

# Course Overview



## Why Agile?

- The Agile Mindset
- The Agile Manifesto
- Agile vs. Waterfall

## Agile Teams

- High Performing Teams
- Size, Structure and Skills
- Agile Governance

## Agile Frameworks

- Why Frameworks?
- Scrum
- Kanban
- XP

## Learning Objectives

---

By the End of the Course You Will Be Able To...

- Identify and apply the characteristics of Agile to implement value-driven delivery
- Determine the appropriate Agile framework for a specific business use case and team
- Apply appropriate Agile practices for the specific Agile framework being used
- Build and evolve a high-performing Agile team
- Differentiate between team roles and responsibilities of a typical Agile team
- Assign team members to an appropriate role.

# Speaking Agile

## Key Terms to know

- Backlog
- Velocity
- Muda
- NVA
- Delivery
- Flow
- Strategic Roadmap
- Sprint
- Retrospective





# History of Agile and Agile Frameworks

## History of Agile



- Emerged organically as a result of a market need
- In the 1990s, it was taking 2-4 years to deliver solutions

## Traditional Process Not Meeting Business Needs

- Delay was not acceptable for business users
- Change was inevitable but hard to achieve using traditional development models



**There Must Be A Better Way!**

Focus on

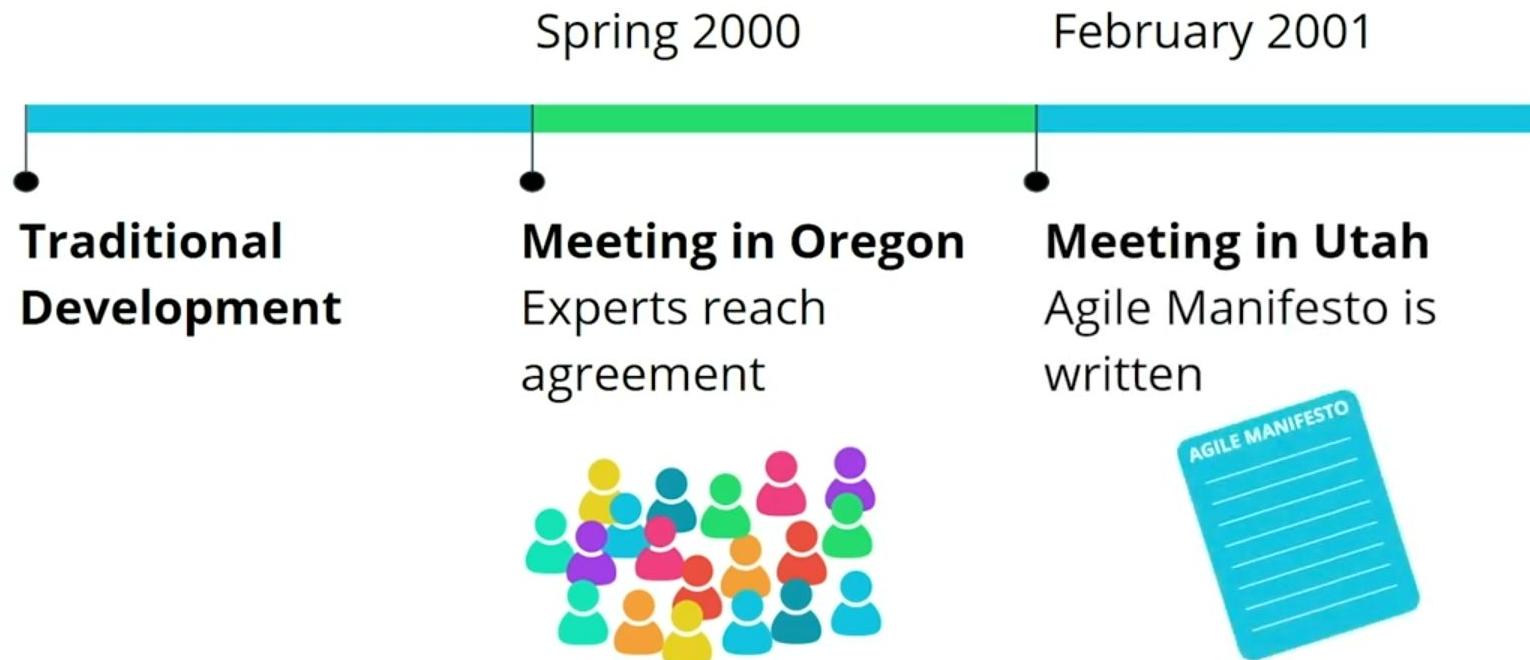


**Business Users**



**Business Value**

# History of Agile



## The Agile Approach to Fast Delivery

- Quickly build working solutions
- Get solutions to end users
- Users get benefits of solution quicker
- Development team gets rapid feedback



## Agile Beyond Software

---



*Agile is only valuable in the software development world*

- Agile Principles are being applied in **all** industries across the globe
- Razor-sharp focus on business value is the secret to Agile's miraculous success.

