Introduction to Scrum Teams

Welcome to our exploration of Scrum Teams! In this module, we'll dive deep into the heart of Scrum, focusing on how people work together effectively to bring ideas to life and create value. Understanding Scrum Teams is crucial for implementing Scrum successfully in any organization.

by Mayko Silva

Overview of Scrum Teams

1 **Diving Deep into Scrum**

We'll explore the core concepts and principles that make Scrum Teams effective.

2 **Focus on Collaboration**

Understanding how people work together is key to successful Scrum implementation.

3 **Effective Teamwork**

We'll examine the factors that contribute to high-performing Scrum Teams.

Exploring Scrum Team Composition

Team Composition

We'll explore the essential roles that make up a Scrum Team and how they work together.

Cross-functional Teams

Learn about the importance of having diverse skills within a self-managing Scrum Team.

Team Size and Structure

Discover the ideal Scrum Team size and how multiple teams can work on a single product.

Additional Topics in Scrum Teams

Debunking Misconceptions

We'll address common misunderstandings about Scrum Teams and their roles.

Practical Considerations

Explore real-world scenarios, such as developers serving as Scrum Masters.

Team Dynamics

Understand the interpersonal aspects that contribute to successful Scrum Teams.

Module Objectives

Scrum Team Structure

By the end of this module, you'll understand how Scrum Teams are structured.

1

2

Design Rationale

You'll learn why Scrum Teams are designed the way they are.

Supporting Scrum Principles

3

Understand how the team structure supports Scrum pillars and values.

The Importance of Scrum Teams

Idea Incubator

Scrum Teams are where ideas turn into reality.

Problem-Solving Hub

Teams work together to overcome challenges and find solutions.

Value Creation Center

Scrum Teams are at the heart of creating value for the organization.

Key Takeaway

1

Understand Scrum Teams

Gain a deep understanding of how Scrum Teams function.

2

Apply Knowledge

Use this understanding to improve team dynamics and performance.

3

Effective Implementation

Understanding Scrum Teams is crucial for effective Scrum implementation.

Let's Begin Our Journey!

1 **Ready to Explore**

Are you ready to dive deep into the world of Scrum Teams?

2 **Exciting Journey Ahead**

Prepare for an informative and engaging exploration of Scrum Team dynamics.

3 **Valuable Insights**

Gain insights that will help you build and work with effective Scrum Teams.

Composition of a Scrum Team

by Mayko Silva

Key Quote from Scrum Guide

> "The fundamental unit of Scrum is a small team of people, a Scrum Team. The Scrum Team consists of one Scrum Master, one Product Owner, and Developers."

- The **Scrum Team** is the core unit of Scrum
- It is composed of **three distinct roles**:
  - One Scrum Master
  - One Product Owner
  - Developers

Scrum Team Roles

Scrum Master (1 per team)

Ensures team understands and follows Scrum practices

Product Owner (1 per team)

Accountable for maximizing product value

Developers

- Create the product increment
- Broad term, not limited to software developers

Important Notes

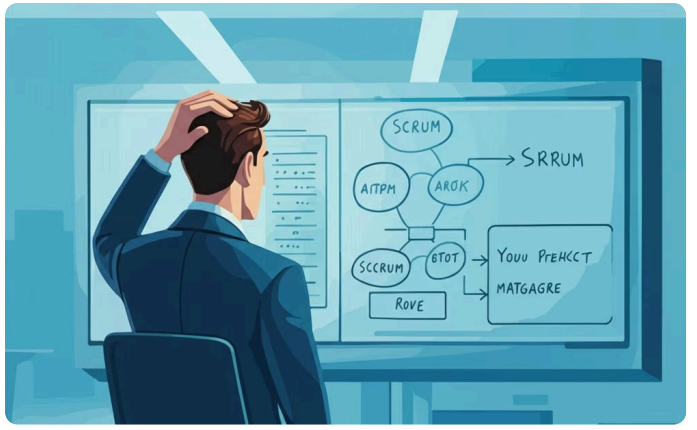
Cohesive Unit

Scrum Team is a cohesive unit with no sub-teams or hierarchies

Single Objective

Team is focused on one objective: the Product Goal

Common Misconceptions



No Project Manager role in Scrum

Scrum doesn't include a traditional Project Manager position

No separate QA teams, testers, or business analysts

Scrum integrates these roles within the Development Team

"Developer" isn't limited to software developers

In Scrum, "Developer" includes anyone contributing to the product's development

Practical Example: Mobile App Team

Leadership Roles

- Scrum Master
- Product Owner

Development Team

Developers including:

- Software engineers
- UX designers
- QA specialists
- Content writers

Understanding Check

Which best describes Scrum Team composition?

- A) One Scrum Master, multiple Product Owners, Developers
- B) One Scrum Master, one Product Owner, Developers
- C) Multiple Scrum Masters, one Product Owner, Developers
- D) One Scrum Master, one Product Owner, one Project Manager, Developers

Answer and Explanation

The correct answer is B: One Scrum Master, one Product Owner, and Developers. This composition aligns perfectly with the definition provided in the Scrum Guide.

This structure represents the essential roles within a Scrum Team, ensuring a balanced and effective approach to project management and product development. Each role has specific responsibilities that contribute to the team's overall success.

Scrum Team Structure Benefits

Cross-functionality

Promotes diverse skill sets within the team

Self-management

Encourages team members to take ownership and make decisions

Enhanced focus

Allows team to concentrate on project goals without distractions

Minimal dependencies

Minimizes external dependencies, reducing bottlenecks

Efficiency and adaptability

Increases overall efficiency and ability to adapt to changes

Remember

Core Scrum Team

- One Scrum Master
- One Product Owner
- Developers

Essential Components

These roles form the core of every Scrum Team, working together to achieve project goals.

Next Steps

Cross-functional Teams

In-depth exploration

Team Dynamics

Analyzing interactions

Skill Development

Enhancing capabilities

Exploring cross-functional teams in depth is the next step in our journey to understand and implement effective Scrum practices. This exploration will involve analyzing team dynamics and focusing on skill development to enhance the overall capabilities of the Scrum team.

No Subteams in Scrum

by Mayko Silva

Key Quote from Scrum Guide

> "Within a Scrum Team, there are no subteams or hierarchies. It is a cohesive unit of professionals focused on one objective at a time, the Product Goal."

- No subteams within a Scrum Team
- No hierarchies within a Scrum Team
- Cohesive unit of professionals
- Focused on one objective: the Product Goal

No Subteams

Unified Team Structure

No separate QA, testing, or specialized subgroups within the Scrum team.

Developer Role

All team members are considered Developers, regardless of their specific skills or expertise.

Shared Responsibility

Collective responsibility for all aspects of product development is emphasized.

No Hierarchies

Equal Standing

No titles like "Team Lead" or "Senior Developer". All Developers are equal in standing within the Scrum team.

Flat Structure

Flat structure promotes open communication and shared responsibility among team members.

Rationale Behind This Approach

The rationale behind the "no subteams" approach in Scrum is twofold. First and foremost, it creates a cross-functional team that is capable of self-management. This means that the team as a whole possesses all the necessary skills and expertise to handle the various aspects of product development without the need for separate specialized groups.

Secondly, this approach ensures that the Scrum team is equipped to handle all aspects of product development. By avoiding subteams or hierarchies, every team member is empowered to contribute across different areas of the project, fostering a more holistic and integrated approach to product creation.

Benefits of No Subteams/Hierarchies

Benefit	Description
Flexibility	Fluid movement between different tasks
Shared Responsibility	Everyone responsible for all aspects
Reduced Bottlenecks	No waiting for specific subteams
Improved Communication	Information flows freely

The absence of subteams and hierarchies in Scrum brings several key benefits to the development process:

- Flexibility: Fluid movement between different tasks
- Shared Responsibility: Everyone responsible for all aspects
- Reduced Bottlenecks: No waiting for specific subteams
- Improved Communication: Information flows freely

These advantages contribute to a more efficient and cohesive team environment, fostering innovation and rapid problem-solving.

Practical Example: Mobile App Development

In mobile app development using Scrum, there are no separate coding, design, and testing subteams. Instead, all team members collaborate on all aspects of the project. This approach provides flexibility to adapt to changing priorities and challenges that may arise during the development process.

- No separate coding, design, and testing subteams
- All team members collaborate on all aspects
- Flexibility to adapt to changing priorities and challenges

Understanding Check

Which scenario aligns with Scrum principles?

A) QA subteam checks development subteam's work

B) Senior Developer approves all code changes

C) All team members collaborate on testing and QA

D) UI/UX subteam hands off designs to coding subteam

Answer and Explanation

Correct answer: C

- All team members collaborate on all aspects
- Reflects "no subteams, no hierarchies" principle

Remember

Collaboration

No subteams and hierarchies promotes collaboration among team members, fostering a unified approach to problem-solving.

Flexibility

The absence of rigid structures allows for greater flexibility in adapting to changing project needs and requirements.

