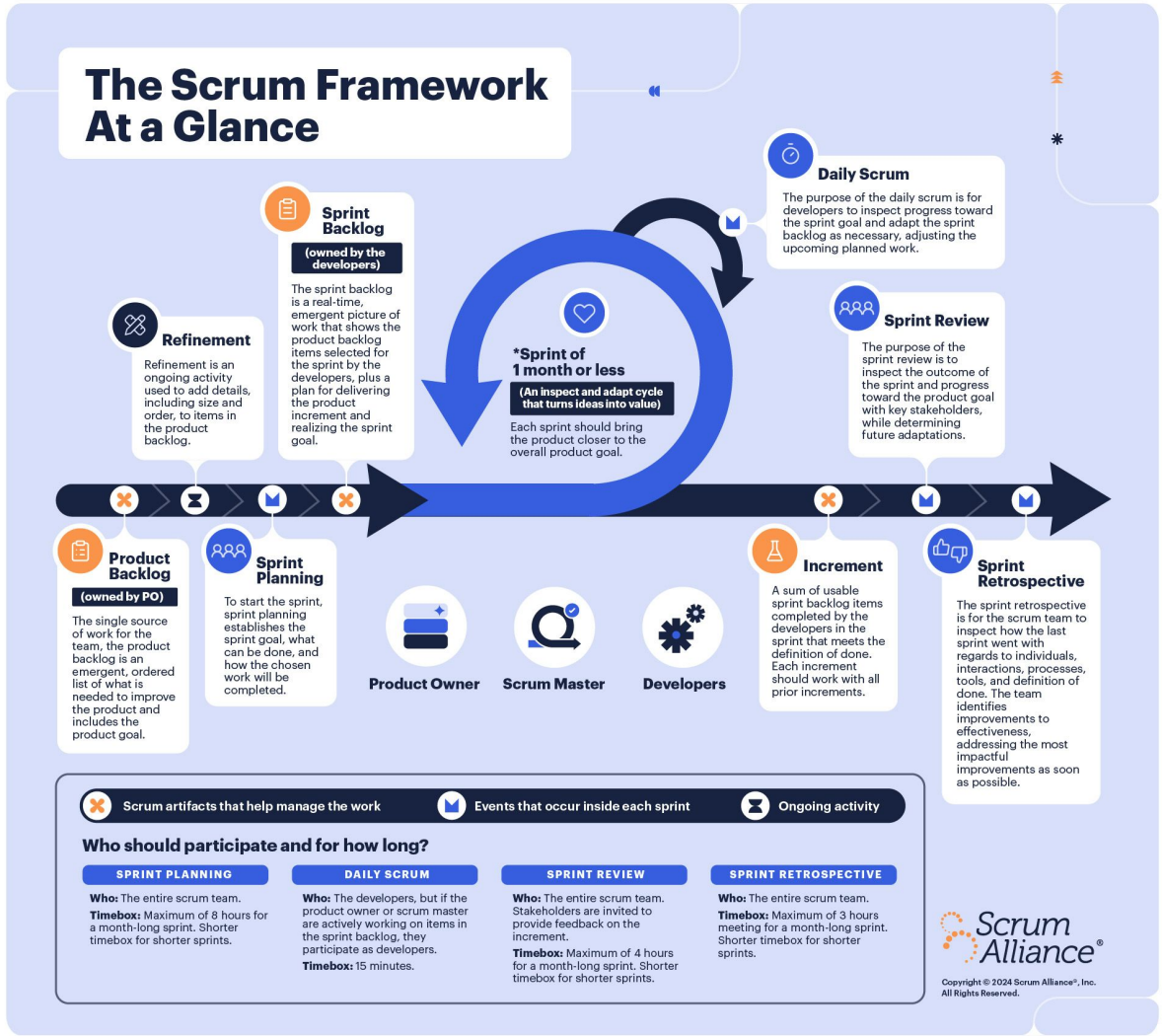
- Product backlog
- Sprint backlog
- Burndown charts

Scrum framework –contd.

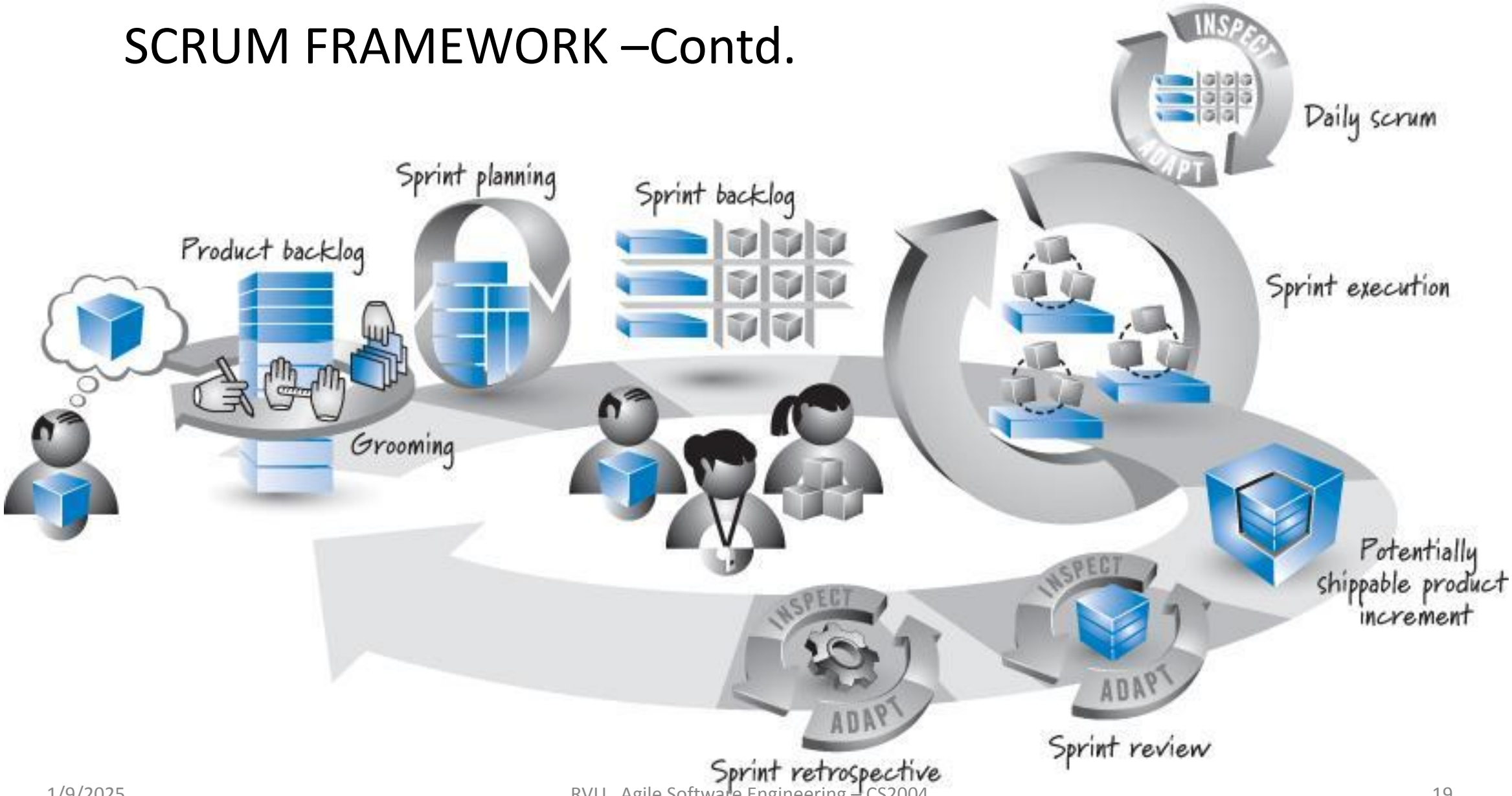


Scrum Framework

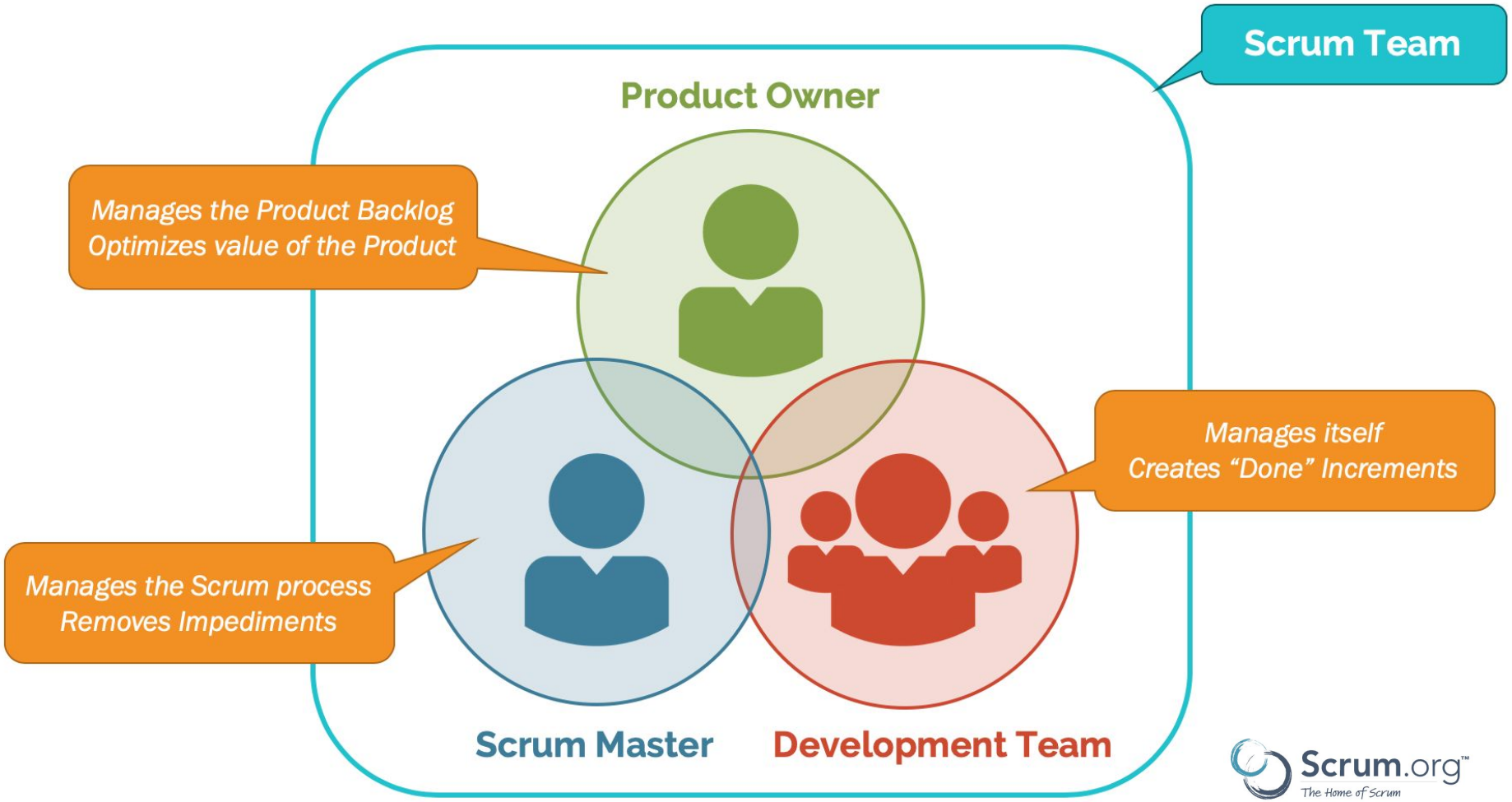
RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS



SCRUM FRAMEWORK –Contd.