## Agile Tools

---



**Microsoft Teams**

**Kanbanize**

**VersionOne**

.....and many others

**Trello**

**Jira**

**Confluence**

**Slack**

**Rally**

## Agile Tools

---



- Excellent communication and collaboration resources
- Track progress, iteratively assign work, track defects, etc.
- Monitor team progress

## Agile Tools

---

- Customizable to accommodate Agile Teams
  - In various industries
  - With various Agile Frameworks



## Agile Tools



- Make cross-functional collaboration easier
- Provide real-time dashboards that show progress
- Helps estimate capacity for future iterations

# Project: Launch Agile Transformation for WorldVisitz

Launch An Agile Transformation



- Travel Agency
- Planning to launch a mobile application
- Hiring YOU as an Agile Consultant
- Define an Agile solution and transform traditional processes

Prepare two presentations:

---



Executive Agile  
Transformation



Agile Team  
Transformation

# Executive Agile Transformation Presentation



- How WorldVisitz Can Benefit From Agile
- The Optimal Agile Framework for WorldVisitz

# Agile Team Transformation Presentation



- Onboard the Team
- Explain Agile benefits
- Set the team up for success

# Lesson Overview



## Why Agile?

- The Agile Mindset
- The Agile Manifesto
- Agile vs. Waterfall
- Misconceptions of Agile

## Agile Teams

## Agile Frameworks

## Learning Objectives

---

By the End of the Lesson You Will Be Able To...

- Explain the **Agile Mindset** and value-driven delivery benefits of Agile
- Differentiate between Doing Agile and Being Agile
- Explain The Agile Manifesto's **4 Paired Core Values**
- Identify and apply the 12 Principles of the **Agile Manifesto**
- Differentiate between **Agile** and **Waterfall** approaches

## Why Are Organizations Adopting Agile?



- Volatility
- Uncertainty
- Complexity
- Ambiguity

AGILE



## Faster Time to Market



- Accelerates product development cycles
- Agile teams are able to release products anywhere from 50-80% faster

## Early Return on Investment (ROI)



- Helps squeeze cash out of WIP (Work in Progress)
- Increase project and portfolio returns

## Squeeze Cash Out of WIP

- Invest cash flexibly as product evolves
- Less work on non-value added features
- Better prioritization



## Increase Returns

- Fail fast/stop early
- Move funds to higher value project
- Better alignment with the market



## Feedback From Real Customers



- Focuses the team on the intended goal
- Ensures delivery of high-value features
- Allows the team to accommodate change

<http://theleanstartup.com/>

## **Build The Right Products**

---

- Connects product strategy to execution
- Iterative approach allows for adjustments
- Communication and Collaboration



## Early Risk Reduction



- Risks are managed proactively
- Risk reduction accomplished through iterative delivery
- Transparency ensures risks are managed early

## Built-in Quality

---

- Business benefits justify the investment in quality
- Elevated customer satisfaction
- Ability to innovate and scale confidently



## Culture and Morale



- Open communication, collaboration and cohesiveness
- Productive sustainable pace
- Motivation to find opportunities that benefit the business