Shared Responsibility

Without subteams, all members share responsibility for the project's success, encouraging a sense of ownership.

The elimination of subteams and hierarchies is a key factor in creating high-performing, self-managing teams. This approach fosters collaboration, flexibility, and shared responsibility among team members.

Next Steps

Explore

Flat Structure

Cross-functionality

Exploring how flat, cohesive structure supports cross-functionality

Cross-Functional and Self-Managed Teams in Scrum

In Scrum, cross-functional and self-managed teams are essential components that drive project success. These teams possess a unique combination of skills and autonomy, allowing them to efficiently deliver value in each Sprint. This presentation will explore the key aspects of cross-functionality and self-management in Scrum teams, their benefits, and practical applications.

by Mayko Silva

Key Quote from Scrum Guide

Cross-Functional Teams

"Scrum Teams are cross-functional, meaning the members have all the skills necessary to create value each Sprint.

Self-Managing Teams

They are also self-managing, meaning they internally decide who does what, when, and how."

Cross-Functionality

1 **Collective Skills**

Team collectively has all necessary skills

2 **Comprehensive Development**

Can handle all aspects of product development

3 **Specialized Roles**

Not every member does everything

Self-Management

1. **Work Allocation**
   Team decides how to accomplish work
2. **Task Distribution**
   Determines who does what, when, and how
3. **Autonomy**
   Not directed by external parties

Benefits of Cross-Functionality and Self-Management

Autonomy

Quick decision-making

Flexibility

Adaptation to changing circumstances

Ownership

Full responsibility for work and outcomes

Innovation

Creative solutions through diverse skills and freedom

Practical Example: Mobile App Team

1

Team Decisions

Team decides:

2

UI Design

Alice focuses on UI design

3

Backend Development

Bob and Charlie pair program on backend API

4

Testing

Diana writes automated tests

5

Collaborative Testing

Everyone participates in daily testing

6

Internal Decision-Making

Decisions made internally based on skills and needs

Understanding Check

Scenario

Product Owner suggests sub-teams for efficiency. Best response?

Options

A) Agree and implement sub-teams
B) Explain Scrum teams should remain cross-functional and self-managing
C) Ask Scrum Master to decide
D) Implement sub-teams on trial basis

Answer and Explanation

Correct Answer

B) Explain Scrum teams should remain cross-functional and self-managing

Explanation

- Maintain cross-functional, self-managing teams
- No sub-teams or hierarchies in Scrum
- Promotes flexibility, shared responsibility, better communication

Remember

1. **Essential Components**

   Cross-functionality and self-management are essential

2. **Key Benefits**

   Enable quick adaptation, ownership, consistent value delivery

3. **Core Principles**

   Not just nice-to-have features

Next Steps

1

Future Topic

Exploring why Scrum Teams are typically limited to 10 or fewer members

Why only 10 to a Scrum team?

by Mayko Silva

Key Quote from Scrum Guide

> "The Scrum Team is small enough to remain nimble and large enough to complete significant work within a Sprint, typically 10 or fewer people. In general, we have found that smaller teams communicate better and are more productive."

- Scrum Team size: **10 or fewer people**
- Small enough to remain nimble
- Large enough to complete significant work within a Sprint
- Smaller teams: **better communication** and increased productivity

Communication Efficiency

Direct and Efficient Communication

Direct and efficient communication is achieved in smaller teams, allowing for streamlined information flow.

Staying in the Loop

Everyone can stay in the loop without excessive meetings, promoting a more productive work environment.

Agility and Flexibility

Rapid Adaptation

Smaller teams adapt more quickly to changes

Swift Decision-Making

Faster decision-making with fewer people involved

Productivity

Less Overhead

Smaller teams experience less overhead, allowing for more efficient use of time and resources. This streamlined approach enables team members to focus on their core tasks without excessive administrative burdens.

Improved Focus

With fewer team members, it's easier to maintain focus and avoid distractions. This concentrated environment promotes deeper engagement with tasks and helps team members stay on track with their objectives.

Accountability

Visible Contributions

Each member's contribution is more visible in a small team, making it easier to recognize individual efforts and accomplishments.

Issue Transparency

In a smaller team setting, it's harder for issues to hide or for work to fall through the cracks, promoting greater accountability among team members.

Team Dynamics

Strong Relationships

Easier to build strong relationships and trust within smaller teams

Conflict Resolution

Conflicts often easier to resolve with fewer people involved

Practical Example: Dinner Party

To illustrate the importance of team size in Scrum, let's consider a dinner party analogy:

- 10 or fewer guests: Single conversation, easy idea sharing, quick decisions
- 20-30 people: Fragmented communication, complicated decision-making, harder engagement

This example demonstrates how smaller groups facilitate better communication and collaboration, much like an ideal Scrum team.

Understanding Check

Your organization is starting a new project and wants to form a Scrum Team. They're considering creating a team of 15 people to cover all necessary skills. What would be the most appropriate response from a Scrum perspective?

A) Agree to the 15-person team to ensure all skills are covered.

B) Suggest creating two smaller Scrum Teams, each with 7-8 members.

C) Propose reducing the team to 10 people and outsourcing any missing skills.

D) Recommend a 10-person team and coach them on becoming more cross-functional.

Answer and Explanation

The correct answer is D. Let's break down the key points of this answer:

Aim for teams of 10 or fewer people: This aligns with the Scrum Guide's recommendation for small, focused teams. Keeping the team size to 10 or fewer members helps maintain effective communication and collaboration.

Focus on developing cross-functional skills within the team: This approach encourages team members to broaden their skill sets, making the team more versatile and capable of handling various aspects of the project without relying heavily on external resources.

Multiple small teams preferred over one large team if more capacity needed: When a project requires more manpower, it's better to create multiple small Scrum teams rather than expanding a single team beyond the recommended size. This preserves the benefits of small team dynamics while increasing overall capacity.

Remember

Recommendation, Not Rule

"10 or fewer" is a recommendation, not a hard rule. It's based on observed patterns of team effectiveness in Scrum implementations.

Team Size Considerations

Keep teams small enough to be nimble and communicate effectively. Teams should have capacity to deliver significant value each Sprint.

Next Steps

Explore

Multiple Teams

Larger Projects

Exploring how multiple Scrum Teams can work together on larger projects

Can Developers also be Scrum Masters?

by Mayko Silva

Scrum Guide Perspective

The Scrum Guide, which is the definitive source for Scrum framework guidelines, doesn't explicitly forbid Developers from taking on the role of Scrum Masters. This lack of a direct prohibition opens up the possibility for such a dual role. However, it's crucial to note that while the guide doesn't outright ban this practice, it also doesn't endorse it.

Given this ambiguity, it becomes important to carefully consider the implications and potential challenges that may arise when a Developer attempts to simultaneously fulfill the responsibilities of a Scrum Master. These considerations are essential for maintaining the integrity and effectiveness of the Scrum process within a team or organization.

Role Clarity

Scrum Master Responsibilities

The Scrum Master role has distinct responsibilities within the Scrum framework. These include facilitating Scrum events, coaching the team on Scrum practices, and removing impediments to the team's progress.

Developer Responsibilities

Developers also have their own set of responsibilities, primarily focused on creating and delivering product increments. This involves coding, testing, and collaborating with other team members to meet sprint goals.

Balancing Act

Balancing both Scrum Master and Developer roles can be challenging for one person. This is due to the distinct nature of each role and the time and focus required to fulfill their respective responsibilities effectively.

Time Management

Significant Demands

Both roles require significant time and attention, making it challenging to balance responsibilities effectively.

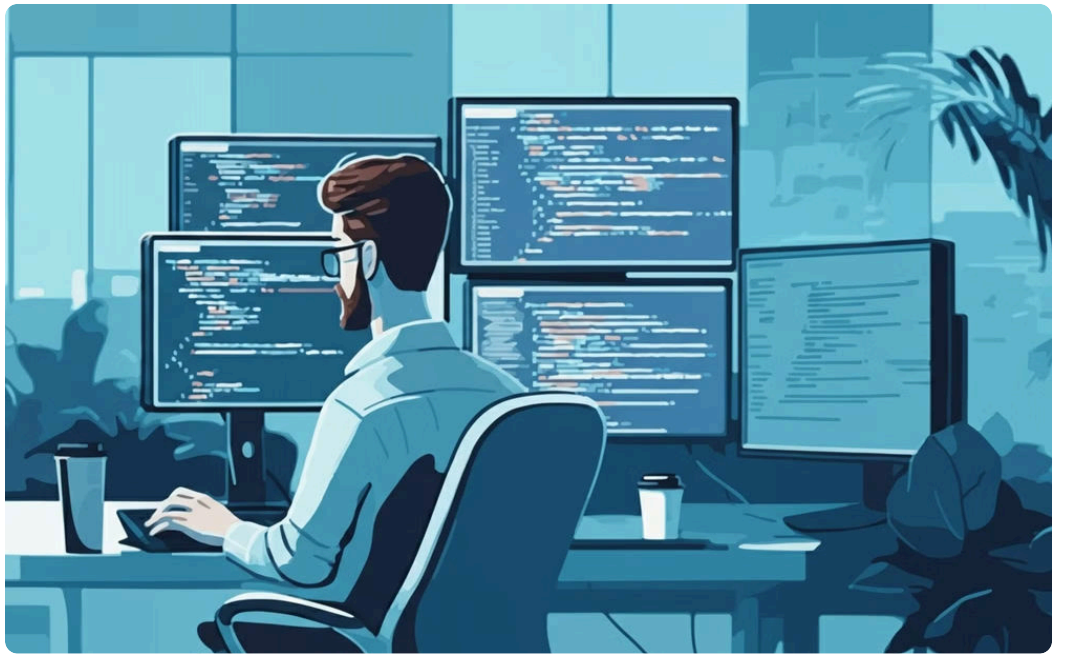
Focus Challenges

Individuals may struggle to give each role proper focus, potentially compromising the quality of work in both areas.