Scrum roles


Scrum Team
Product Owner
Manages the Product Backlog
Optimizes value of the Product
Manages itself
Creates "Done" Increments
Manages the Scrum process
Removes Impediments
Scrum Master
Development Team

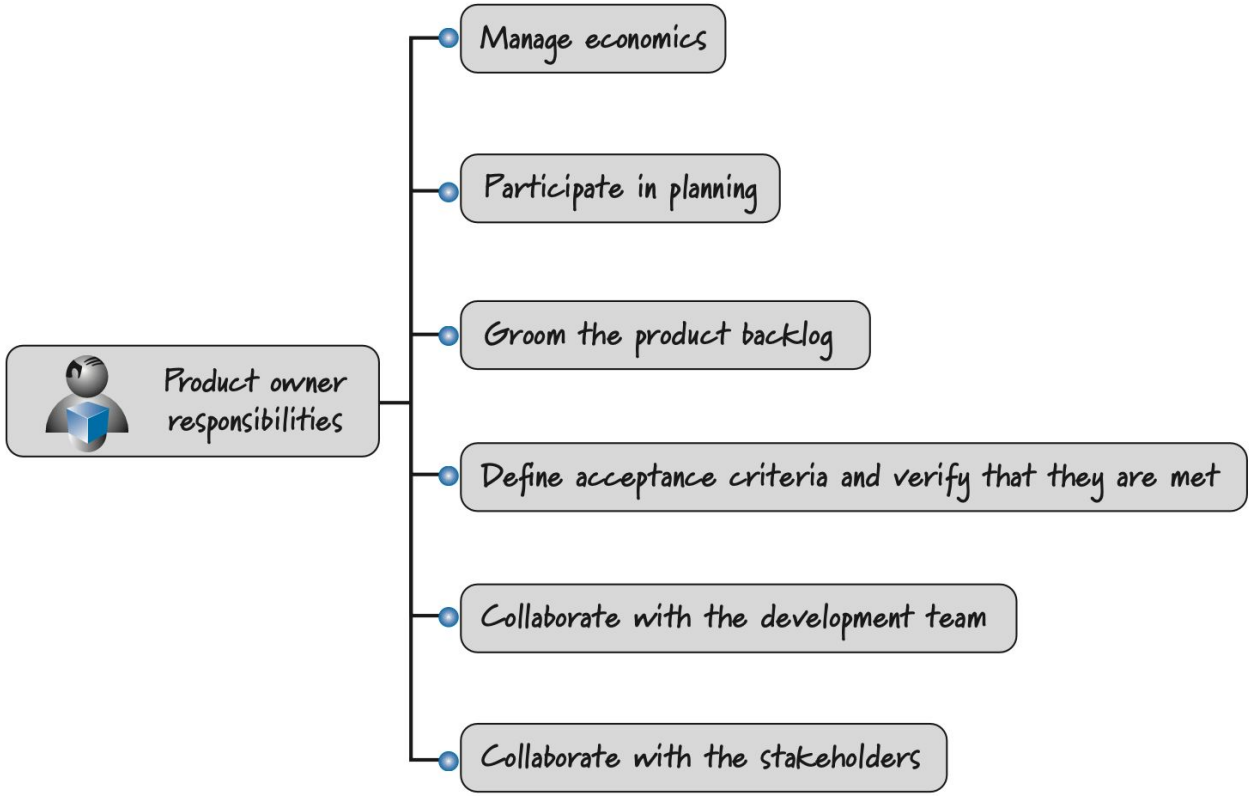
Scrum Role: Product Owner

Represents the stakeholders and customers.

Defines the product backlog (a prioritized list of features and tasks).

Ensures the development team works on the most valuable items first.

Regularly reviews and reprioritized backlog items based on customer feedback and business needs.

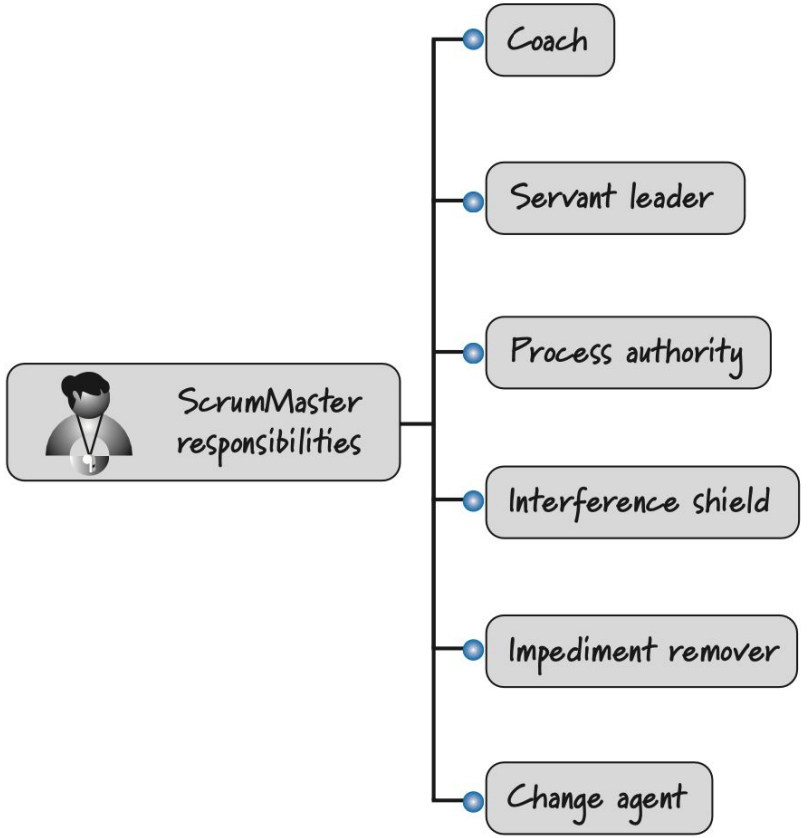


Scum Role: Scrum Master

Acts as a facilitator and coach for the Scrum team.

Ensures the team follows Scrum principles and removes any obstacles (impediments) that might hinder progress.

Helps the team stay focused and productive while protecting them from external distractions.



Principal ScrumMaster responsibilities

Scum Role: Team

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

A **cross-functional** group responsible for delivering a potentially shippable product increment at the end of each sprint.

The team is **self-organising**, meaning they decide how to accomplish the tasks and break down work.

The development team **typically consists of 5-9** members with skills across disciplines such as development, testing, and design.

Additional Role: Release Manager

- In a Scrum framework, the role of a Release Manager is not explicitly defined, but it can be crucial in larger organisations or complex projects. Some key responsibilities of a Release Manager are:

  Coordination and Planning: They coordinate with various teams (Development, QA, Operations) to plan and manage software releases. This includes creating and maintaining release schedules

- **Quality Assurance**: Ensuring that all releases meet quality standards before deployment. This involves overseeing testing processes and addressing any defects or issues.

  Risk Management: Identifying potential risks and issues that could impact the release and developing mitigation strategies.

  Communication: Acting as a liaison between different teams to ensure everyone is aligned on release goals and timelines.

  Process Improvement: Continuously improving release processes to enhance efficiency and reduce time to market.

What is role of Release Manager in Scrum? (1 of 2)

In larger organizations or projects, a dedicated Release Manager or Release Engineer role may exist

The responsibilities of such a role might include:

- **Release Planning:**
  - Collaborating with the Product Owner, Development Team, and other stakeholders to plan and coordinate releases based on business priorities.
- **Versioning and Tagging:**
  - Managing versioning and tagging of the software to ensure proper identification of release versions.
- **Release Documentation:**
  - Preparing and maintaining release documentation, including release notes and installation guides.

What is role of Release Manager in Scrum? (2 of 2)

- **Coordination:**
  - Coordinating with different teams (development, testing, operations) to ensure a smooth and timely release process.
- **Communication:**
  - Communicating release plans, status, and potential risks to stakeholders and the Scrum Team.
- **Continuous Improvement:**
  - Contributing to the continuous improvement of release processes and identifying areas for optimization.

It's important to note that the specific responsibilities of a Release Manager in Scrum can vary based on the organization's structure, size, and the nature of the project. In smaller, more agile environments, these responsibilities might be distributed among existing Scrum roles, while in larger organizations, a dedicated role may be defined.

Additional Role: Project Manager

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- In a Scrum framework, the role of a Project Manager is not explicitly defined, but it can be crucial in larger organisations or complex projects.

  Project Manager might still be involved in

  higher-level planning

  risk management, and

  coordination across multiple teams or projects.

  They might also assist with budgeting and scheduling, ensuring that the overall project aligns with organisational goals.

Additional Role: Product Manager

- In a Scrum framework, the role of a Product Manager is not explicitly defined, but they often work closely with the Product Owner and other team members to ensure the product's success. Here are some key responsibilities of a Product Manager in a Scrum environment:

  Product Vision : The Product Manager helps define and communicate the product vision, ensuring that it aligns with the overall business goals and customer needs.

  Backlog Management: They assist in managing the product backlog, prioritising features based on business value, user feedback, and market demands.

  Stakeholder Communication: They act as a bridge between the Scrum team and external stakeholders, ensuring that everyone is aligned on the product goals and progress.

  Sprint Planning and Review: The Product Manager participates in sprint planning and review meetings to provide input on priorities and to ensure that the team is focused on delivering value.

  Continuous Improvement: They focus on continuous improvement by gathering feedback, analysing performance metrics, and making necessary adjustments to the product strategy

Comparison of roles of Managers

Role	Focus	Key Responsibilities in Scrum
Release Manager	**Coordination of software releases to production.**	- Manages the release cycle. - Plans and schedules releases. - Ensures compliance and stability of releases.
Project Manager	**Facilitating coordination and reporting across teams and stakeholders.**	- Manages cross-team collaboration. - Oversees timelines and resources. - Reports on progress and risks. - Coordinates with non-Scrum teams.
Product Manager	**Defines the long-term product vision and strategic goals.**	- Creates and maintains product roadmap. - Communicates product vision. - Aligns multiple Scrum teams' work. - Interacts with stakeholders and customers.

Additional Role: Architect

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- In a Scrum team, the role of an Architect, while not formally defined, is crucial for ensuring the technical integrity and long-term vision of the product. Here are some key responsibilities of an Architect in a Scrum environment:

  Defining the Architecture: The Architect is responsible for creating and maintaining the overall structure of the system, ensuring it meets both functional and non-functional requirements.