## Efficiency vs. Effectiveness

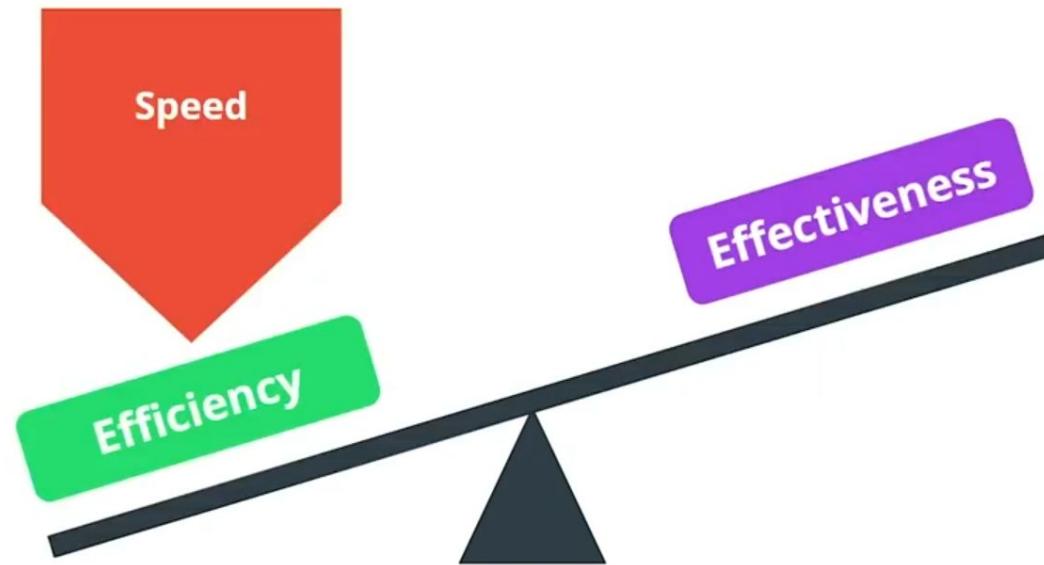
---

Efficiency

Effectiveness

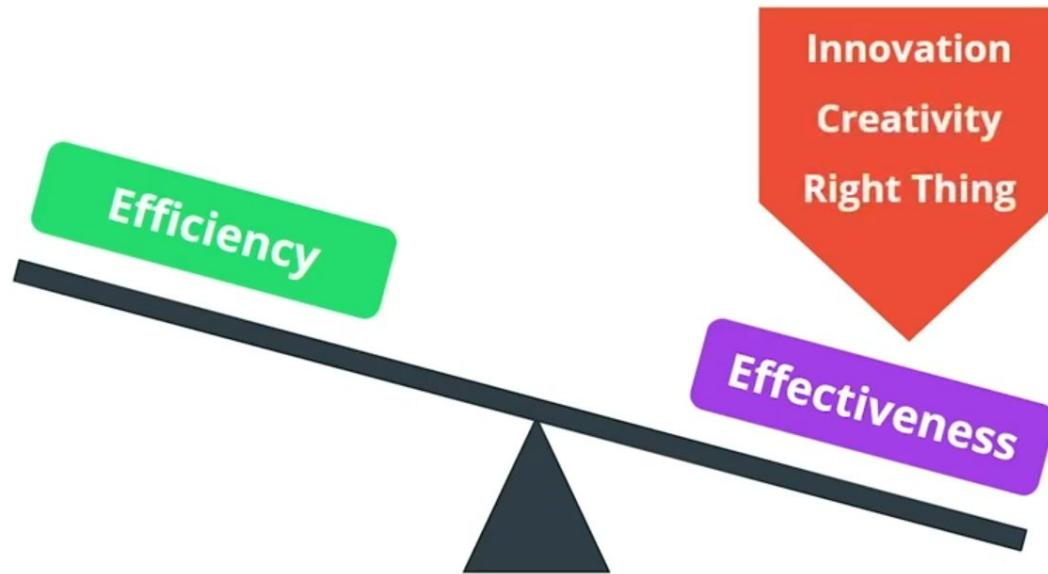


## Efficiency vs. Effectiveness



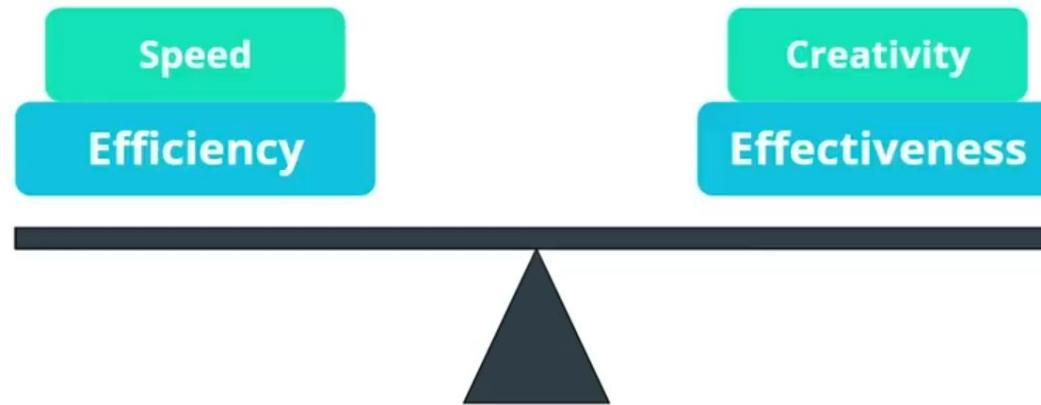
## Efficiency vs. Effectiveness

---



## Efficiency vs. Effectiveness

---



## Customer Satisfaction



- A key Agile principle
- Value driven high quality product features and functionality
- Customer satisfaction is ensured through Product Owner

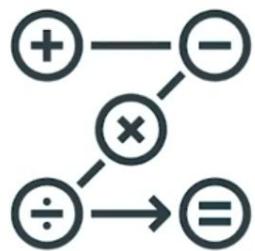
## Alignment

---

- Keep the **Vision** in focus
- Helps the business build a decision making framework for prioritization
- Sets clear guidelines for incremental product reviews



## Emergent Outcomes



- Works well in developing complex dynamic systems
- Valuable results can emerge
- Agile can accommodate such uncertainty

## Predictability

---

- Low risk and highly predictable approach
- Adaptive planning
- Ensures structure within the '**chaos**' of complexity



## Why Companies Are Switching



1. Faster time to market
2. Early ROI
3. Feedback from real customers
4. Build the right products
5. Early risk reduction
6. Better quality
7. Culture and morale
8. Efficiency
9. Customer satisfaction
10. Alignment
11. Emergent outcomes
12. Predictability