Potential Conflicts of Interest

Scrum Master Neutrality

Scrum Master should be neutral and process-focused



Developer's Perspective

Developer might be too close to day-to-day work for neutrality

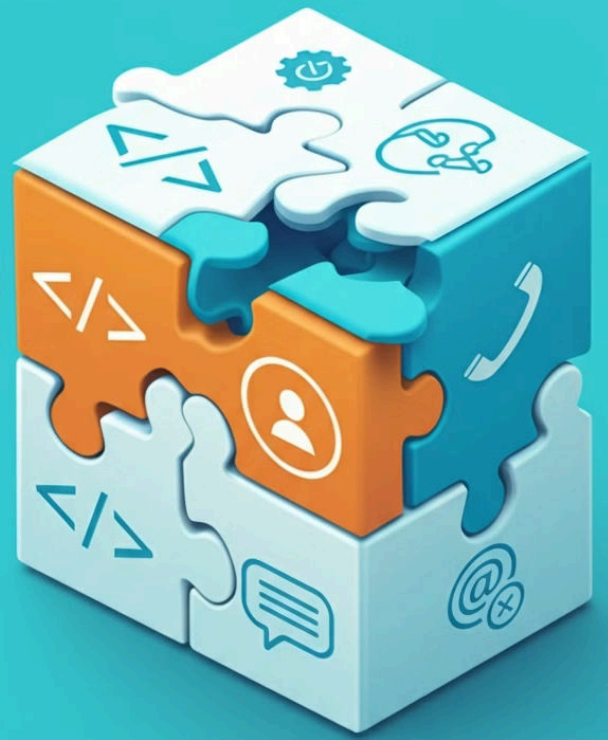
Team Dynamics

Dedicated Scrum Master Advantage

A dedicated Scrum Master provides an outside perspective to the team, offering a unique viewpoint that can be valuable in identifying and addressing team dynamics issues.

Role-Switching Challenges

When a developer takes on the Scrum Master role, they may struggle to switch between roles effectively. This can lead to confusion and potentially impact the team's dynamics negatively.



Skill Set

Developer Skills

Developers require technical programming skills and problem-solving abilities.

Scrum Master Skills

Scrum Masters need facilitation and coaching skills, which not all Developers possess.

Developer and Scrum Master roles require different skill sets. While Developers focus on technical aspects, Scrum Masters need strong interpersonal and leadership abilities. It's important to note that not all Developers have the facilitation and coaching skills necessary for effective Scrum Master performance.

Situations Where It Might Work

Small Teams or Startups

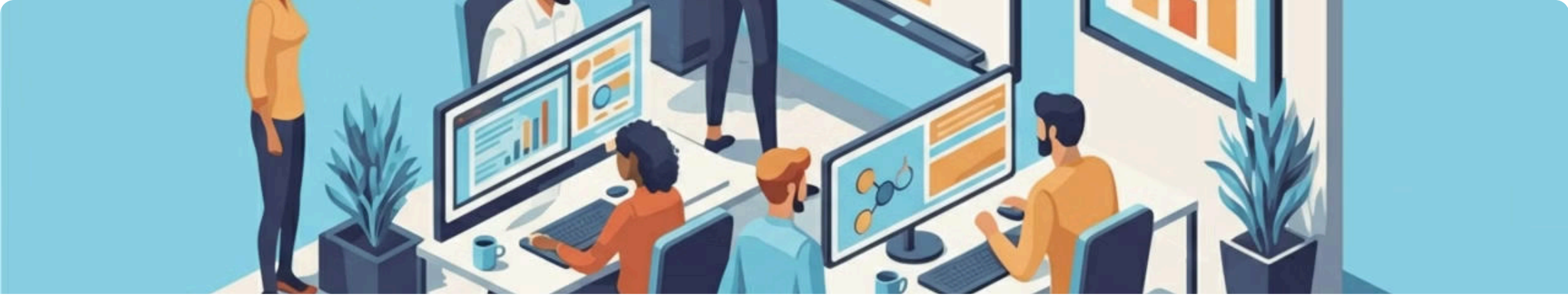
In smaller organizations or startup environments, where resources are limited and roles often overlap, a developer taking on the Scrum Master role might be more feasible and even beneficial.

Temporary Situations

During transitional periods or when there's a sudden need for a Scrum Master, a developer might step in temporarily to fill the role until a dedicated Scrum Master can be found.

Part-Time Needs

In scenarios where the Scrum Master role doesn't require full-time attention, a developer might be able to balance both responsibilities effectively, especially if they have strong organizational and leadership skills.



Practical Example: Small Startup

- Team of five people
- All working on product development
- Need someone to facilitate Scrum events and coach
- Developer with good Scrum understanding might take on both roles

Understanding Check

The Scenario

In a Scrum Team of 7 people, one of the Developers has extensive Scrum knowledge and offers to also serve as the Scrum Master.

The Question

What would be the most appropriate response from a Scrum perspective?

Options

- A) Agree, as the Scrum Guide doesn't forbid this arrangement.
- B) Disagree, as the Scrum Master and Developer roles must always be separate.
- C) Suggest a trial period to see if the arrangement works for the team.
- D) Recommend hiring a dedicated Scrum Master to keep the roles separate.

Answer and Explanation

Correct answer: C

- Scrum Guide doesn't forbid Developer as Scrum Master
- Important to consider potential challenges
- Trial period allows team to assess effectiveness in their context