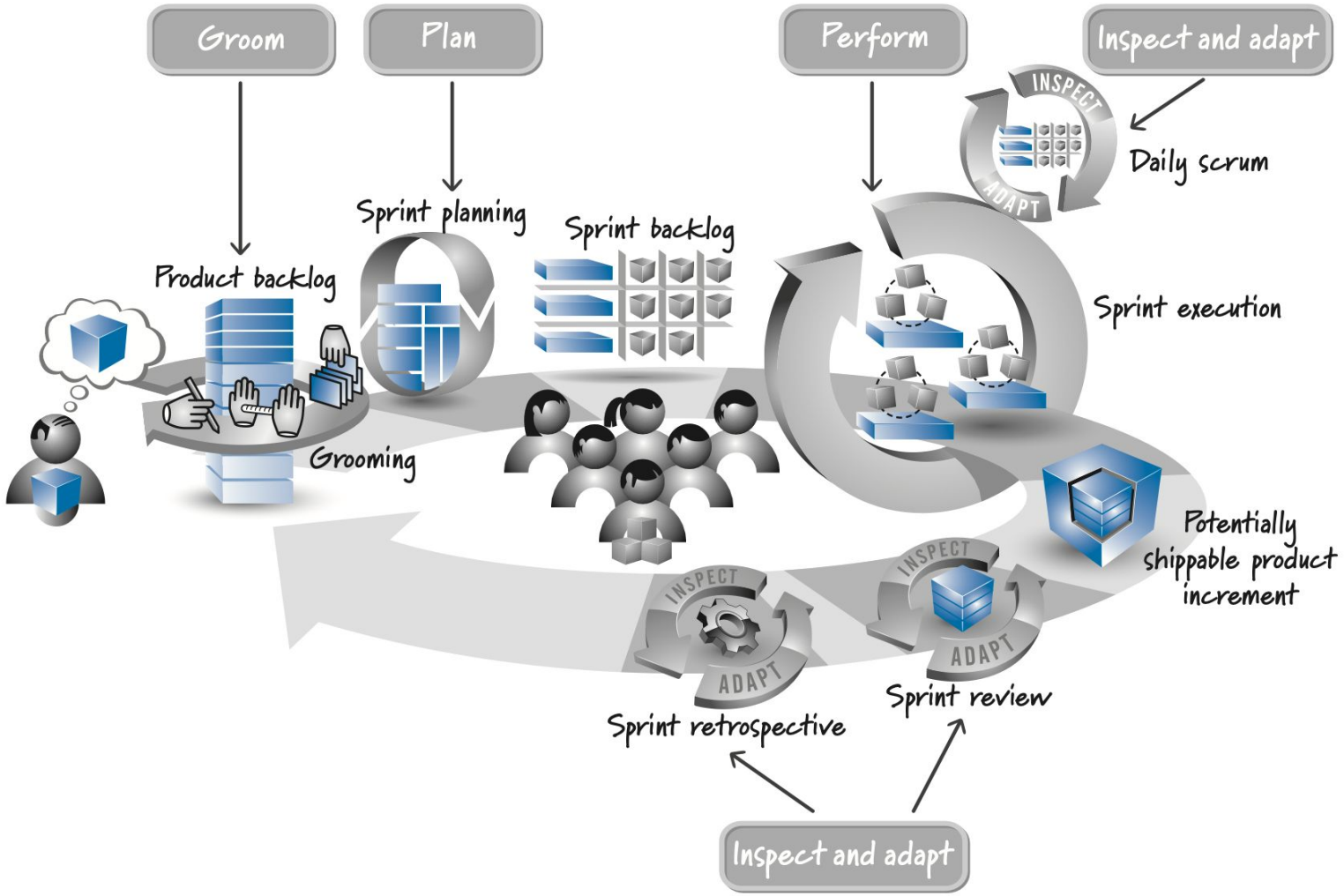
  Technical Leadership: They provide technical guidance and mentorship to the development team, helping to resolve complex technical issues and making sure the team adheres to best practices.

  Collaboration: Architects work closely with the Product Owner to understand business needs and translate them into technical solutions. They also ensure that the architecture aligns with the product vision and goals.

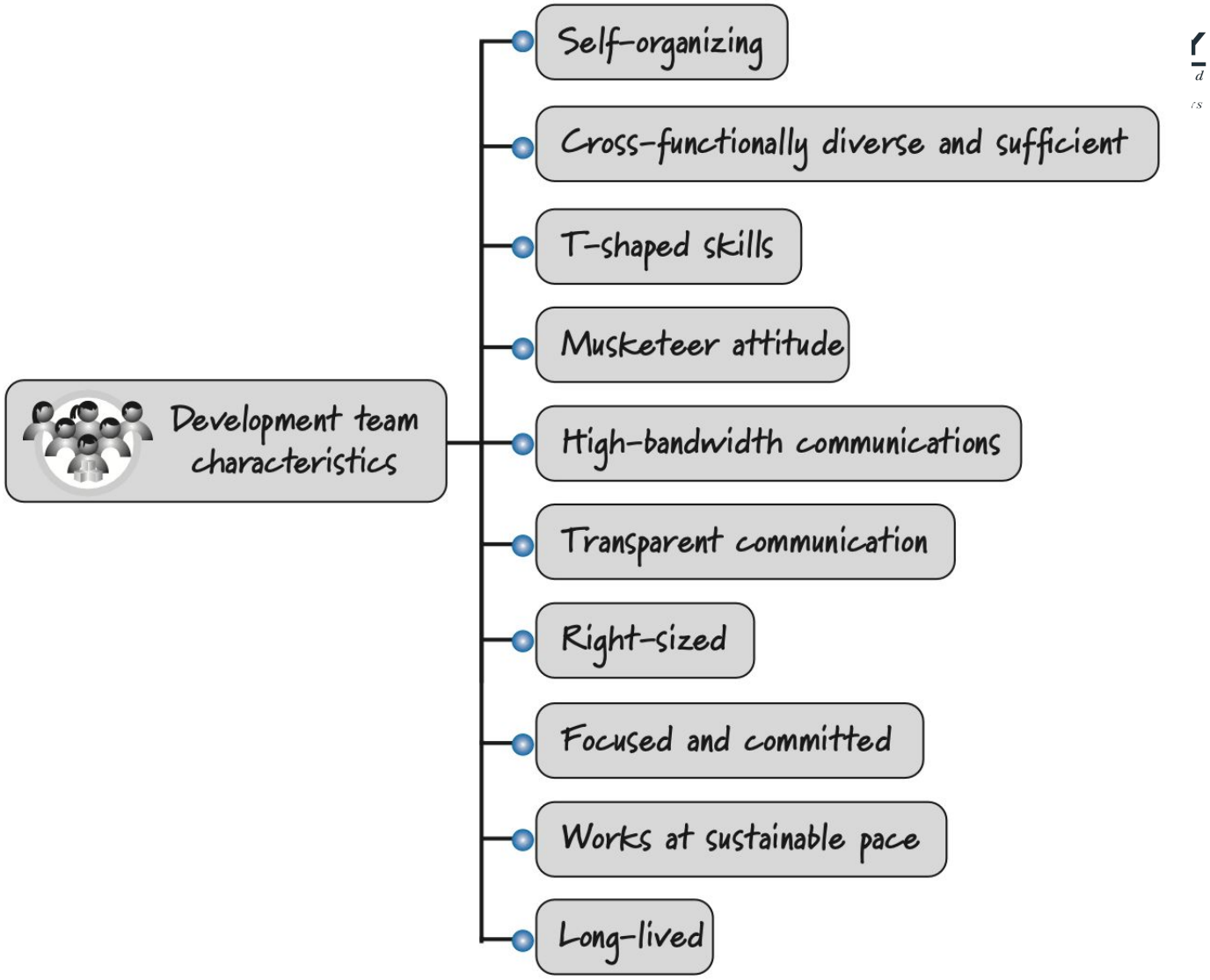
  Ensuring Quality: They oversee the implementation to ensure that the system is built according to the architectural guidelines and standards. This includes code reviews and ensuring that the system is scalable, maintainable, and secure.

  Continuous Improvement: They continuously evaluate and improve the architecture to adapt to changing requirements and technologies.

Team Responsibilities



Team Characteristics



The daily scrum

- Parameters
  - Daily
  - 15-minutes
  - Stand-up
- Not for problem solving
  - Whole world is invited
  - Only team members, ScrumMaster, product owner, can talk
- Helps avoid other unnecessary meetings

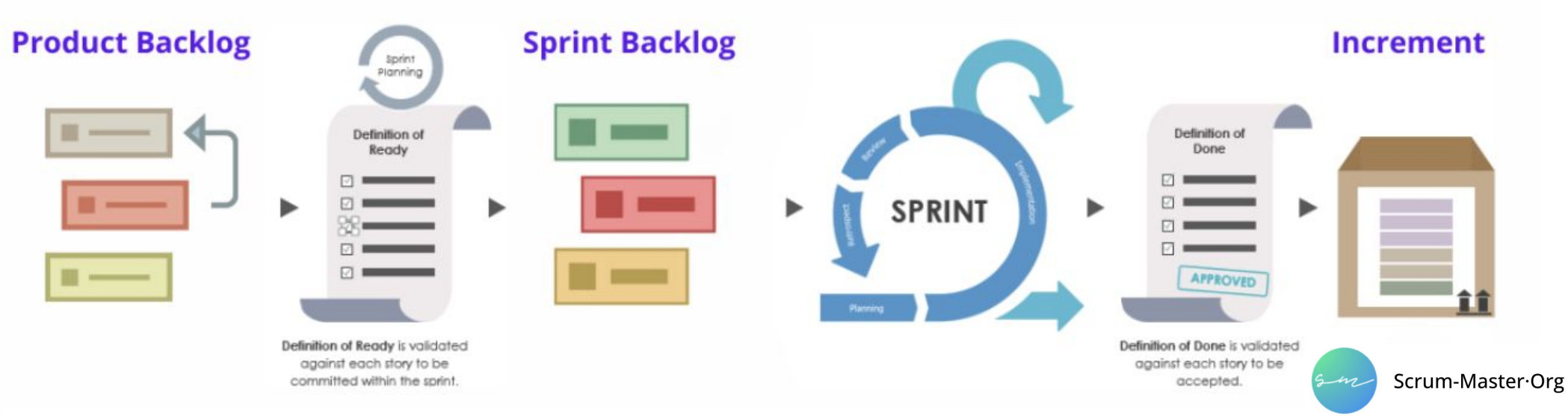
Everyone answers 3 questions

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

1. What did you do yesterday?
2. What will you do today?
3. Is anything in your way?

- These are *not* status for the ScrumMaster
  - They are commitments in front of peers

Scrum artifacts



- **Product Backlog:** A prioritised list of tasks, features, or changes required in the project.
- **Sprint Backlog:** A set of tasks selected from the product backlog for a specific sprint.
- **Increment:** A potentially shippable product that is delivered at the end of a sprint.

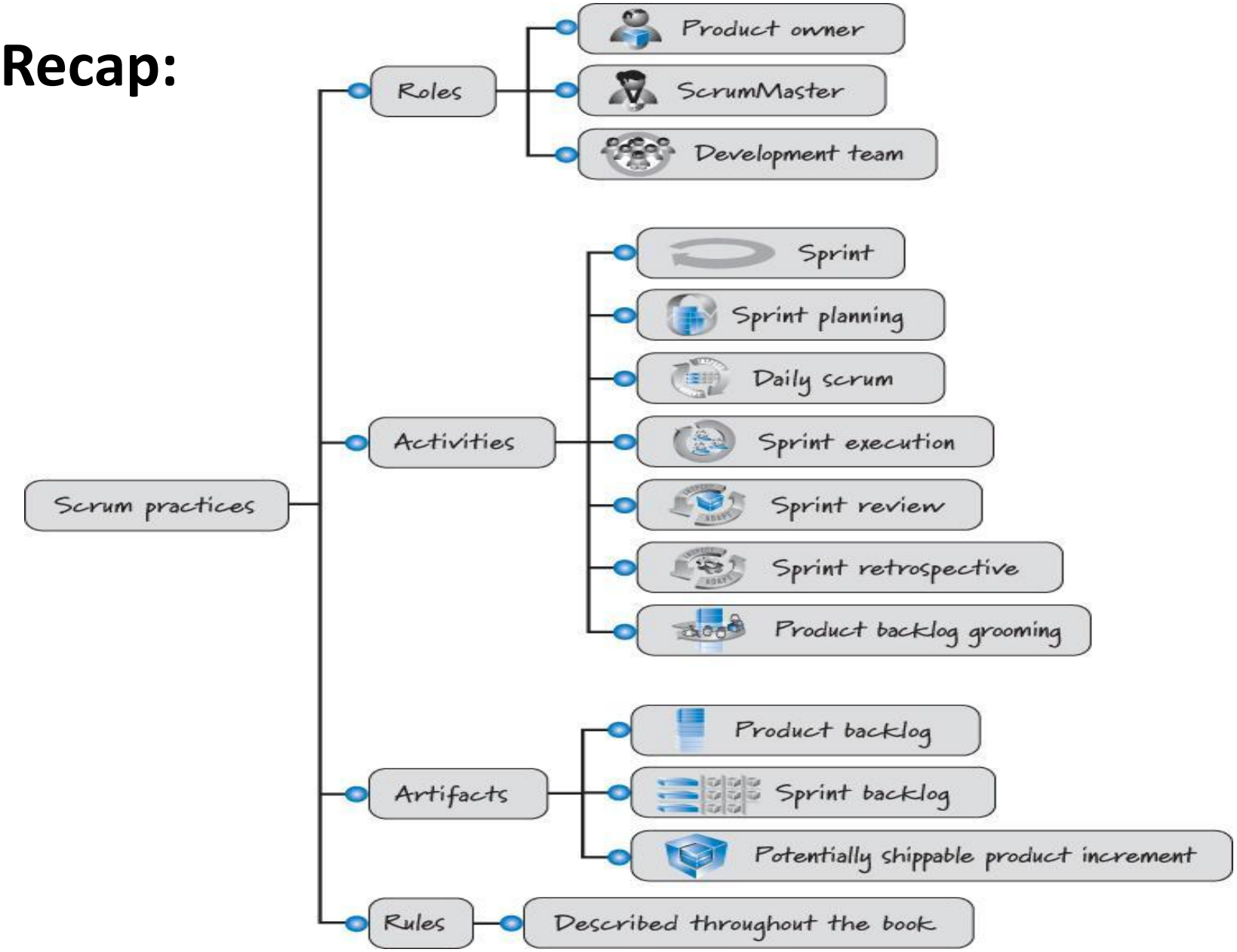
Scrum Events (Ceremonies)