## How Experts Approach Agile



## How Experts Approach Agile



Systems  
Thinking



Growth  
Mindset



Optimal  
Business Value

## Traditional Thinking

---

Looks at the parts separately



## System Thinking

---

Looks at interdependent and connected components of a system

Considers how systems interrelate and influence each other

How components come to deliver a solution together



## Systems Thinking

*Components of a system must come together to deliver a cohesive solution*

- Team understands that optimizing a component does not optimize the entire system
- Demos help team visualize the actual user experiences



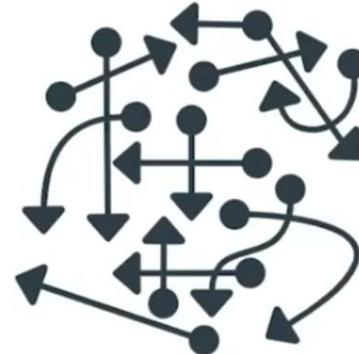
## Growth Mindset

*Abilities can be developed through dedication and hard work*

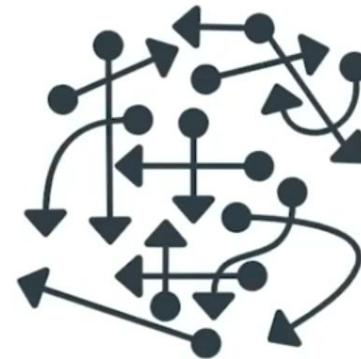


- Promotes resilience and passion for learning
- Balanced with ethics, principles and values
- integrity, commitment and respect are **not** compromised for speed

## The Agile Mindset



## The Agile Mindset



## The Agile Mindset



# VUCA

---



**VOLATILITY**



**UNCERTAINTY**



**COMPLEXITY**



**AMBIGUITY**

## VUCA

---



**VOLATILITY**

Unexpected or unstable possibilities



**UNCERTAINTY**



**COMPLEXITY**



**AMBIGUITY**

## VUCA

---



**VOLATILITY**



**UNCERTAINTY**



**COMPLEXITY**



**AMBIGUITY**

Unknown future

## VUCA

---



**VOLATILITY**



**UNCERTAINTY**



**COMPLEXITY**



**AMBIGUITY**

Interconnected parts

## VUCA

---



**VOLATILITY**



**UNCERTAINTY**



**COMPLEXITY**



**AMBIGUITY**

Lack of clarity

## What is Agile?

---

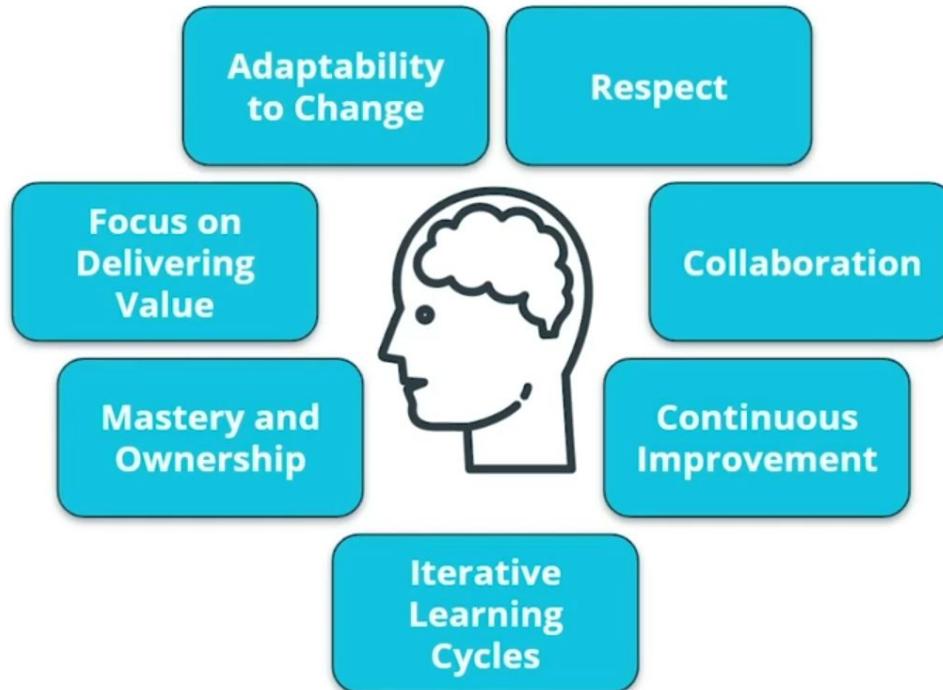
*A set of value and principles  
that guide how we as individuals and  
organizations think about and  
act upon the work that we do*