Remember

Flexibility in Scrum

Scrum allows flexibility based on team needs and circumstances

Dedicated Roles

Generally recommended to have dedicated roles

Effective Fulfillment

Most important: all responsibilities are fulfilled effectively

Next Steps

1

Explore Multi-Team Collaboration

Exploring how multiple Scrum Teams can work on a single product

2

Clarify Roles and Responsibilities

Strive for clarity in roles and responsibilities

3

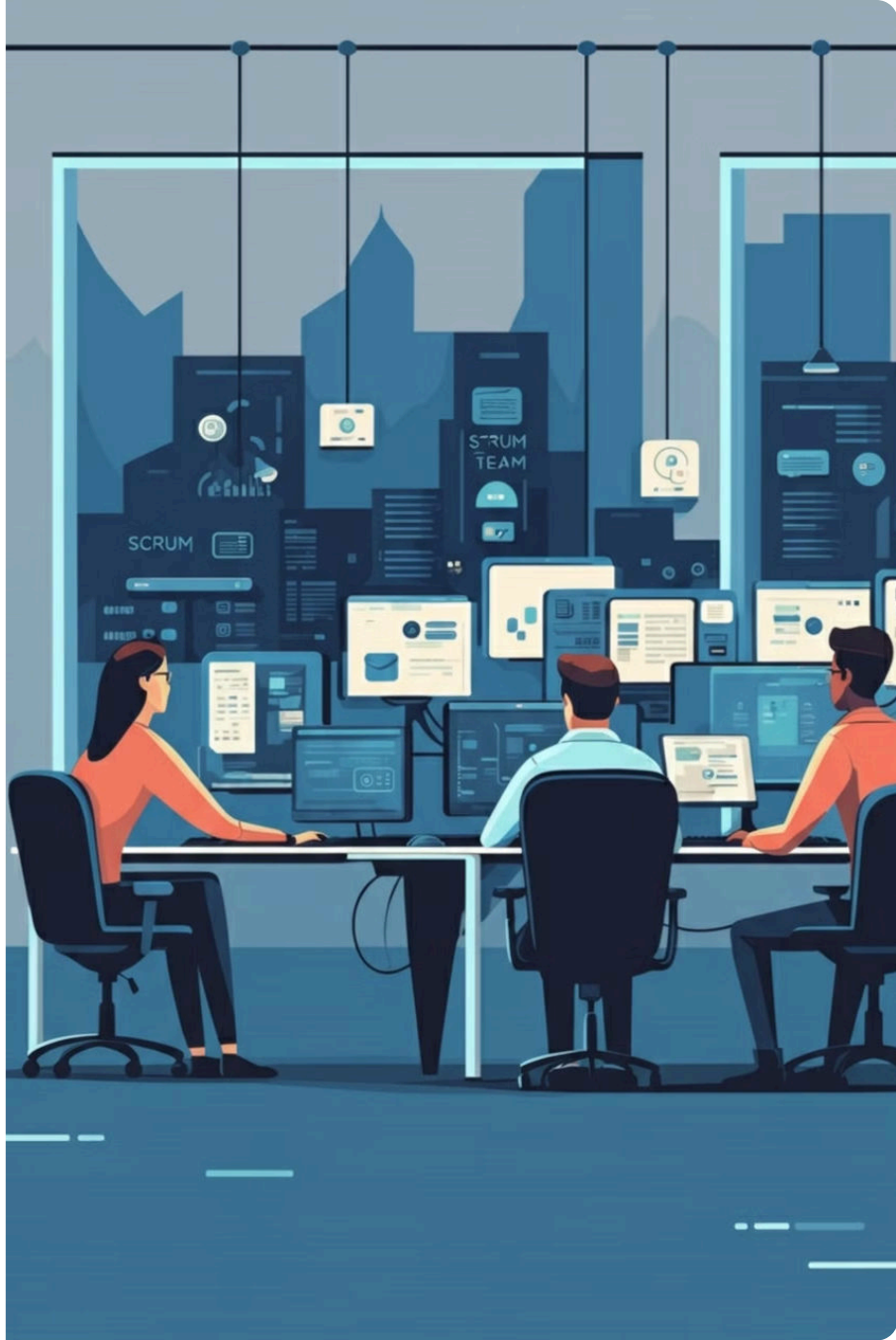
Maintain Flexibility

Remain flexible and pragmatic in applying Scrum framework

Multiple Scrum Teams Working on One Product

Coordinating multiple Scrum teams on a single product to maximize efficiency and collaboration.

by Mayko Silva



Key Quote from Scrum Guide

Scrum Guide states that if Scrum Teams become too large, they should reorganize into multiple cohesive teams focused on the same product.

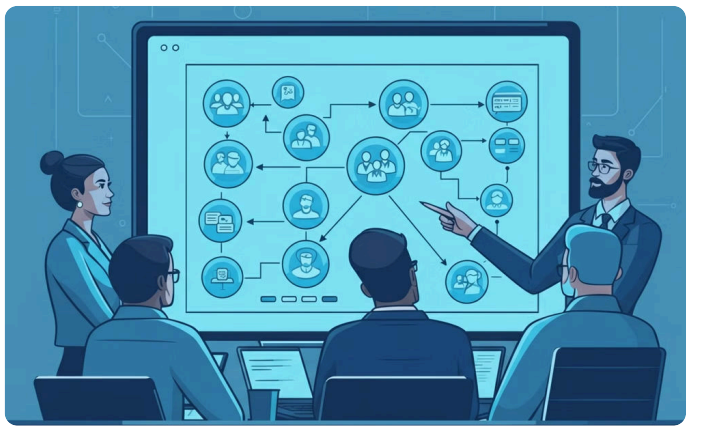
Shared Product Goal

Aligned Direction

All teams focused on same overarching goal

Unified Vision

Ensures teams work towards a common purpose



Coordinated Efforts

Aligns teams to deliver a cohesive product

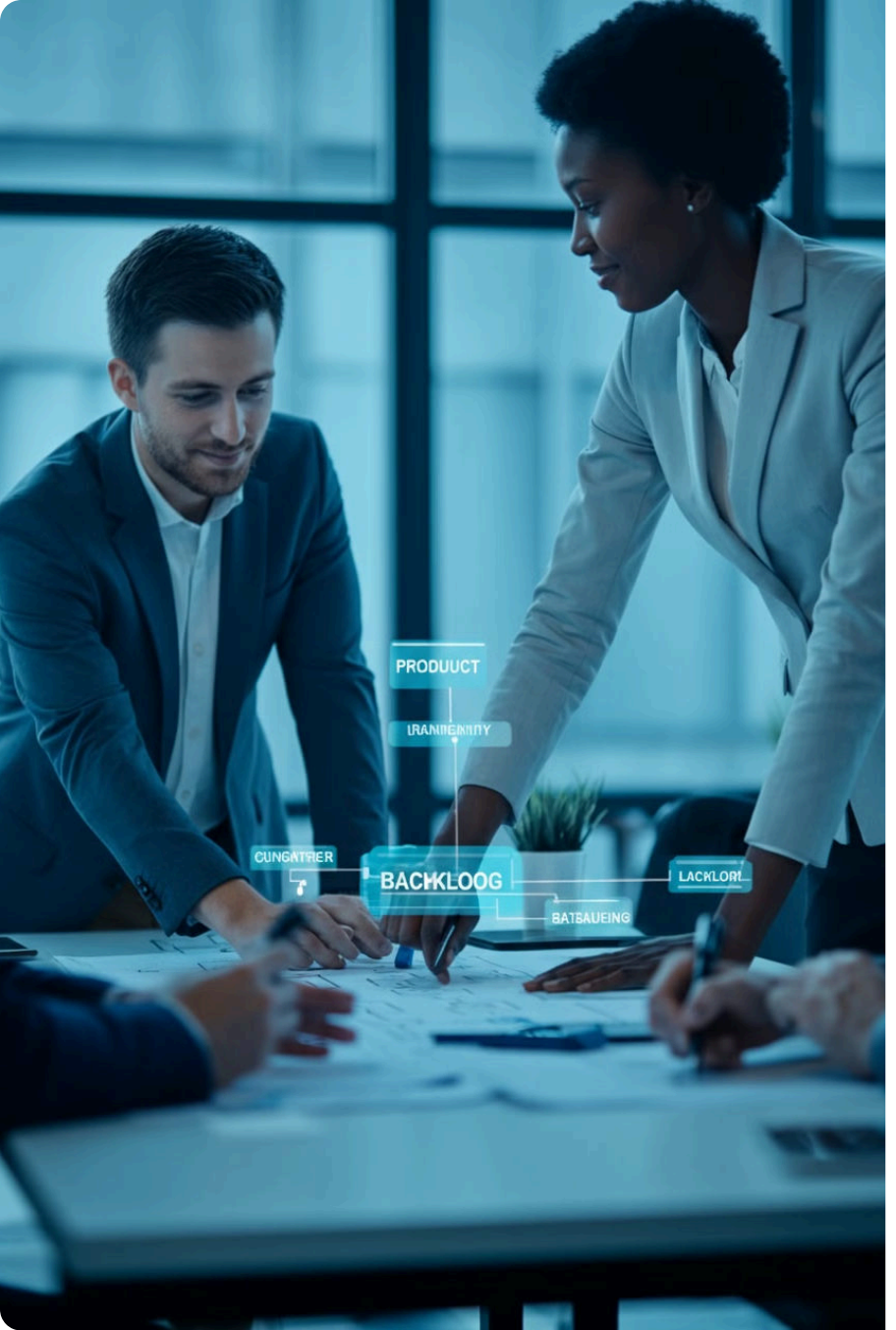
Shared Product Backlog

Single Backlog

Maintains unified vision and prioritization

Entire Product

Covers all work for the complete product



Shared Product Owner

One Product Owner

Manages Backlog for all teams

Consistent Decisions

Ensures prioritization and decision-making

Team Size

Split into Smaller Teams

When teams get too large (>10 people), split into multiple teams.

Better Communication

Smaller teams communicate better and are more productive.

Increased Productivity

Smaller teams are more efficient and get more done.

Team Independence

Independent Operations

Teams work autonomously despite shared elements

Unique Sprint Backlogs

Each team maintains its own Sprint Backlog

Team Decision-Making

Teams make own choices to achieve Sprint Goal

Practical Example: E-commerce Platform

Three teams of 6-7 people each:

- Team A: User interface and customer experience
- Team B: Backend systems and databases
- Team C: Payment processing and security

All share same Product Owner and Product Backlog. Each manages own Sprint Backlog and work.