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

Event:	Purpose:	Timebox:	Frequency:	How this reduces meetings
Sprint	Deliver a done, usable increment which meets the Sprint goal	1 month or less	N/A	Limits the time horizon for delivery to one month
Sprint Planning	Plan the work for the upcoming Sprint	Eight hours. For shorter Sprints, the event is usually shorter.	Once at the beginning of the Sprint	Planning together saves time
Daily Scrum	Increase the likelihood of delivering a done, usable increment which meets the Sprint goal	15 minutes	Daily	Improved communication, quick decision-making and early identification of impediments
Sprint Review	Discuss what was done during the Sprint and to collaborate on what to do next	4 hours. For shorter Sprints, the event is usually shorter.	Once at the end of the Sprint	Feedback from Stakeholders frequently allows the team to adjust direction early
Sprint Retrospective	Improve the Scrum team's effectiveness	3 hours. For shorter Sprints, the event is usually shorter.	Once at the end of the Sprint	Opportunity to actively brainstorm ways to improve processes and interactions

Scrum-Master·Org

Recap:


Scrum practices
Roles
Product owner
ScrumMaster
Development team
Activities
Sprint
Sprint planning
Daily scrum
Sprint execution
Sprint review
Sprint retrospective
Product backlog grooming
Artifacts
Product backlog
Sprint backlog
Potentially shippable product increment
Rules
Described throughout the book

Product Inception

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- Product Inception is a crucial phase that sets the foundation for a successful project. It involves a series of collaborative activities aimed at defining the product vision, goals, and roadmap before any development work begins. Here are the key aspects of Product Inception:

  Defining the Vision: Establishing a clear and shared understanding of what the product aims to achieve and why it is being developed.

  Stakeholder Alignment: Bringing together all relevant stakeholders, including the development team, product owner and other key participants, to ensure everyone is on the same page regarding the product objectives and priorities.

  Identifying Requirements: Gathering and prioritizing the initial set of requirements, often through user stories, to create a product backlog that guides the development process.

- Risk and Constraint Analysis: Identifying potential risks, constraints, and challenges that could impact the project, and developing strategies to mitigate them.

  Creating a Roadmap: Developing a high-level plan that outlines the major milestones, deliverables, and timelines for the project.

  User Persona and Journey Mapping: Understanding the target users, their needs, and how they will interact with the product through persona analysis and user journey mapping.

Product Vision

- In Scrum, the **Product Vision** is a concise statement that defines the long-term goal and purpose of the product. It serves as a guiding star for the Scrum team and stakeholders, providing a clear direction and inspiring everyone involved. Here are the key aspects of a Product Vision:

  Purpose and Value: It describes the purpose of the product and the value it aims to deliver to customers and users.

  Feature State: The vision outlines a feature state of the product, highlighting what problems it aims to solve or what ambitions it seeks to fulfill.

  Inspiration and Motivation: A well-crafted Product Vision motivates and inspires the Scrum team, stakeholders, and users by connecting with them emotionally and practically.

- Guidance for Decision-Making: It helps the Product Owner and the team make informed decisions about what features to build and prioritise.

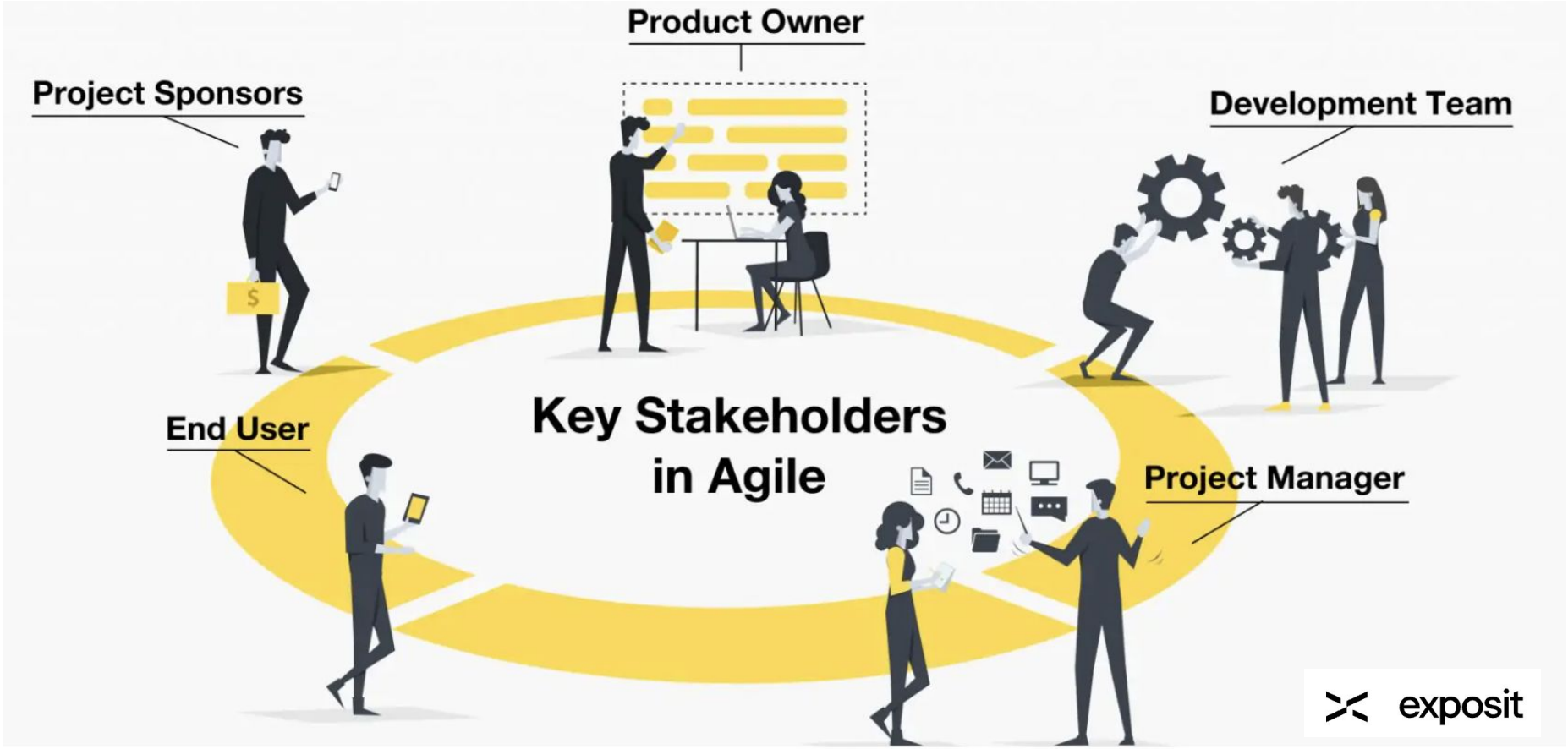
  Creating a Roadmap: Developing a high-leveplan that outlines the major milestones, deliverables, and timelines for the project.

  Collaboration: The Product Vision is typically created by the Product Owner, often with input from stakeholders and the Scrum team, ensuring a shared understanding and alignment.

Product Vision Statement Examples

- Empower teams worldwide to collaborate effortlessly, increase productivity, and deliver exceptional results by providing the most intuitive and comprehensive project management platform available, making work both efficient and enjoyable.
- Our e-learning platform transforms the way the world learns by providing a personalised, accessible, and engaging learning experience for learners of all ages and backgrounds, empowering them to unlock their full potential and shape a brighter future.