## What Is Business Agility?



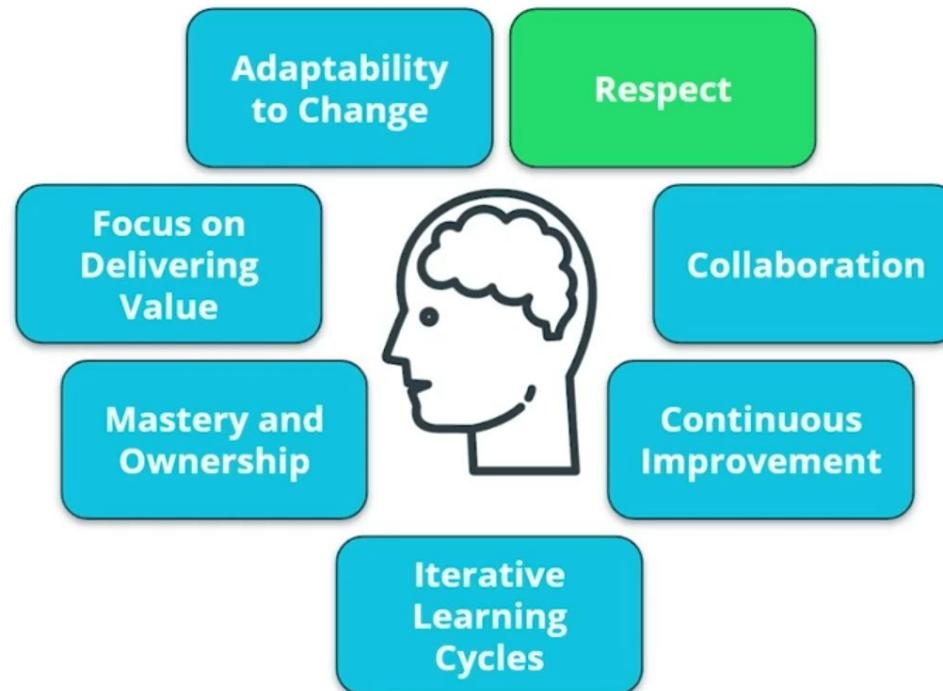
- Adapt to market changes
- Respond to customer demands
- Cost-effective without compromising quality