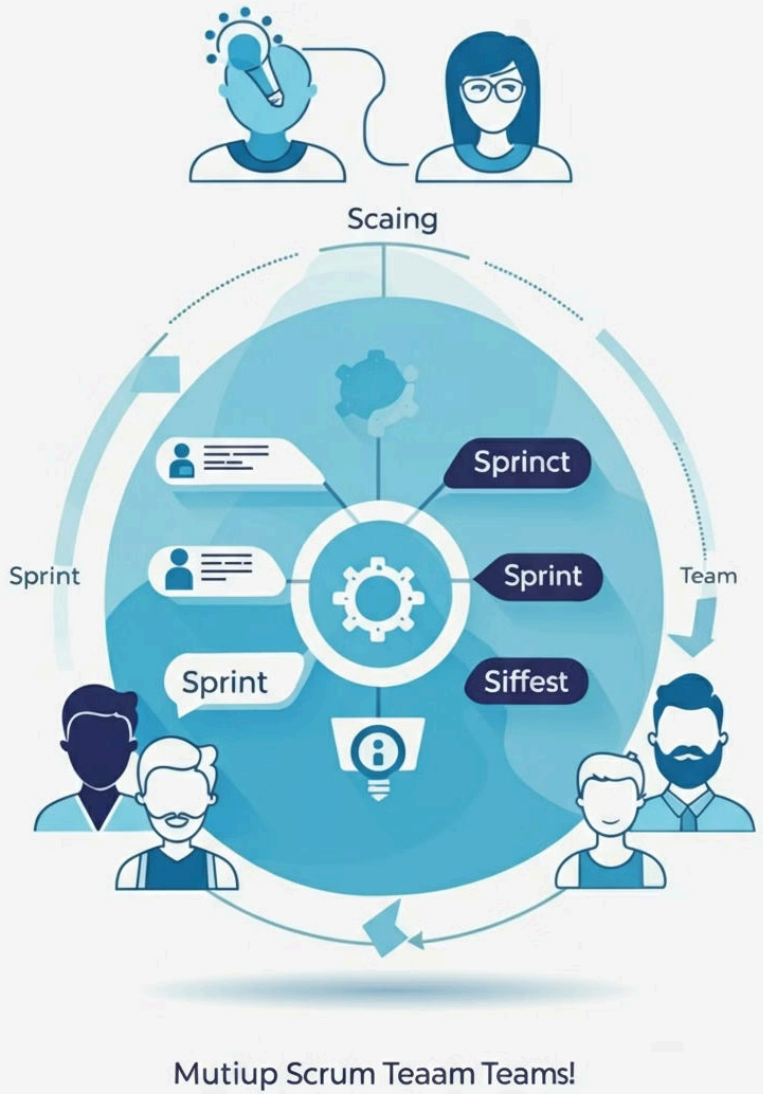
Understanding Check

A large project has been divided into three Scrum Teams. Which of the following is NOT a correct approach according to Scrum?

A) Each team has its own Product Owner to focus on their specific area. B) All teams share the same Product Backlog.

C) Each team has its own Sprint Backlog.

D) All teams work towards the same Product Goal.



Answer and Explanation

The correct answer is A. In Scrum, when multiple teams work on the same product, they share the same Product Owner. Having separate Product Owners could lead to conflicting priorities and a fragmented product vision.

Shared Product Owner

Multiple teams share the same Product Owner

Conflicting Priorities

Separate Product Owners could lead to conflicting priorities

Fragmented Product Vision

Separate Product Owners could lead to a fragmented product vision

Remember

Alignment

Multiple teams aligned towards same product goal

Guidance

Guided by same product owner

Consistency

Working from same product backlog

Next Steps

1

Explore Coordination

Learn how multiple teams can effectively coordinate their work

2

Understand Dependencies

Identify and manage dependencies between teams working on the same product

3

Optimize Processes

Refine processes to improve collaboration and productivity across teams

Scrum Team Responsibilities

by Mayko Silva

Key Quote from Scrum Guide

"The Scrum Team is responsible for all product-related activities including stakeholder collaboration, verification, maintenance, operation, experimentation, research and development, and anything else that might be required."

End-to-End Responsibility

Entire product development process

Scrum teams are responsible for the complete product development journey

No handoffs to separate teams

The same Scrum team handles all aspects without transferring to other groups

From ideation to delivery and maintenance

Scrum teams oversee the product from initial concept through to ongoing support

Stakeholder Collaboration

Active Engagement

Active engagement with stakeholders

Understanding Needs

Understanding needs and gathering feedback

Whole Team Involvement

Whole team involved, not just Product Owner

Verification and Quality Assurance

Ensuring Quality

The Scrum Team is responsible for ensuring the quality of their work. This involves implementing rigorous quality control measures throughout the development process.

Testing Activities

All testing activities fall under the team's responsibility. This includes a wide range of tests, from unit tests to user acceptance testing, ensuring comprehensive coverage.

Maintenance and Operations

Ongoing Responsibility

The Scrum Team is responsible for ongoing maintenance and operation of the product

Beyond Development

The team's role extends beyond just building and handing off the product

Experimentation and Innovation

New Ideas

Scrum teams are encouraged to experiment with new ideas, fostering a culture of innovation within the project.

Creative Problem-Solving

Teams are empowered to solve problems creatively, thinking outside the box to overcome challenges.

R&D

Conducting research and development is an integral part of the Scrum team's responsibilities, driving progress and improvement.

Continuous Improvement

Improving Processes and Practices

Scrum teams are responsible for continuously enhancing their workflows and methodologies

Ongoing Responsibility

The commitment to improvement is a constant aspect of the Scrum team's duties

Practical Example: Mobile Banking App

Development and Design

- Collaborating with bank stakeholders
- Designing and developing features

Quality Assurance

- Conducting thorough testing
- Deploying to app stores

Maintenance and Improvement

- Monitoring performance and feedback
- Maintaining with regular updates
- Researching new technologies

Understanding Check

Your Scrum Team has just finished developing a new feature. The Product Owner suggests handing it off to a separate QA team for testing before the Sprint Review. What would be the most appropriate response from a Scrum perspective?

- A) Agree, specialized QA provides thorough testing
- B) Disagree, testing is Scrum Team's responsibility
- C) Suggest QA team joins Scrum Team for this Sprint
- D) Ask Scrum Master to decide

Answer and Explanation

Correct answer: B

- Scrum Team responsible for all aspects of development
- Including testing and quality assurance
- Separate QA team contradicts end-to-end responsibility

Remember

Comprehensive Responsibilities

Scrum Team responsibilities are comprehensive, spanning the entire product development lifecycle

Full Ownership

Ensures full ownership of product, allowing quick response to changes and challenges

Next Steps

1

Explore Responsibility Management

Exploring how to manage wide-ranging responsibilities

2

Maintain Focus

While maintaining focus and productivity

3

Balance and Efficiency

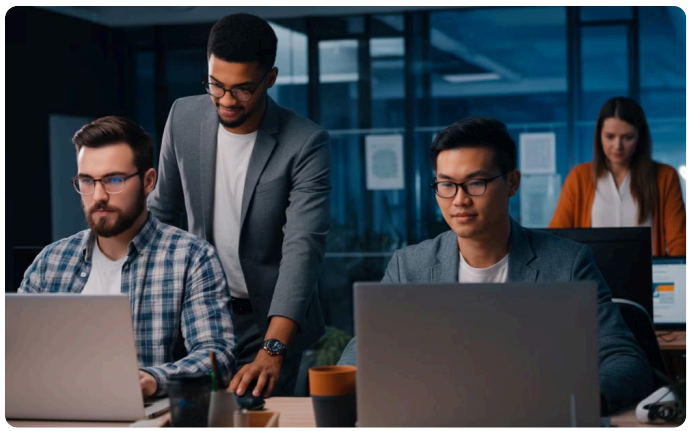
Striking a balance between diverse tasks and efficient execution

End-to-End Feature Development in Scrum

Explore the full lifecycle of feature development in an Agile Scrum framework.

by Mayko Silva

Core Idea

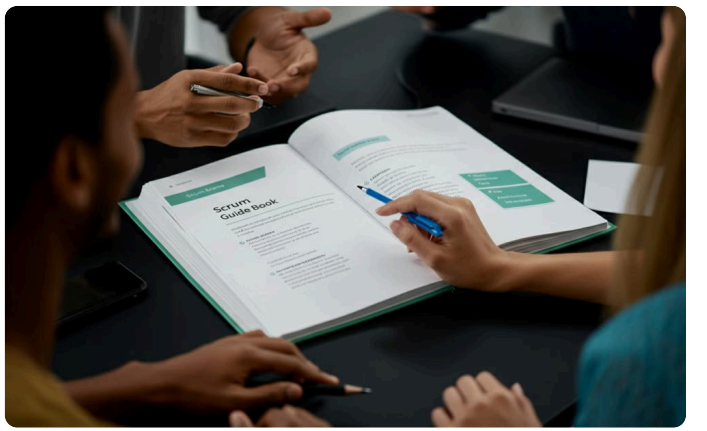


Team Responsibility

Team responsible for all aspects of creating a feature

From Conception to Delivery

Responsible for feature from start to finish



Aligns with Scrum

Follows Scrum Guide: "The Scrum Team is responsible for all product-related activities"

Cross-Functional Responsibility

Comprehensive Skills

Team must have or acquire all necessary skills

No Handoffs

No handoffs to separate teams for specific tasks

Comprehensive Development Process

Handles Everything

From design to deployment

Includes UAT

Traditionally separate activities

Quality Assurance

Built-in Quality

Quality is integrated into the development process.

Cross-Functional

Quality is not the responsibility of a separate QA team.

Definition of Done

Work must meet quality criteria in the Definition of Done.