Stakeholders

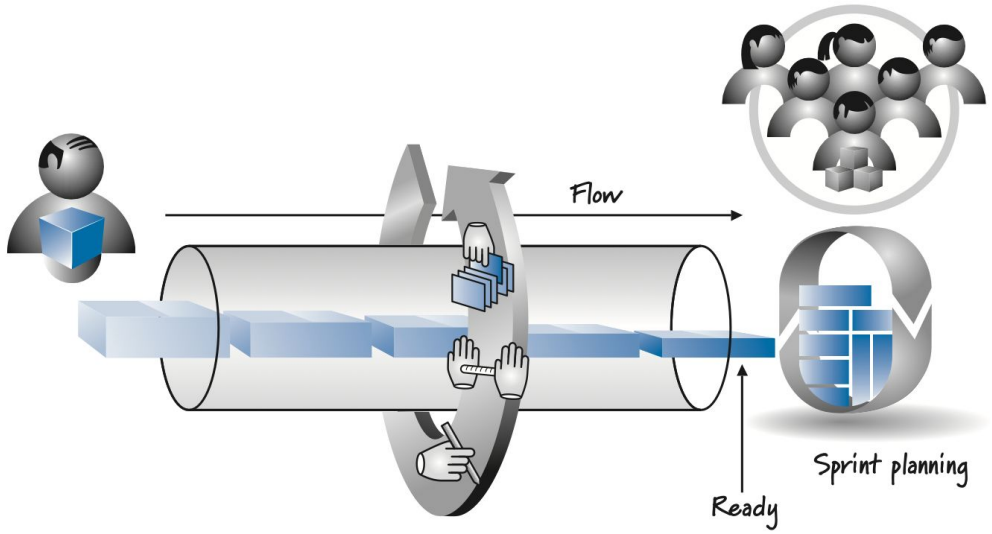


Sprint

- "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

Product Backlog

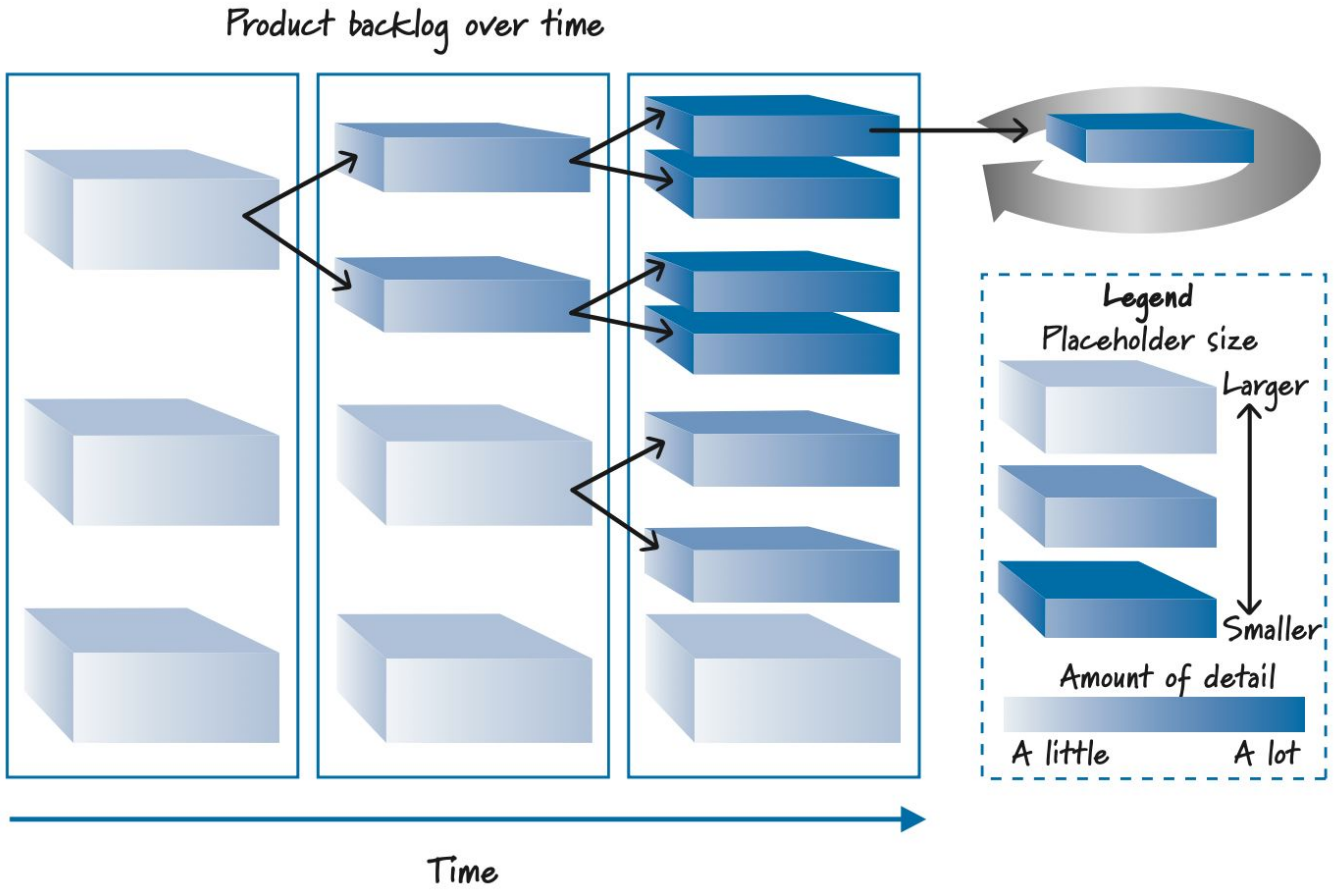
- In Agile methodology, a **Product Backlog** is a dynamic, prioritised list of features, improvements, and bug fixes that need to be addressed in a product.
- It serves as a central repository for all the work that needs to be done



Product backlog as a pipeline of requirements

PBI as placeholders for requirements

Progressive refinement of requirements over time



Requirements : Sequential Vs Scrum

Why all requirements can’t be at the same level of detail at the same time

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

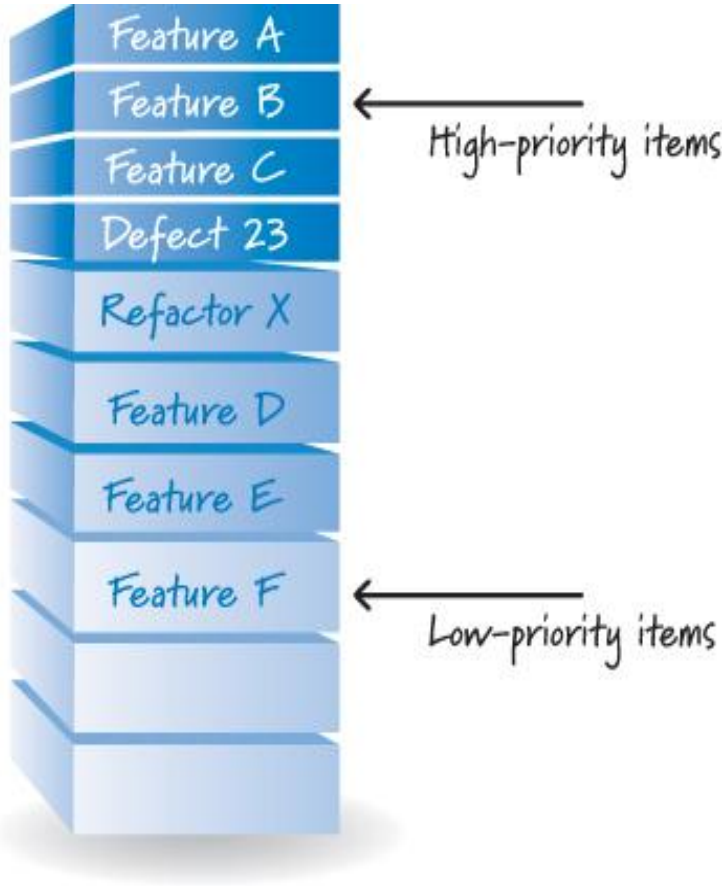
- reduces flexibility,
- delays feedback,
- makes it harder to adapt to changes, and
- often results in over-specification or incorrect prioritization of features.
- these issues can lead to wasted effort,
- increased costs, and
- ultimately, a product that doesn't meet user needs or market demands.

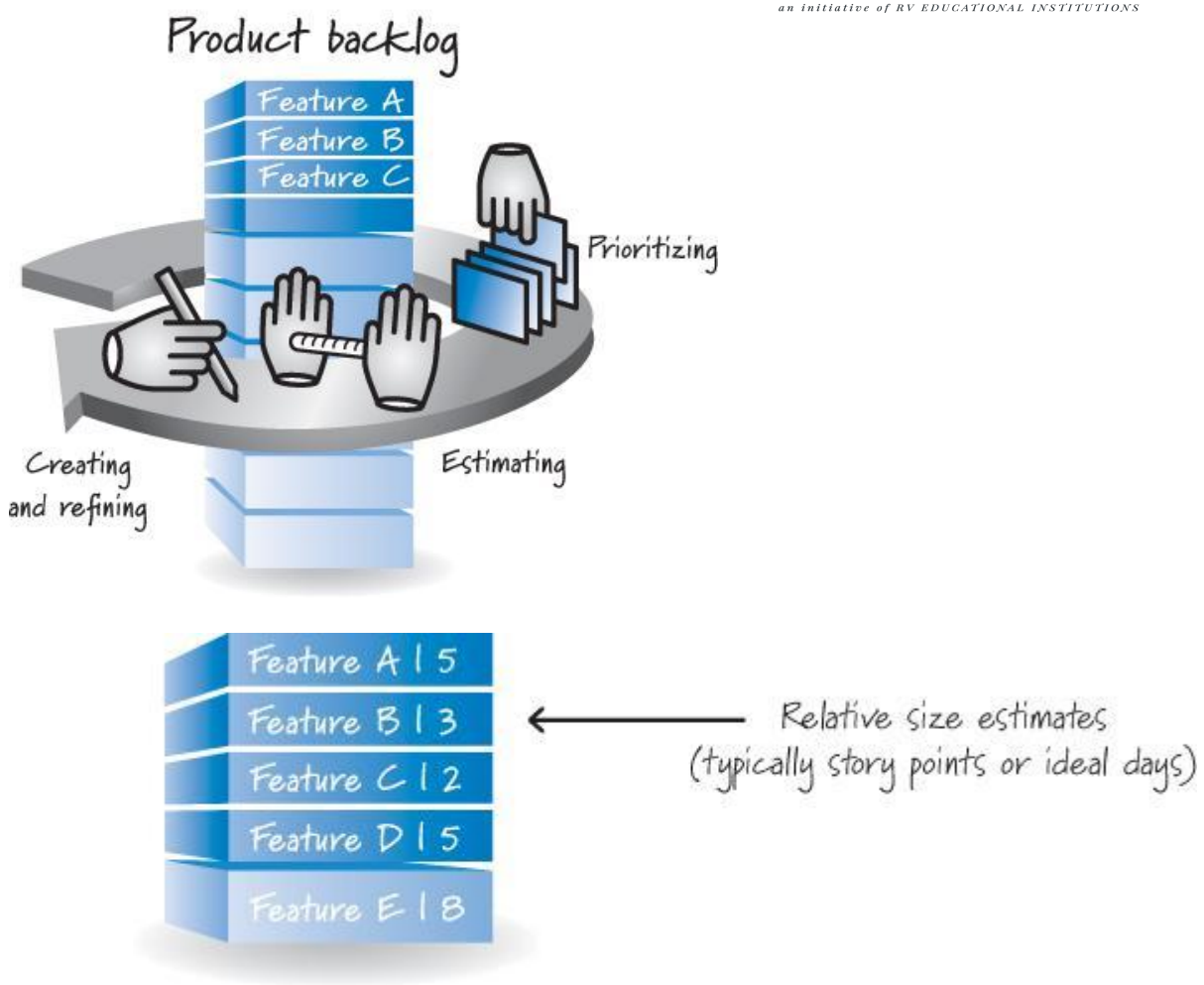
Requirements : Sequential Vs Scrum

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **Sequential** :
  - Requirements are non-negotiable
  - Detailed up-front
  - Meant to stand-alone
- Scrum :
  - Details negotiated through conversations
  - Act as an important degree of freedom that we can manipulate to meet our business goals.

Product Backlog Grooming





User personas

A **user persona** is a fictional character that represents a typical user or customer of the product. It's used to help Agile teams understand the end user's behavior, needs, goals, and challenges.

- **Role in agile**
  - **Understanding the user**: Personas provide a clear image of who the users are, allowing teams to develop solutions tailored to specific user types.
  - **Empathy**: Teams can better empathize with users and prioritize features or functionalities that address their needs.
  - **Guiding development**: User personas serve as a constant reminder to the team about for whom they are building the product, guiding decisions throughout the project.

Example of User persona

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **E-commerce Website Persona**

- **Name:** Sarah Johnson
  Role: Busy Working Professional
  Age: 34
  Location: New York, USA
  Demographics: Married, no children, mid-level manager in marketing

- **Goals:**
  - Buy products quickly and easily during breaks
  - Keep track of orders and receive fast deliveries.
  - Frustrated with complicated checkouts that take too much time.
  - Often abandons the cart when shipping costs are unclear.
  - **Behavior:**
  - Shops mostly on mobile, during lunch breaks or commuting.
  - Prefers email and push notifications for order tracking and deals.

What are Story Points -1

- Story points are a unit of measure used in Scrum and other agile development methodologies to estimate the effort required to complete a specific task, user story, or feature.
- Story points are a relative measure rather than an absolute one, and they help development teams estimate the size and complexity of work items in comparison to one another.

What are Story Points -2

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

Relative Estimation: Story points are not based on specific time units like hours or days but are instead relative to other work items. Team members assign story points to a user story or task based on how they perceive its complexity, taking into account factors such as technical difficulty, risks, and uncertainties.

Fibonacci Sequence: Teams often use a modified Fibonacci sequence (1, 2, 3, 5, 8, 13, 21, etc.) to assign story points. This sequence reflects the idea that as tasks become more complex, the level of uncertainty and effort also increases exponentially.

What are Story Points -3

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

Consensus-Based: Estimating story points is a collaborative effort involving team members, typically during a planning meeting like a sprint planning session. Team members discuss the task and collectively agree on the number of story points to assign. This encourages shared understanding and alignment within the team.

Velocity Measurement: Over time, teams track their velocity, which is the number of story points they complete in each sprint. Velocity helps teams understand their capacity and can be used for sprint planning to determine how many story points can be taken on in the upcoming sprint.

No Absolute Scale: Story points are not meant to represent an absolute measure of time or effort but are a tool for relative estimation. Different teams may assign different story point values to the same user story, but what's important is that the team consistently uses their own scale.

What are Story Points -4

Helps with Prioritization: Story points help Product Owners and Scrum Masters prioritize the backlog. Higher story point values may indicate more complexity, and teams may decide to tackle simpler items first to build momentum.

Evolutionary Estimation: As the team learns more about the project and gains experience with estimating, their ability to estimate story points more accurately tends to improve. This makes it easier to plan future sprints and releases.