## The Agile Mindset



*The set of attitudes supporting an Agile working environment*

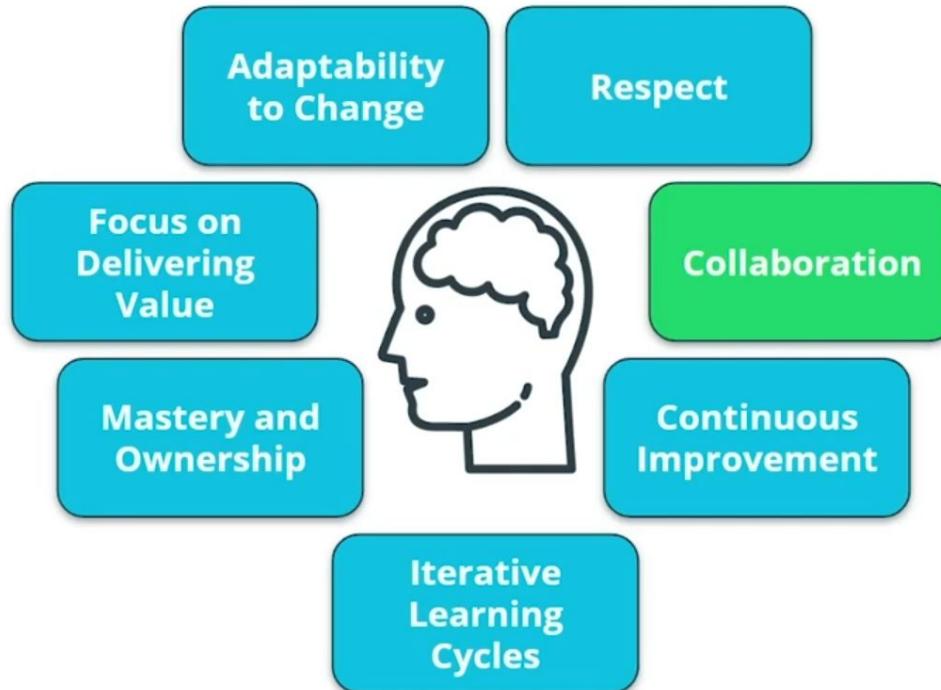
# The Agile Mindset



## Respect

- All levels of the organization
- Customers
- Backlog

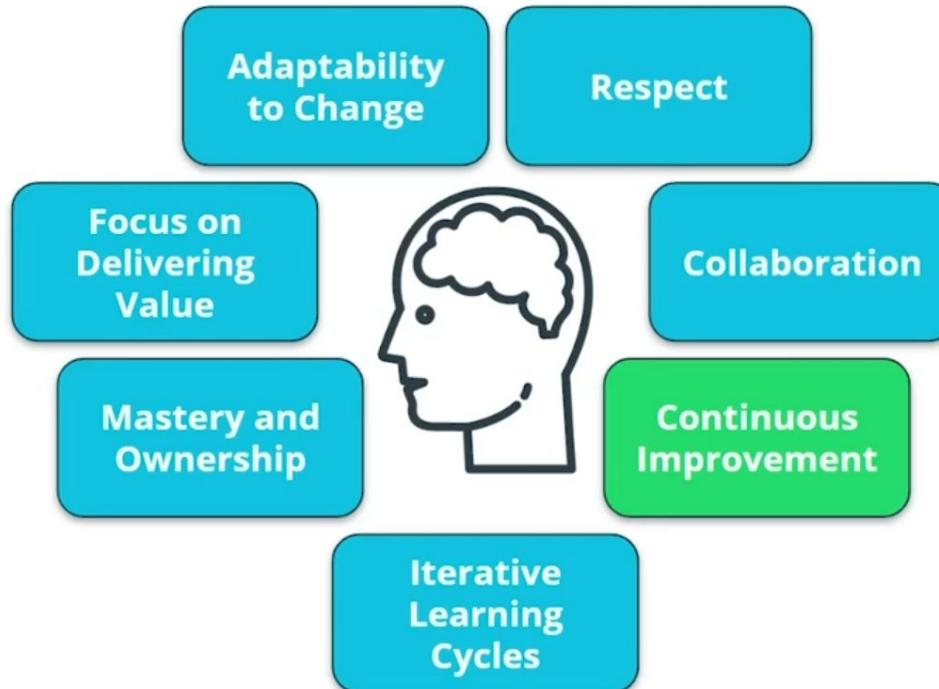
# The Agile Mindset



## Collaboration

- Cross-functional, skilled teams are critical
- Collaboration improves the efficiency of handoffs