Continuous Learning

New Skills

Team expected to learn new skills as needed

Skill Development

Promotes continuous skill development

Team Versatility

Increases team versatility over time

Ownership and Accountability

Full Ownership

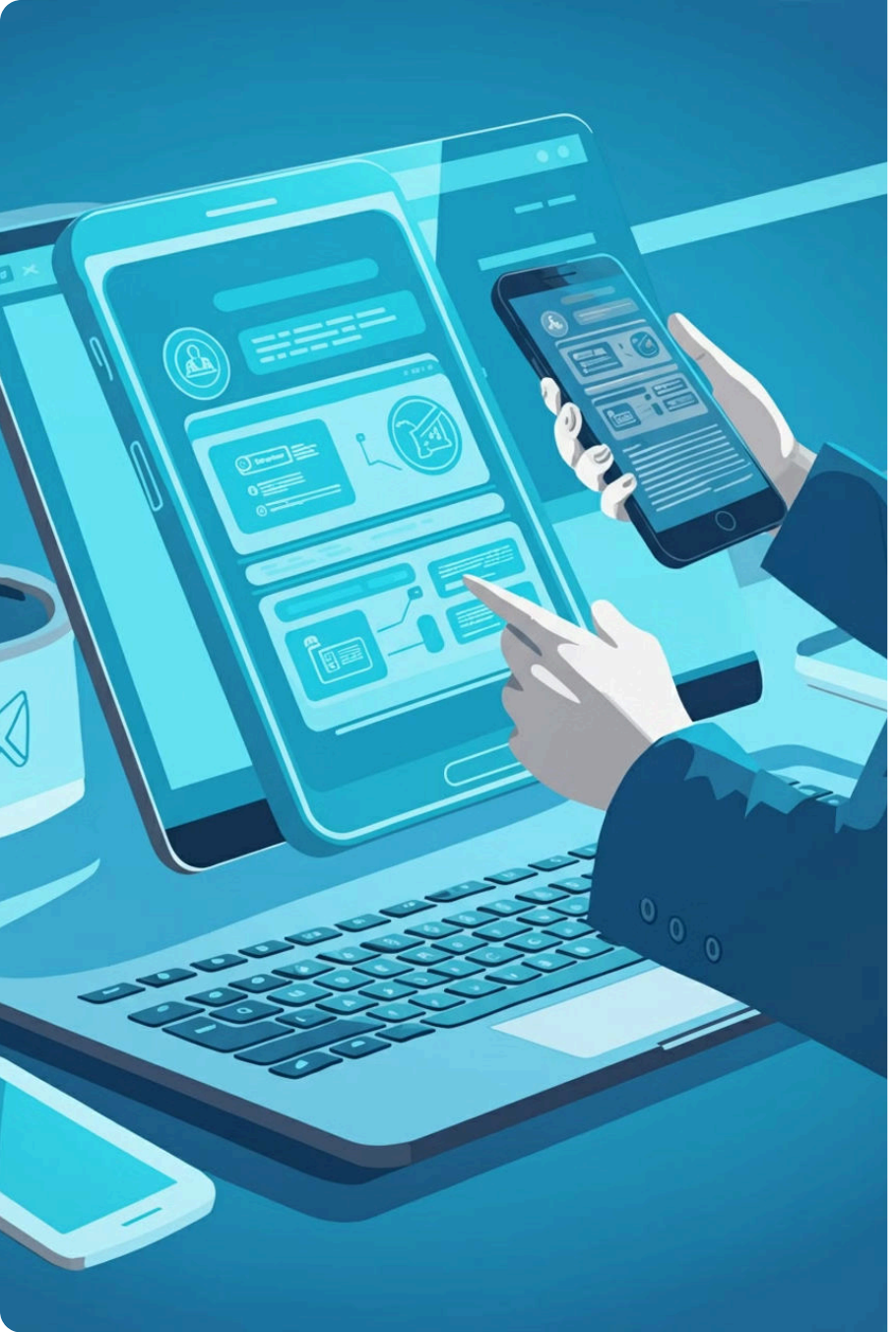
Features owned from start to finish

Pride in Work

Fosters a sense of pride and accomplishment

High-Quality Output

Encourages team members to deliver excellent work



Practical Example: Mobile Banking App Feature

1 **Collaboration on Requirements**

Team works together to define feature needs

2 **Comprehensive Development**

Covers UI design, backend logic, testing, security

3 **Deployment and Monitoring**

Feature is documented, deployed, and performance tracked

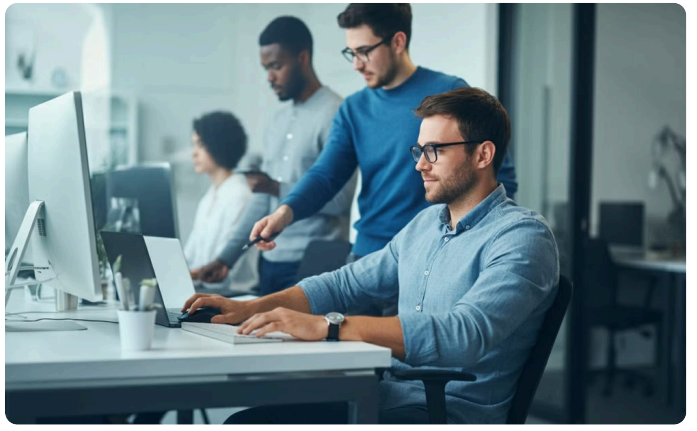
Understanding Check

Your Scrum Team has just finished coding a new feature. A team member suggests sending it to the company's QA department for testing before the Sprint Review. What would be the most appropriate response from a Scrum perspective?

A) Agree, as specialized QA teams can provide more thorough testing.
B) Disagree, explaining that testing is part of the team's end-to-end responsibility.
C) Suggest inviting the QA team to join the Scrum Team for this Sprint.
D) Ask the Product Owner to decide whether to involve the QA team.

Answer and Explanation

The correct answer is B. In Scrum, the team is responsible for all aspects of feature development, including testing and quality assurance. Sending the feature to a separate QA department goes against the principle of end-to-end responsibility and could introduce delays and handoff issues.



Team Responsibility

Team responsible for all aspects, including testing

Separate QA

Separate QA contradicts end-to-end responsibility

Handoff Issues

Could introduce delays and handoff issues

Remember

More than Efficiency

End-to-end development creates ownership and fosters skill development.

Comprehensive Understanding

Ensures comprehensive understanding of features.

Higher Quality

Leads to higher quality products and engaged team.

Next Steps

1

Explore Management

Manage comprehensive responsibilities

2

Maintain Focus

Stay productive while adapting

3

Continuous Improvement

Identify areas for growth

Sustainable, Self-Managed Teams in Scrum

by Mayko Silva

Key Quote from Scrum Guide

> "They are structured and empowered by the organization to manage their own work. Working in Sprints at a sustainable pace improves the Scrum Team's focus and consistency."

- **Organizational empowerment:** Teams are structured and empowered to manage their own work
- **Sustainable pace:** Working in Sprints at a sustainable pace
- **Improved outcomes:** Enhances the Scrum Team's focus and consistency

Organizational Empowerment

Trust and Empower

Organization must trust and empower Scrum Team

Minimal Interference

Minimal interference from outside the team

Authority to Manage

Team has authority to manage their work

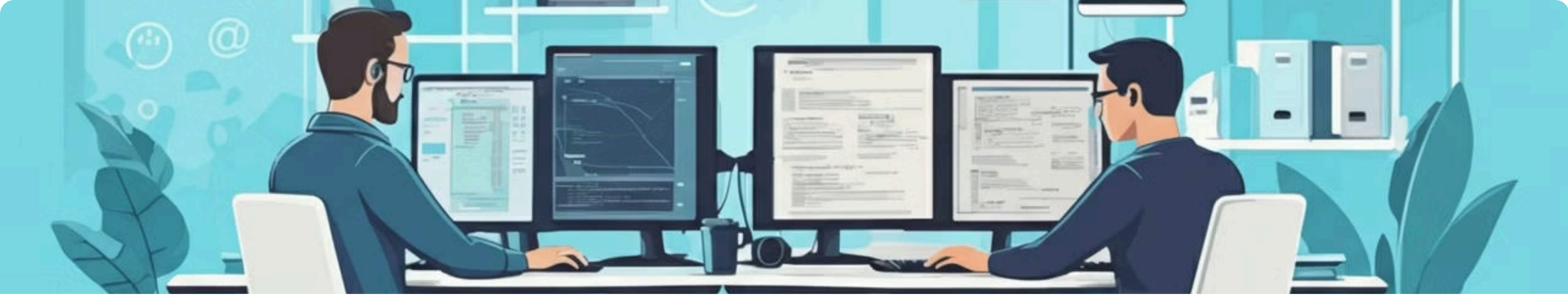
Self-Management

Team Decision-Making

Team decides how to best accomplish their work

Authority

Authority to make decisions about processes and practices



Sustainable Pace

Indefinite Maintenance

Team works at a pace they can maintain indefinitely

Burnout Prevention

Prevents burnout

Long-term Benefits

Ensures long-term productivity and quality

Focus and Consistency

Sustainable Pace

Sustainable pace maintains focus

Allows consistent delivery over time

Preventing Burnout

Prevents peaks and troughs of overwork and recovery

Practical Example: E-commerce Platform Team

In this practical example of a sustainable, self-managed Scrum team working on an e-commerce platform, we observe the following key characteristics:

- Full authority to implement features and choose technologies
- Sets own working hours while meeting commitments
- Resists pressure for overtime
- Establishes consistent velocity over time

These attributes demonstrate how the team embodies the principles of self-management and sustainability in their daily operations.