In summary, story points in Scrum are a tool for estimating the relative effort and complexity of work items in a way that promotes collaboration, helps with sprint planning, and enables teams to track their progress effectively. They are a key element of agile project management and are used to facilitate the iterative and incremental development process

User Story

- **User Stories:** User stories are specific, user-centric descriptions of functionality or requirements. They are small, independently deliverable units of work. Here are some examples:

- **Example User Story 1:**
  - As a registered user, I want to reset my password, so that I can regain access to my account
  - This user story addresses a specific user need related to password management

- **Example User Story 2:**
  - As a customer, I want to be able to add products to my shopping cart, so that I can easily purchase them later
  - This user story focuses on enhancing the shopping experience by allowing users to add items to their cart

- **Example User Story 3:**
  - As a blog author, I want to be able to edit and update my published articles, so that I can correct errors or make improvements
  - This user story addresses the need for content management by authors.

What is meaning of 3Cs in Scrum

- In Scrum, the term "3Cs" refers to the three essential elements that are critical for effective communication and collaboration when defining requirements for a software project
- The 3Cs are:
- **Card:**
  - The "Card" represents a physical or digital card that contains a user story. A user story is a brief, written description of a feature or functionality from an end-user's perspective. It typically follows the template: "As a [user type], I want [an action] so that [benefit or value]."
  - The user story is often written on a physical card or a digital tool, such as a digital card on an Agile project management board
- **Conversation:**
  - The "Conversation" emphasizes the importance of face-to-face or direct communication when elaborating on user stories. It encourages collaboration among team members, including the Product Owner, Scrum Master, and Development Team
  - The conversation is crucial for clarifying details, addressing questions, and ensuring a shared understanding of the user story among all team members
- **Confirmation:**
  - The "Confirmation" refers to the acceptance criteria or conditions of satisfaction associated with a user story. Acceptance criteria define the specific conditions that must be met for a user story to be considered complete
  - By establishing clear acceptance criteria, the team ensures that they understand the expected behavior of the software and that the Product Owner has a measurable standard for determining when a user story is done

Key Principles of 3Cs

- **Collaboration:** The 3Cs emphasize collaboration among team members and stakeholders to ensure a shared understanding of the requirements
- **Focus on Users:** The user story format puts a strong emphasis on understanding and meeting the needs of end-users
- **Adaptability:** The 3Cs support the Agile principle of embracing change. User stories can be easily adapted, and conversations help in refining requirements based on feedback and changing priorities

By focusing on the 3Cs, Scrum teams aim to enhance communication, foster collaboration, and deliver software that directly addresses the needs and expectations of end-users. The 3Cs provide a simple and effective framework for expressing, discussing, and confirming requirements in an Agile and iterative development environment.

3Cs of User stories (Ron Jeffries - 2001)

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

A framework for writing and managing user stories in Agile development

- **Card**: Captures the basic idea of the user story
  Eg., As a [user], I want [some feature] so that [some benefit]

- Not meant to hold every detail about the story but to serve as a **reminder** to facilitate further discussion

- User stories are typically written on 3x5” index cards or sticky notes

3Cs of User stories – contd.

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

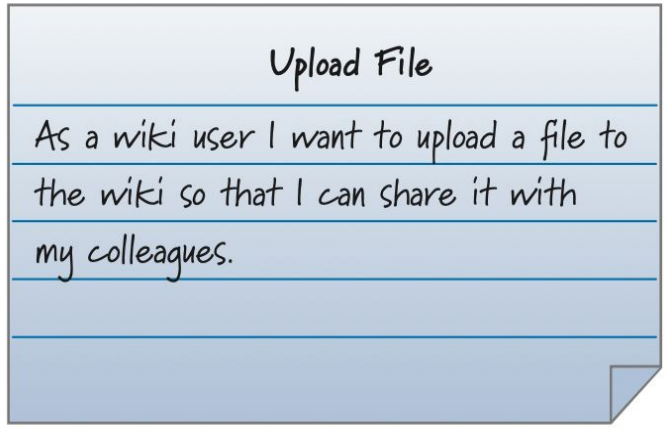
- **Conversation :** the ongoing dialogue between team members (developers, testers, product owners, and stakeholders) about the user story.
- The goal of the conversation is to clarify the details, understand the user's needs, and ensure that the development team fully comprehends what is being asked.
- The conversation allows flexibility—new information can be added, and the scope of the story can evolve based on feedback and discussion

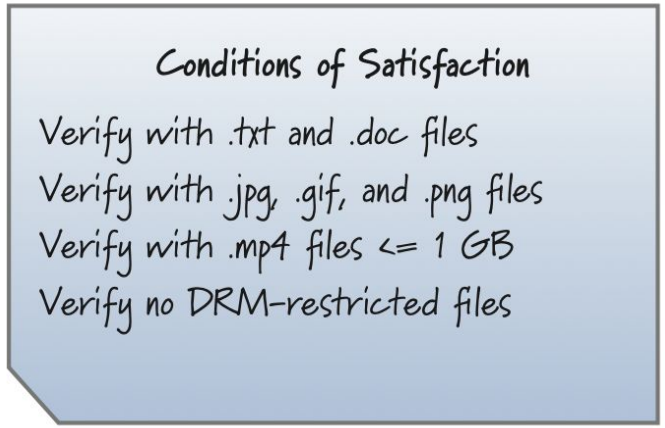
A sample conversation in the "Upload File story

Initially, let's limit uploaded file sizes to be 1 GB or less. Also, make sure that we can properly load common text and graphics files. And for legal reasons we can't have any files with digital rights management (DRM) restrictions loaded to the wiki.

3Cs of User stories (Contd.)

- **Confirmation :** refers to the acceptance criteria, which are the conditions that need to be met for the story to be considered complete.
- These are **specific, testable criteria** that ensure the functionality works as intended and meets the user's needs.
- Confirmation helps both the development team and product owner agree on when the story is done. This may include functional tests, user acceptance tests (UAT), or non-functional criteria like performance or security requirements.





3Cs of User stories (Contd.)

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **Conversation :** the ongoing dialogue between team members (developers, testers, product owners, and stakeholders) about the user story.
- The goal of the conversation is to clarify the details, understand the user's needs, and ensure that the development team fully comprehends what is being asked.
- The conversation allows flexibility—new information can be added, and the scope of the story can evolve based on feedback and discussion

T

A sample conversation in the "Upload File story

Initially, let's limit uploaded file sizes to be 1 GB or less. Also, make sure that we can properly load common text and graphics files. And for legal reasons we can't have any files with digital rights management (DRM) restrictions loaded to the wiki.

What is meaning of “INVEST” in Scrum -1?

INVEST is an acronym used in Scrum and Agile software development to define a set of criteria for creating well-formed user stories. Each letter in the acronym represents a characteristic that a user story should possess to be considered valuable and effective

- **Independent:**
  - The "I" in INVEST stands for "Independent." Each user story should be independent of others, meaning that it can be developed and tested in isolation. This promotes flexibility and allows for easier prioritization and adaptation to changing requirements
- **Negotiable:**
  - The "N" stands for "Negotiable." User stories should be negotiable, implying that they are not contracts but rather starting points for discussions. The details can be negotiated between the Product Owner and the development team during Sprint Planning

What is meaning of “INVEST” in Scrum-2?

- **Independent:**
- **Negotiable:**
- **Valuable:**
  - The "V" represents "Valuable." Each user story should deliver value to the end-users or stakeholders. It's crucial to focus on stories that contribute directly to the project's goals and provide a tangible benefit
- **Estimable:**
  - The "E" stands for "Estimable." A user story should be clear and understandable enough for the development team to estimate the effort required for its implementation. Ambiguous or vague stories make accurate estimation challenging

What is meaning of “INVEST” in Scrum -3?

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **Independent:**
- **Negotiable:**
- **Valuable:**
- **Estimable:**
- **Small:**
  - The "S" denotes "Small." User stories should be small enough to be completed within a single Sprint. Smaller stories are easier to estimate, develop, test, and manage. They also allow for faster feedback and iteration
- **Testable:**
  - The "T" in INVEST stands for "Testable." Each user story should have clear acceptance criteria that make it testable. This ensures that the development team and stakeholders have a shared understanding of what constitutes a successful implementation

What is meaning of “INVEST” in Scrum?

- **Why INVEST Matters:**
  - **Flexibility:** INVEST promotes flexibility in adapting to changing requirements by ensuring that user stories are independent and negotiable
  - **Value-Driven:** Focusing on valuable user stories ensures that the team prioritizes work that aligns with the project's goals and delivers meaningful outcomes
  - **Efficiency:** Estimable and small user stories contribute to more accurate estimation, better planning, and efficient development
  - **Quality Assurance:** Having testable user stories with clear acceptance criteria facilitates effective testing and quality assurance

INVEST serves as a valuable guideline for Agile teams to create user stories that are well-defined, manageable, and contribute to the overall success of the project

User story Vs Story point

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

1. **User Story**:
    1. A user story is a brief, simple description of a feature or functionality from the perspective of the end user. It typically follows the format: "As a [type of user], I want [some goal] so that [some reason]."
    2. User stories help teams understand what needs to be built and why, focusing on the user's needs and the value the feature will provide.
2. **Story Point**:
    1. A story point is a unit of measure used to estimate the effort required to implement a user story. It considers factors like complexity, risk, and the amount of work involved1
    2. Story points are relative and abstract, meaning they don't correspond directly to a specific amount of time. Instead, they help teams gauge the relative effort of different tasks and plan their work accordingly.

Example of Acceptance Criteria for a User Story:

User Story: "As a registered user, I want to reset my password so that I can regain access to my account.“

Acceptance Criteria:

1. The "Forgot Password" link is visible on the login page
2. Clicking on the "Forgot Password" link opens a page with a form to enter the registered email address
3. Entering a valid email address and submitting the form triggers an email with a password reset link to be sent to the user
4. The password reset link is valid for 30 minutes
5. Clicking on the password reset link redirects the user to a page where they can enter and confirm a new password
6. Entering a new password and clicking "Submit" updates the user's password in the system
7. The user receives a confirmation message indicating that their password has been successfully reset

These acceptance criteria provide a detailed and testable definition of what it means for the user story to be completed. They guide the development team in building the expected functionality and help stakeholders in evaluating whether the user story meets their requirements

Acceptance Criteria

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

In Scrum, **Definition of Done (DoD)** and **Acceptance Criteria** are distinct concepts, though they both ensure quality and completeness of work

- **Scope**: Specific to individual PBIs or user stories.
- **Purpose**: Defines the conditions that a particular PBI must meet to be accepted by the product owner or stakeholders. It focuses on the functionality and behaviour of the item.
- **Creation**: Typically written by the product owner in collaboration with stakeholders and the development team.