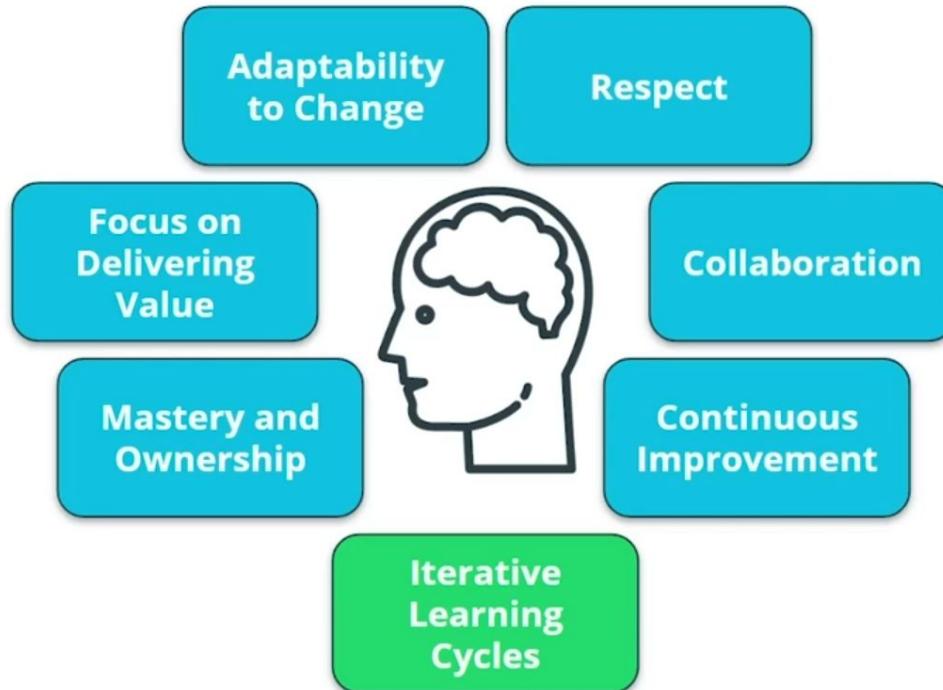
# The Agile Mindset



## Continuous Improvement

- Focus on quality is habitual
- Always room for improvement

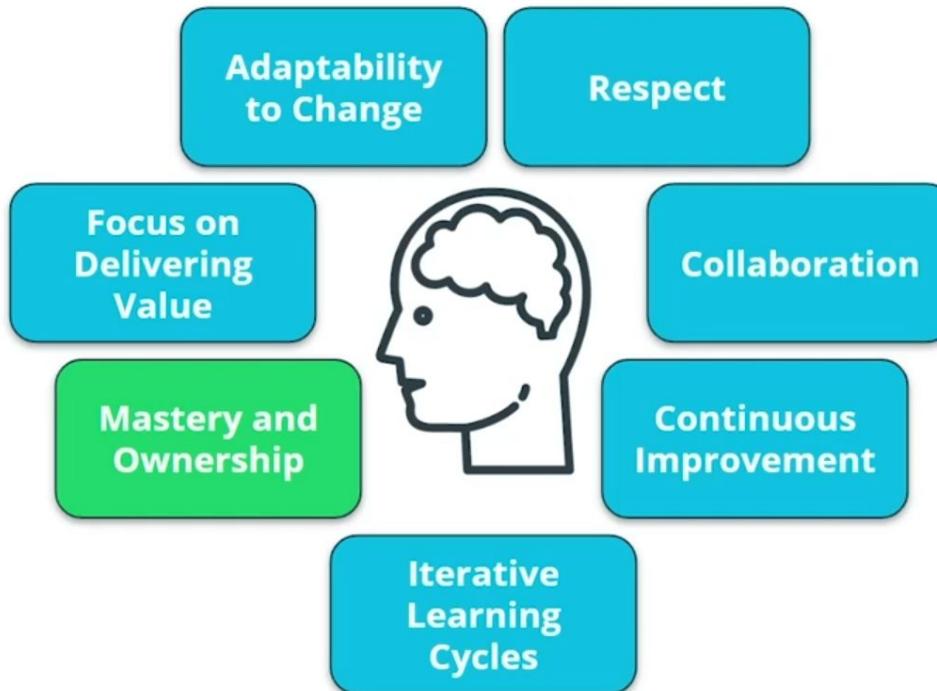
# The Agile Mindset



## Iterative Learning Cycles

- Use a growth mindset
- Turns failures into opportunities for learning
- Abilities evolve over time

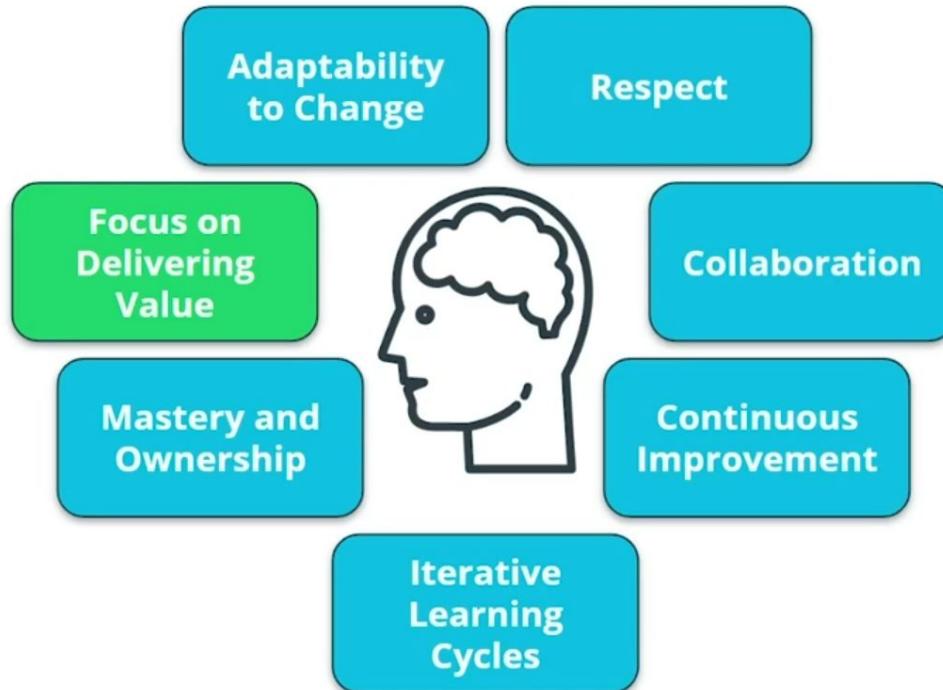
# The Agile Mindset



## Mastery and Ownership

- Increases desire to deliver high quality work

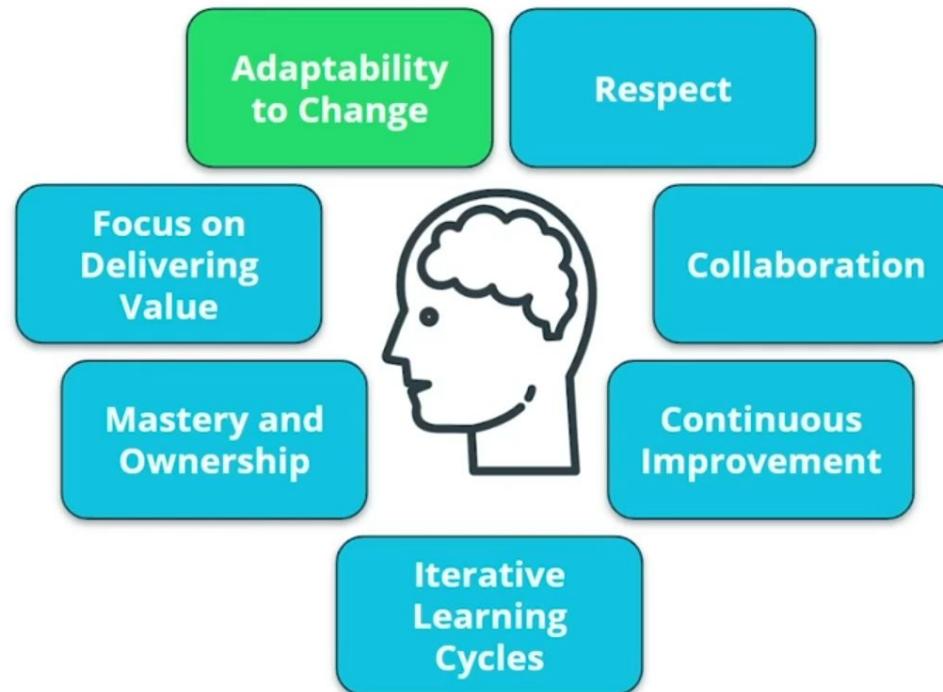
# The Agile Mindset



## Focus on Delivering Value

- Maintain a customer satisfaction-centric culture
- Customer's needs are paramount