Understanding Check

The management of your organization is concerned that the Scrum Team isn't working long enough hours. They want to implement mandatory overtime to increase productivity. As a Scrum Master, what would be your most appropriate response?

- A) Agree to implement overtime
- B) Explain sustainable pace benefits
- C) Suggest overtime only during critical periods
- D) Ask Product Owner to decide on overtime

Answer and Explanation

The correct answer is B. Let's break down why this is the most appropriate response:

First and foremost, it's crucial to educate the team about the importance of maintaining a sustainable pace. This aligns with one of the core principles of Scrum and agile methodologies. A sustainable pace ensures long-term productivity and prevents burnout, which is essential for the team's overall health and effectiveness.

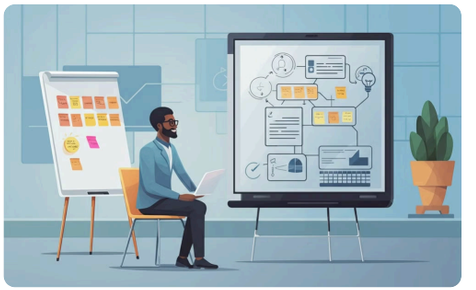
Secondly, the suggestion of mandatory overtime directly contradicts the principles of self-management and sustainability in Scrum. Self-managed teams should have the autonomy to decide how they work and organize their time. Imposing mandatory overtime goes against this principle and can lead to decreased morale and productivity in the long run.

Lastly, the focus should be on long-term productivity and team health rather than short-term gains through overwork. This approach ensures that the team can consistently deliver value over time without sacrificing their well-being or the quality of their work.

Benefits of Sustainable, Self-Managed Teams

Higher motivation and job satisfaction

Team members show increased engagement and happiness in their work



Increased creativity and innovation

Self-managed teams foster an environment that encourages new ideas

Better quality work

Empowered teams take pride in their output, leading to superior results

Consistent, predictable delivery

Self-managed teams maintain a steady pace, ensuring reliable outcomes

Long-term productivity and team stability

Sustainable practices lead to enduring team success and member retention

Remember

Essential in Scrum

Sustainable, self-managed teams are essential in Scrum

Trust and Empowerment

Trust teams to manage themselves

Sustainable Pace

Work at a sustainable pace indefinitely

High Performance

Key to high-performing, motivated teams

Consistent Value

Delivers value consistently over time

Next Steps

1. **Explore**
   Scrum Master's role
2. **Foster**
   Sustainability
3. **Promote**
   Self-management

Exploring how Scrum Masters foster sustainability and self-management

Increments Must Be Valuable and Useful

Incremental improvements should provide tangible value. Avoid changes that are insignificant or unnecessary.

by Mayko Silva

Key Quote from Scrum Guide

"The entire Scrum Team is accountable for creating a valuable, useful Increment every Sprint."

Every Sprint Produces an Increment

1

Tangible Piece

Each Sprint results in a usable part of the product

2

No Setup Sprint

No "Sprint Zero" in Scrum, just productive Sprints

The Increment Must Be Valuable

Contributes to the Overall Product Goal

Increments align with and advance the product vision

Even Small Improvements Can Be Valuable

Increments provide continuous value, no matter the size

The Increment Must Be Useful

Usable Functionality

Can potentially be used by stakeholders

More Than Technical Work

Purely technical work without usable functionality doesn't count

Whole Team Accountability

Team Responsibility

Not just Developers' task

Entire Scrum Team

Accountable for the Increment

Includes PO and SM

Everyone is responsible

Practical Example: Mobile Banking App

Sprint 1

Implement user login functionality

Sprint 2

Add feature to view account balance

Sprint 3

Implement ability to transfer funds

Each Increment is valuable and useful for the mobile banking app.

Understanding Check

Your Scrum Team is working on a complex software project. A team member suggests having a "research Sprint" where no new features are developed, but the team investigates new technologies. As a Scrum Master, what would be your most appropriate response?

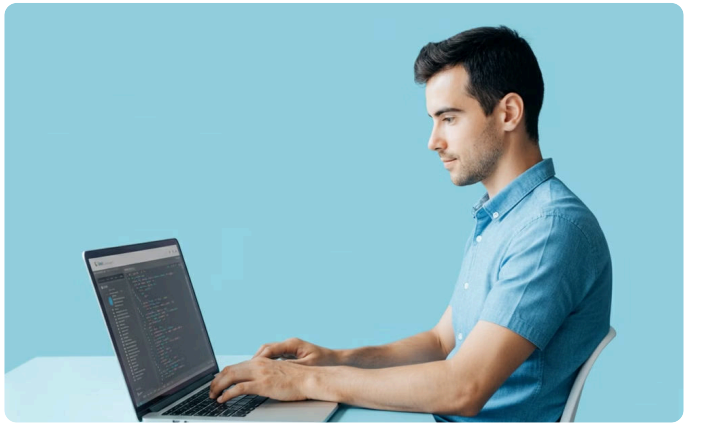
A) Agree to the research Sprint, as learning new technologies is valuable. B) Suggest incorporating the research into regular Sprints while still delivering an Increment.
C) Ask the Product Owner to decide if a research Sprint is acceptable.
D) Reject the idea, stating that every Sprint must produce new features.

Answer and Explanation

The correct answer is B. While research is important, Scrum requires that every Sprint produces a valuable and useful Increment. The team should find ways to incorporate necessary research while still delivering functionality.

Research is Important

But every Sprint must produce Increment



Find Ways to Incorporate Research

While delivering valuable functionality

Whole Team Accountability

Ensure Increments are valuable and useful

Purpose of Valuable and Useful Increments

Stakeholder Feedback

Provides frequent opportunities for stakeholder feedback

Progress Towards Goal

Ensures progress towards Product Goal

Team Motivation

Maintains team motivation with tangible results

Empirical Process

Supports empirical process control with concrete evidence

Remember

Tangible Increment

Every Sprint should result in something real and usable.

Moves Product Forward

The Increment should advance the product or service.

Usable by Stakeholders

The Increment could potentially be used by stakeholders.

Next Steps

Definition of Done

Ensure increments meet standards

Team Collaboration

Work together to deliver value

Continuous Improvement

Identify ways to enhance process

The next steps involve exploring how teams can ensure that each increment they deliver meets the Definition of Done, fostering strong collaboration to deliver valuable and useful work, and continuously improving their processes to enhance the overall Scrum experience.

Scrum Accountabilities

by Mayko Silva

Key Quote from Scrum Guide

"Scrum defines three specific accountabilities within the Scrum Team: the Developers, the Product Owner, and the Scrum Master."

Accountabilities, Not Roles

Scrum Terminology

Scrum uses "accountability" rather than "role" or "job"

Emphasizes responsibilities and expectations

Flexibility

Not rigid job descriptions

Three Specific Accountabilities

Developers

The team members responsible for creating the product increment

Product Owner

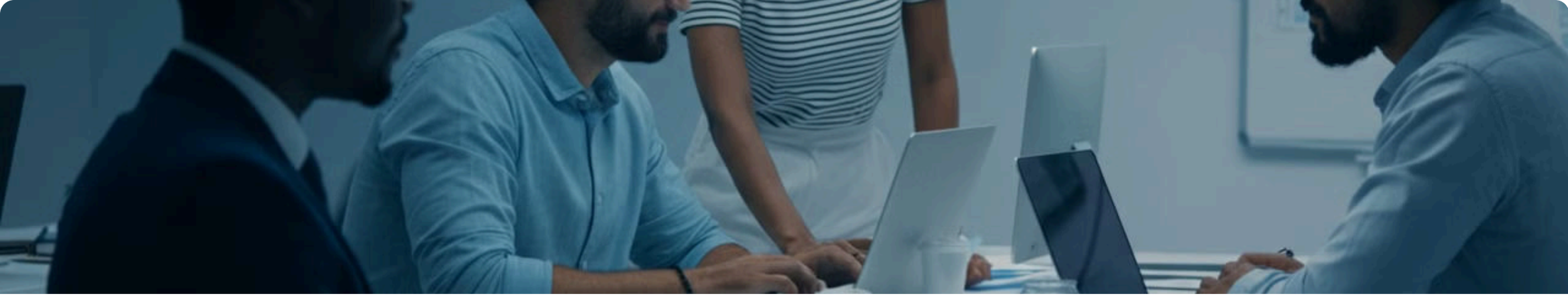
Accountable for maximizing the value of the product resulting from the work of the Scrum Team