Definition of Done (DOD)

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

In Scrum, **Definition of Done (DoD)** and **Acceptance Criteria** are distinct concepts, though they both ensure quality and completeness of work

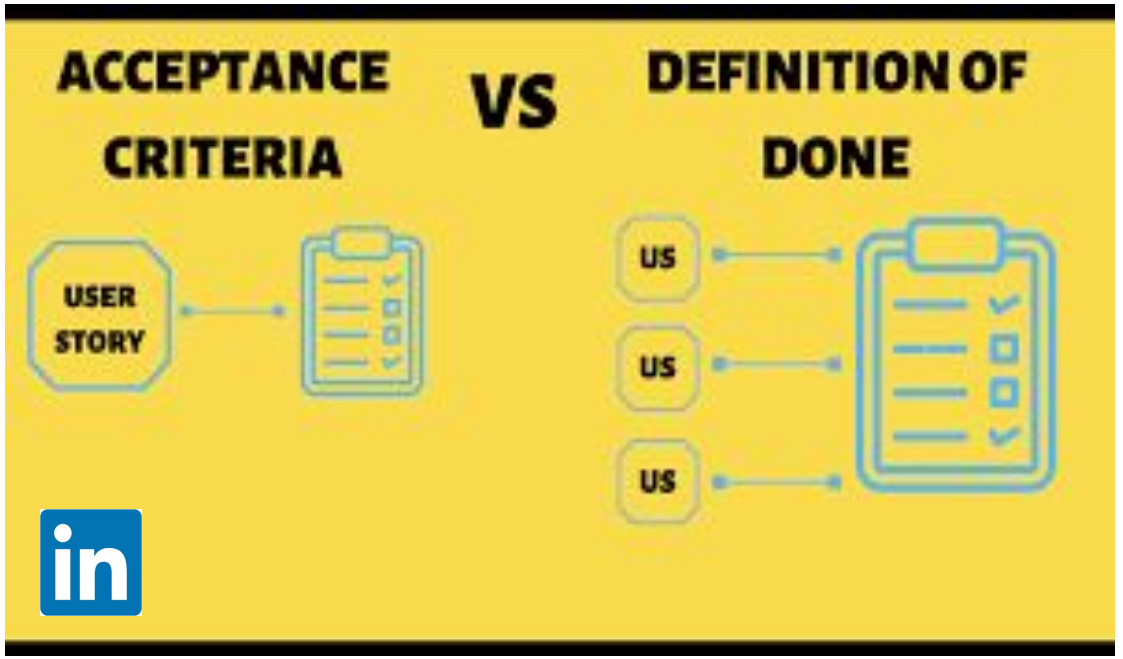
- **Scope**: Applies to all Product Backlog Items (PBIs) and ensures that each increment meets the team's quality standards.
- **Purpose**: Ensures consistency and completeness across all work items. It includes criteria like code quality, testing, documentation, and compliance[1]
- **Creation**: Defined by the entire Scrum team and remains relatively stable

	Definition of Done
❑	Design reviewed
❑	Code completed
❑	Code refactored
❑	Code in standard format
❑	Code is commented
❑	Code checked in
❑	Code inspected
❑	End-user documentation updated
❑	Tested
❑	Unit tested
❑	Integration tested
❑	Regression tested
❑	Platform tested
❑	Language tested
❑	Zero known defects
❑	Acceptance tested
❑	Live on production servers

DOD Vs Acceptance Criteria

In Scrum, **Definition of Done (DoD)** and **Acceptance Criteria** are distinct concepts, though they both ensure quality and completeness of work

- While the Definition of Done ensures that all increments meet a consistent quality standard, Acceptance Criteria are specific to each backlog item and ensure that the item meets the expected functionality and requirements.[1]



Features

- Features represent broader functionalities or capabilities within a software product. They often encompass multiple user stories that collectively deliver a more significant part of the product's functionality. Here are some examples:
- **Example Feature 1: User Account Management:**
  - This feature includes user stories related to user registration, login, password reset, and profile management. It represents the overall functionality for managing user accounts in the application
  - User Story 1 (as shown above) could be one of the user stories within this feature
- **Example Feature 2: Shopping Cart:**
  - This feature includes user stories related to adding products to the cart, viewing the cart, updating quantities, and proceeding to checkout. It represents the end-to-end shopping cart functionality
  - User Story 2 (as shown above) is a part of this feature
- **Example Feature 3: Content Management:**
  - This feature includes user stories related to creating, editing, deleting, and publishing articles or content on a blog or website. It encompasses all the functionality required for content management
  - User Story 3 (as shown above) is one of the user stories within this feature

User Stories vs Features

- User stories are individual, user-focused requirements that are small in scope and deliverable within a single iteration.
- Features, on the other hand, represent larger, customer-visible functionalities that may consist of multiple related user stories.
- User stories are used to break down features into actionable, implementable units of work, making it easier for development teams to plan and prioritize their work in an agile environment.

Scrum Artifacts

- In Scrum, there are three primary artifacts that serve as information radiators and provide transparency into the work being done
- These artifacts help the Scrum Team and stakeholders understand the current state of the project and facilitate effective communication
- The three main artifacts in Scrum are:
  1. Product Backlog
  2. Sprint Backlog
  3. Increment

Product Backlog

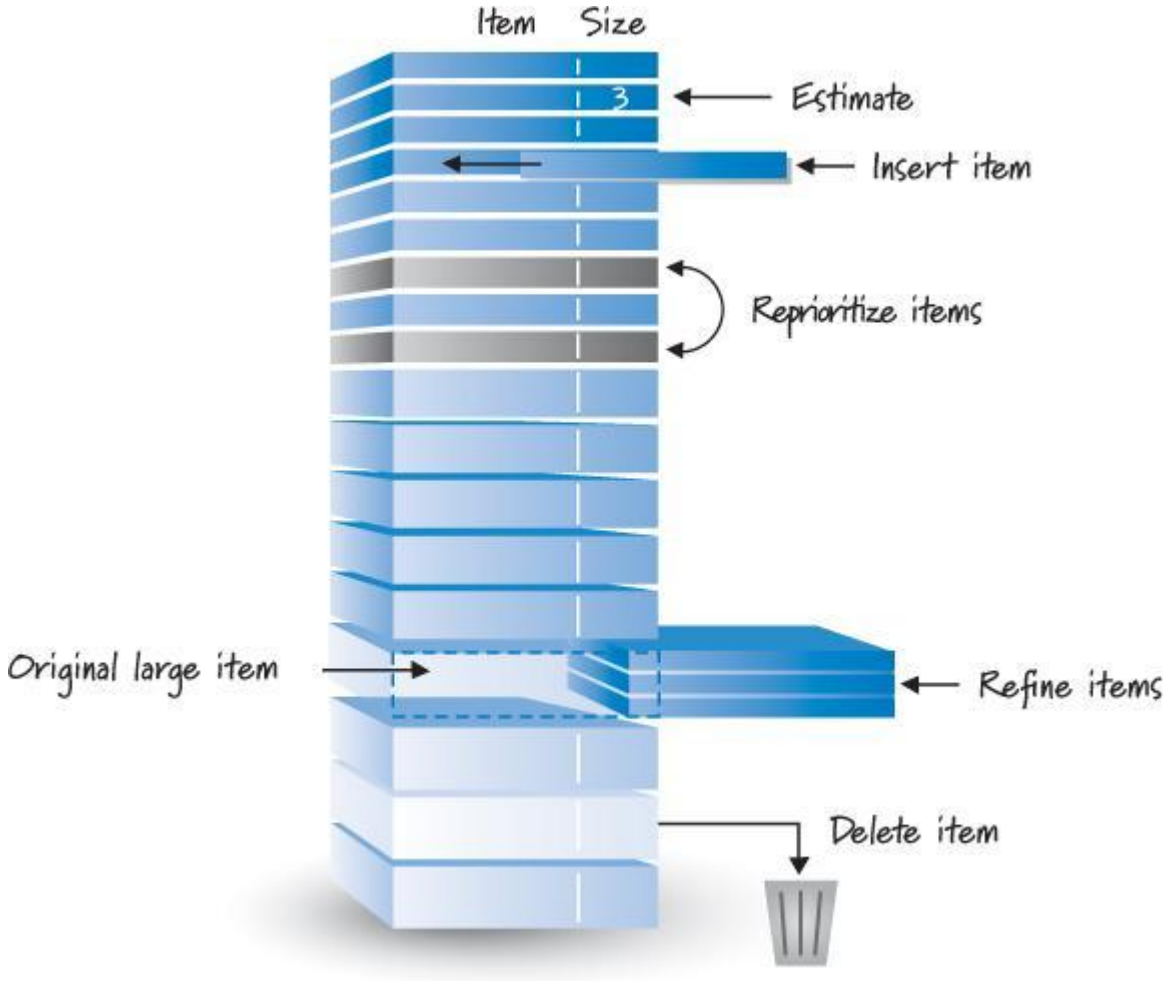
- The Product Backlog is a dynamic, ordered list of all the work that needs to be done on the project
- It is managed by the Product Owner and serves as the single source of truth for what needs to be built
- The Product Backlog is continuously refined, with new items added, existing items reprioritized, and details added as the understanding of the product evolves

Product Backlog Item (Examples)

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

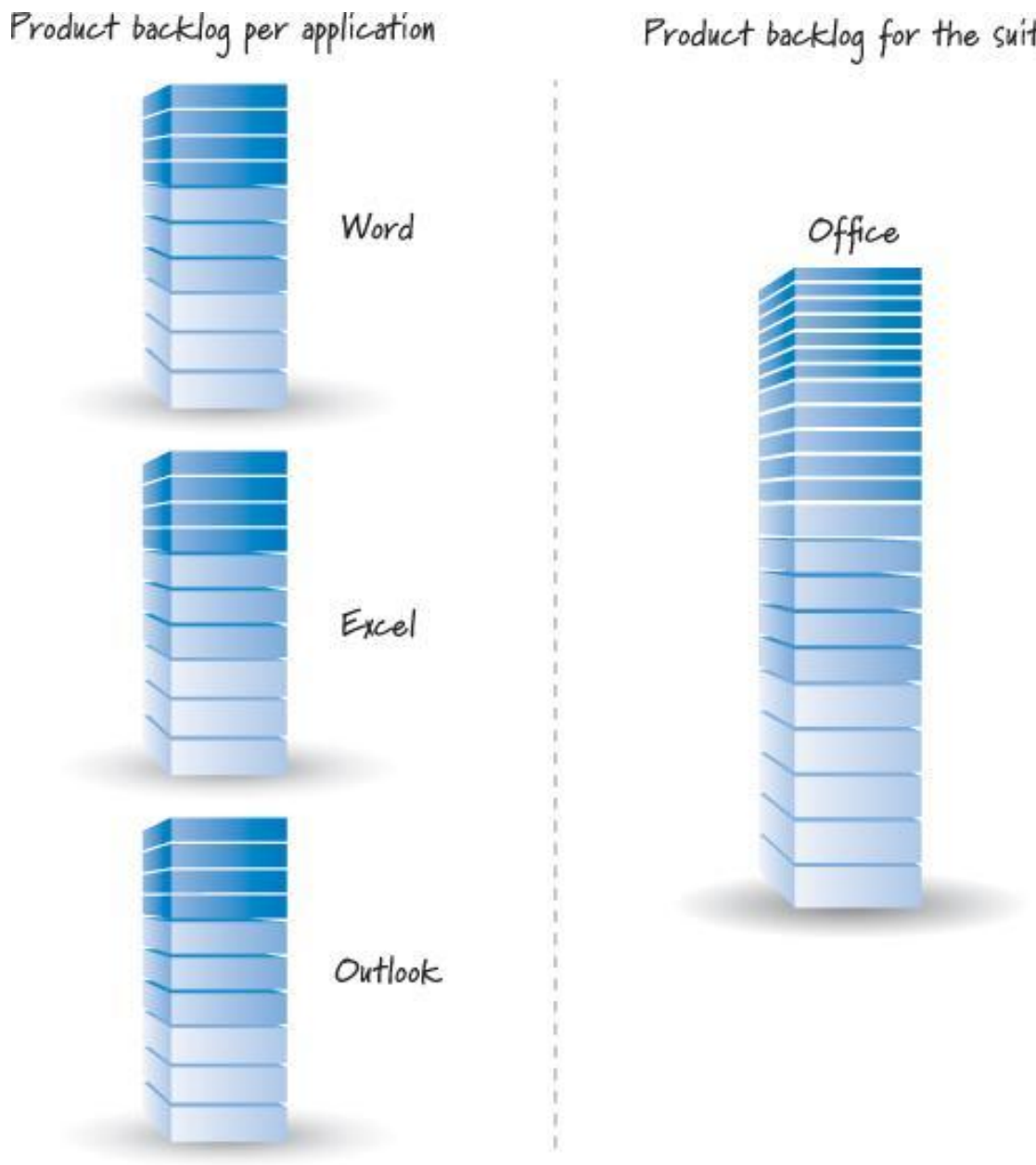
PBI Type	Example
Feature	As a customer service representative I want to create a ticket for a customer support issue so that I can record and manage a customer's request for support.
Change	As a customer service representative I want the default ordering of search results to be by last name instead of 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.