Scrum Master

Responsible for establishing Scrum as defined in the Scrum Guide

Flexibility Within Accountabilities

- How accountabilities are fulfilled can vary
- Depends on organization and team



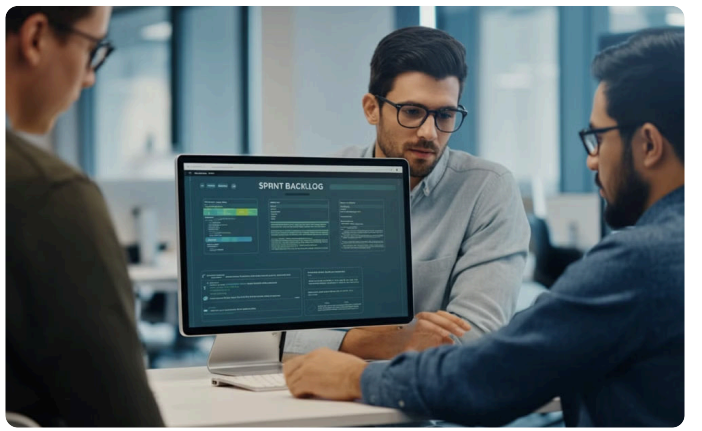
Whole Team Responsibility

- **Despite specific accountabilities**, each team member has unique responsibilities
- **Entire Scrum Team collectively responsible** for creating value through their combined efforts

Developers

Create the Increment

Developers are responsible for creating the Increment each Sprint, ensuring tangible progress.



Manage Sprint Backlog

Developers are responsible for the Sprint Backlog, organizing and prioritizing tasks.

Adhere to Definition of Done

Developers must adhere to the Definition of Done, ensuring quality and completeness.

Product Owner Responsibilities

Maximize Value

Maximizes product value

Manage Backlog

Manages the Product Backlog

Scrum Master Responsibilities

Establishing Scrum

Establishes Scrum as defined in Scrum Guide

Coaching

Coaches team and organization in Scrum adoption

Practical Example: Mobile Banking App Team

In a mobile banking app development team, we can see how the Scrum accountabilities play out in practice:

- Developers: Responsible for coding, UI design, and testing features of the mobile banking app
- Product Owner: Focuses on prioritizing features based on user feedback and industry trends
- Scrum Master: Facilitates Scrum events, coaches the team, and removes obstacles to ensure smooth development process

Understanding Check

In a Scrum Team, who is accountable for maximizing the value of the work done by the team?

A) Scrum Master

B) Developers

C) Product Owner

D) Entire Scrum Team collectively

Answer and Explanation

The correct answer is C. This reflects the key principles of Scrum accountability and teamwork:

- The Product Owner is accountable for maximizing the value of the work
- The entire team collaborates to achieve this goal

This answer highlights the balance between individual accountability and collective responsibility in Scrum. While the Product Owner has a specific accountability, the success of the project relies on the collaborative efforts of the whole team.

Remember

Clarity on Responsibilities

Accountabilities provide clarity on responsibilities

Collaborative Nature

Not about creating hierarchies or silos

Emphasizes collaborative nature of Scrum

Team Effort

Team works together, each fulfilling their accountabilities

Goal

Goal: Create valuable product increments

Next Steps

1

Current Progress

We've covered the overview of Scrum Accountabilities

2

Diving Deeper

Our next focus is on diving deeper into each accountability

3

Starting with Developers

In the next session, we'll begin by exploring the Developers' accountability in detail

Balancing Terminology and Accessibility in Scrum

by Mayko Silva

The Importance of Terminology

Specific Terms in Scrum

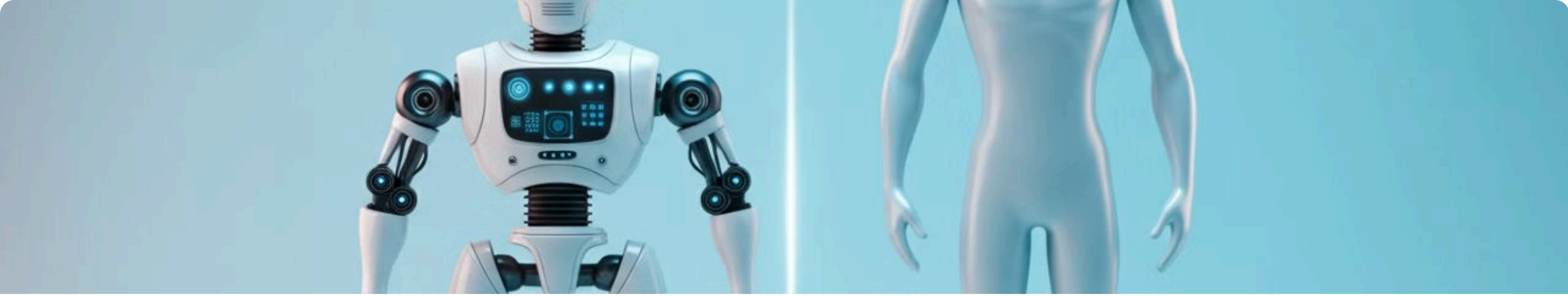
Scrum uses specific terms (e.g., "accountabilities" vs "roles")

Precise Meanings

Terms chosen to convey precise meanings

Avoiding Misunderstandings

Correct terminology helps avoid misunderstandings



The Risk of Rigidity

Overfocus on Wording

Some practitioners become overly focused on exact wording

Gatekeeping

Can lead to gatekeeping and alienating newcomers

Spirit over Letter

Spirit of Scrum more important than the letter

Balancing Act

1

Precision vs. Accessibility

Need to balance precision and accessibility in Scrum terminology

2

Goal

Make Scrum understandable while maintaining its essence

3

Achieving Balance

Strive for clarity without compromising the core principles of Scrum

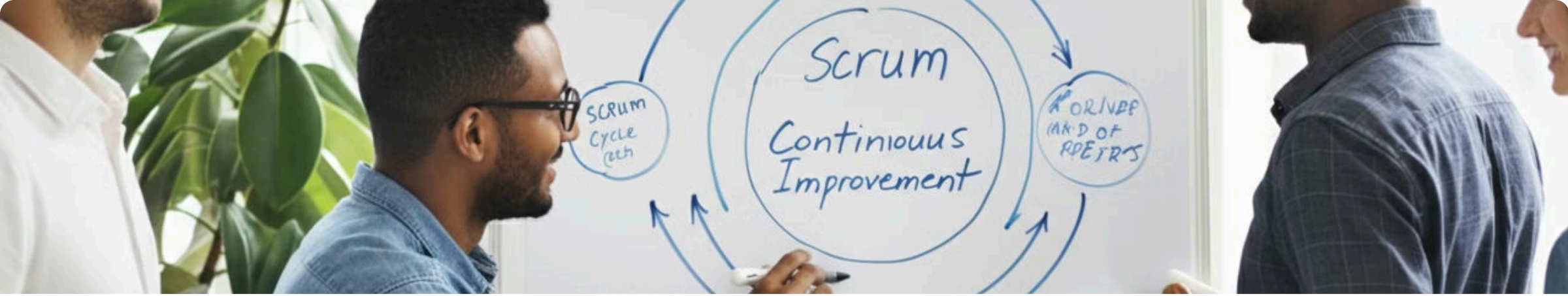
Context Matters

Formal Settings

In formal settings and certification exams, precise terminology is crucial. This ensures standardization and accuracy in understanding Scrum concepts.

Everyday Practice

When introducing Scrum or in everyday practice, more colloquial language might be appropriate. This allows for easier understanding and adoption of Scrum principles.



The Spirit of Scrum

- **Core focus:** Empiricism, continuous improvement, and delivering value
- **Potential distraction:** Getting hung up on terminology can detract from core principles

Practical Example

Introducing Scrum to a new team:

> "In Scrum, we have three main roles... sorry, I mean accountabilities: the Developers, the Product Owner, and the Scrum Master."

- Acknowledges correct terminology
- Uses familiar term to aid understanding

Understanding Check

Let's test your understanding with a scenario:

You're discussing Scrum with a colleague who's new to the framework. They refer to the Product Owner as a "role". What's the most appropriate response?

- A) Correct them immediately
- B) Ignore terminology and continue
- C) Acknowledge point, gently introduce "accountability"
- D) Tell them to study Scrum Guide more

Answer and Explanation

Correct answer: C

- Respects current understanding
- Introduces correct terminology non-confrontationally
- Balances accuracy with accessibility

Key Takeaways

Scrum's Core Purpose

Scrum's goal: Help teams deliver value effectively

Terminology Usage

Use correct terminology, especially in formal settings

Accessibility

Make Scrum accessible and understandable

Communication Priority

Effective communication and improvement more important than perfect terminology

Focus Areas

Focus on Scrum principles and values

Language Approach

Use language that helps others understand and apply concepts

Next Steps

1. **Apply**
   Implement balanced approach
2. **Analyze**
   Evaluate effectiveness
3. **Explore**
   Various Scrum events

Exploring balanced approach in various Scrum events