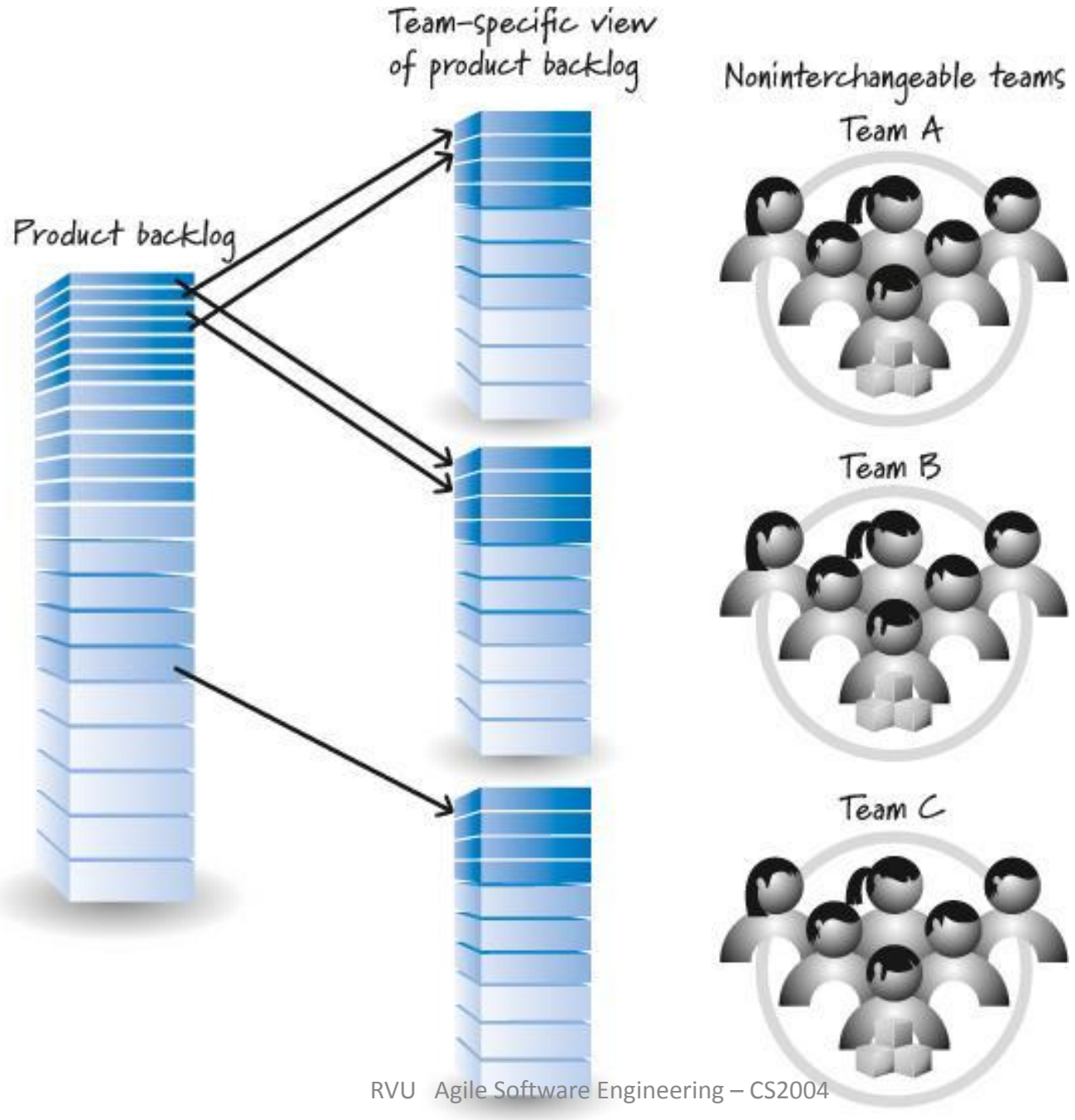
Grooming



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
Word
Excel
Outlook
Product backlog for the suite
Office


Team-specific view of product backlog
Noninterchangeable teams
Product backlog
Team A
Team B
Team C

TFD (Test First Development)

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **Approach**: In TFD, tests are written before the production code. The primary goal is to ensure that the code is testable from the start.
- **Process**: It focuses on writing tests first but does not necessarily follow a strict iterative process. The design and implementation can be more upfront

TDD (Test Driven Development)

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **Approach**: TDD also involves writing tests before the code, but it emphasises a cyclical process of writing a test, writing just enough code to pass the test, and then refactoring.

- **Process**: TDD follows a specific cycle known as Red-Green-Refactor

  - **Red**: Write a test that fails (since the feature isn't implemented yet).

  - **Green**: Write the minimum code necessary to pass the test.

  - **Refactor**: Improve the code while ensuring the tests still pass

TFD Vs TDD

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- Test First Development (TFD) and Test-Driven Development (TDD) are related but distinct methodologies in software development.
- While both methodologies start with writing tests, TDD is more structured and iterative, focusing on incremental development and continuous refactoring.
- TFD, on the other hand, is more flexible and can involve more upfront design

Pair programming

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

Pair programming is a core practice in Extreme Programming (XP), where two developers work together at one workstationProcess. Here's how it works:

- **Roles**: One developer, known as the "driver," writes the code, while the other, the "observer" or "navigator," reviews each line of code as it is written. The roles are frequently switched to ensure both developers stay engaged and share knowledge.

- **Collaboration**: This practice promotes continuous code review, leading to higher code quality and fewer defects. It also facilitates knowledge sharing between team members, which can be particularly beneficial when pairing a more experienced developer with a less experienced one

Pair programming –contd.

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **Communication**: Pair programming enhances communication and collaboration within the team. It encourages developers to discuss their ideas and approaches, leading to better problem-solving and innovation
- **Efficiency**: Although it might seem counterintuitive, pair programming can be more efficient in the long run. The immediate feedback and shared responsibility often result in faster identification and resolution of issues

Code Review

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- Code review in Agile is a systematic process in which team members examine each other's code to ensure quality, improve the development process, and share knowledge. It is an essential practice in Agile methodologies that:
  - helps maintain high standards of code,
  - fosters collaboration, and
  - promotes teamwork

Code Review – Purpose

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **Quality Assurance**: Identify bugs, errors, and potential issues before the code is merged into the main codebase.
- **Knowledge Sharing**: Facilitate learning and knowledge transfer among team members, particularly for less experienced developers.
- **Encourage Best Practices**: Promote coding standards, design principles, and best practices.

Code Review : Process

RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **Pull Requests**: Developers create pull requests (PRs) to propose their code changes and request review from teammates.
- **Review Comments**: Reviewers leave comments, suggestions, and feedback on the PR, which the author can respond to and incorporate into their code.
- **Approval and Merging**: Once the reviewer is satisfied with the changes, they approve the PR, and the code can be merged into the main branch.

Code Review : Tools & Timing

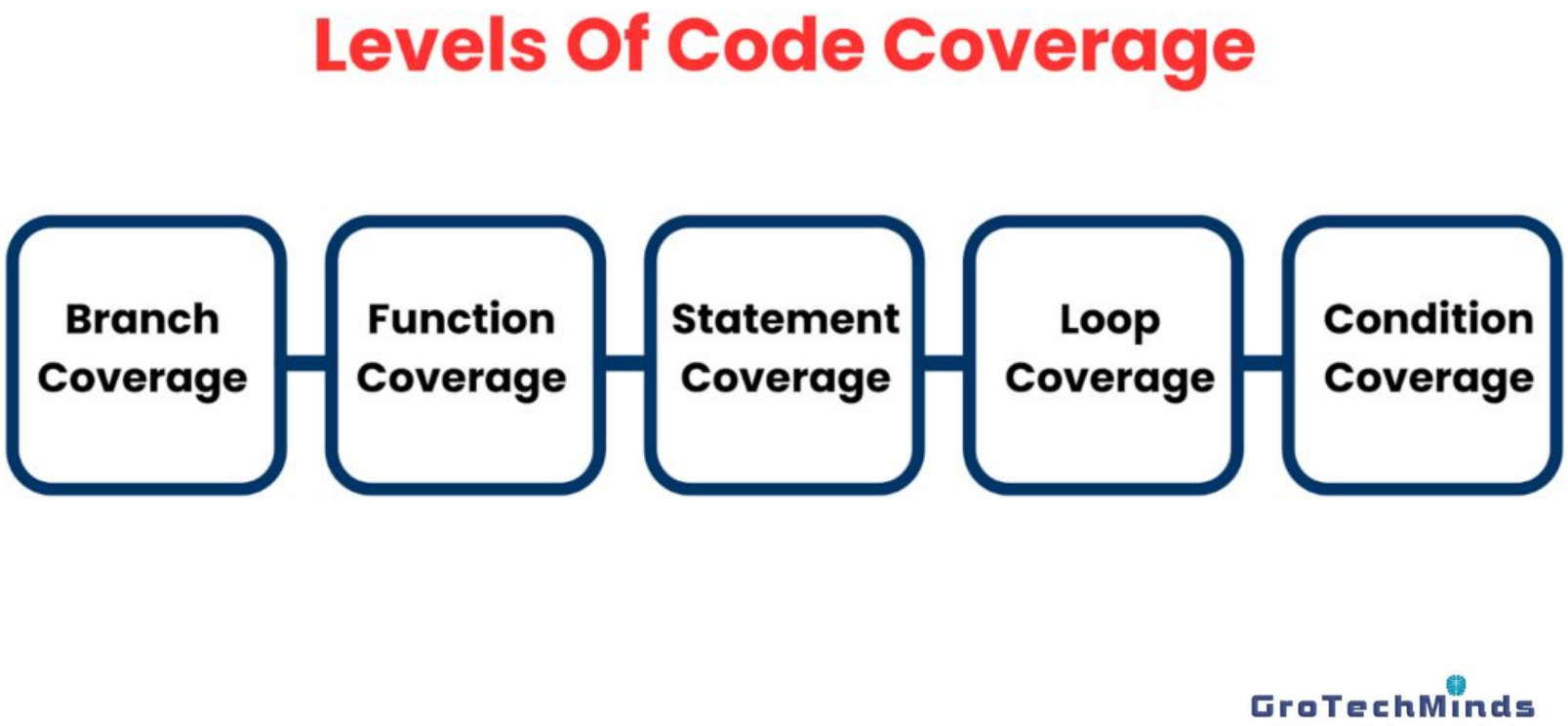
RV UNIVERSITY
Go, change the world
an initiative of RV EDUCATIONAL INSTITUTIONS

- **Tools** : GitHub, GitLab, Bitbucket, or others that facilitate code review processes by providing interfaces for inline comments, discussions, and tracking changes.
- **Timing**: Typically done after the initial implementation is done but before any major releases. Agile's iterative approach means reviews can happen frequently and be integrated into sprints

Code Coverage

- **Code coverage** is a measure used to describe the degree to which the source code of a program is tested by a particular test suite.
- It provides an indication of how much of the code is exercised during testing, helping to identify untested parts of the codebase.
- Higher code coverage often correlates with higher code quality, as it suggests that more code paths and scenarios have been tested.

Code Coverage



Key Metrics of Code Coverage

- **Line Coverage**: The percentage of lines of code executed by the tests.
- **Branch Coverage**: The percentage of branches (e.g., if-else statements) taken during testing.
- **Function Coverage**: The percentage of functions/methods called during testing.
- **Statement Coverage**: The percentage of executable statements run by the tests.
- **Path Coverage**: The percentage of possible execution paths that have been followed

Code Coverage Tools

- **Cobertura:** It is one of the most used and widely known code coverage tools. It is popular for being a free Java tool that calculates the percentage of code accessed by various tests performed.
- **JaCoCo:** It is a free Open Source code coverage tools for Java
- **Coverage.py:** It is a code coverage tool built for Python. Mainly, Coverage.py is used in the execution, analysis, and reporting phases. This tool monitors the Python programs and after monitoring the parts of the code that have been executed, it analyzes the source to identify code that could have been executed but couldn't.
- **Istanbul:** Istanbul is a test coverage tool that works with many different frameworks. It tracks the parts of your code that are executed by your unit tests.