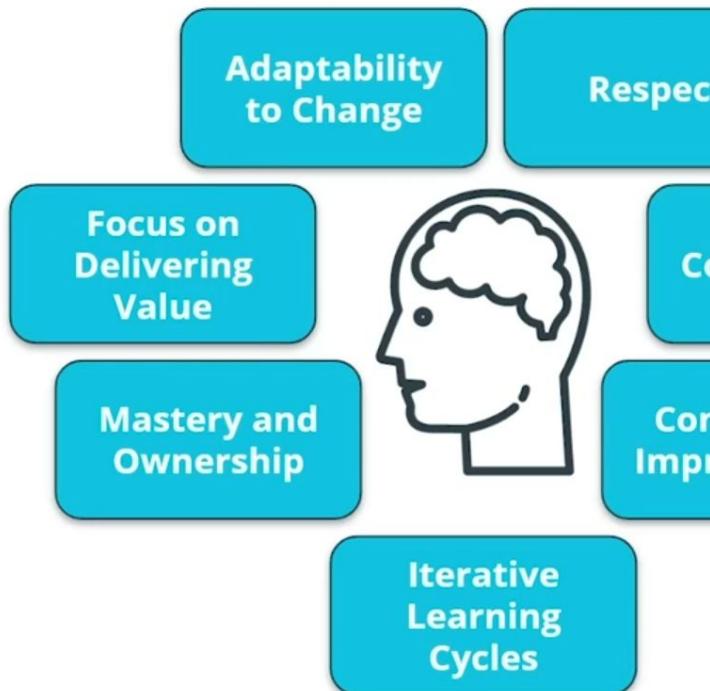
# The Agile Mindset



## Adaptability to Change

- Processes should not be impediments to change

## The Agile Mindset



Value for  
customers  
and business  
owners

## Doing Agile

---

- Adopting practices
- No commitment to principles and values



## Being Agile

---

- Agile Mindset
- Agile Manifesto
- Agile Principles



## Agile Teams

---

- Are aligned with vision
- Have autonomy to self-direct
- Cultivate mastery



## Agile Mindset Exercise

---

- Identify three anti-patterns
- Explain why each anti-pattern is problematic



## Anti-Pattern 1: Environment

Team was not self-directed and self-organized



### Evidence:

- Each team member worked on their own list of tasks
- The Project Manager monitored everyone's progress individually
- PM stepped in to troubleshoot technical problems

### Results:

- Team had no opportunity to build autonomy and confidence
- Team morale was low

## Anti-Pattern 2: Quality

Quality was not built-in



### Evidence:

- Major coding flaws
- Supporting systems integration testing was patchy

### Results:

- Delays in going to production
- End user frustration
- Issues were identified after going to production

## Anti-Pattern 3: Demos

Working solutions not presented frequently



### Evidence:

- Product team did not get an opportunity to obtain feedback from user groups
- No properly vertically sliced demos were conducted

### Results:

- Incomplete feature set in production
- Complaints from end users after going to production
- Team had to rush in and react

## Anti-Pattern 4: Documentation

Excessive and Unnecessary documentation



### Evidence:

- A lot of process driven paperwork
- Oversight was inadequate
- Team dedicated time to excessive documentation

### Results:

- Time taken away from working on actual product deliverables
- Paper trail was not effective

## Anti-Pattern 5: Vetting of Requirements

Inadequate vetting of requirements



### Evidence:

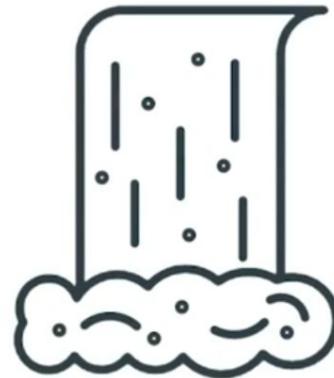
- Key requirements uncovered after going to production
- Several requirements that were built not useful

### Results:

- Resources wasted on low priority requirements
- Higher priority needs were not met

## **Before The Agile Manifesto...**

*Waterfall method based on adherence to strict processes*



## The Waterfall Process



Gathering Requirements

Focus on Development and Customer Team



Products Get More Complex

Features May Not Work as Expected



## Before The Agile Manifesto...



Process Centric  
not Product Centric



The Agile Manifesto

## Four Paired Core Values

**Individuals** and interactions over **processes** and tools

Customer **collaboration** over contract **negotiations**

**Four Paired Core Values**

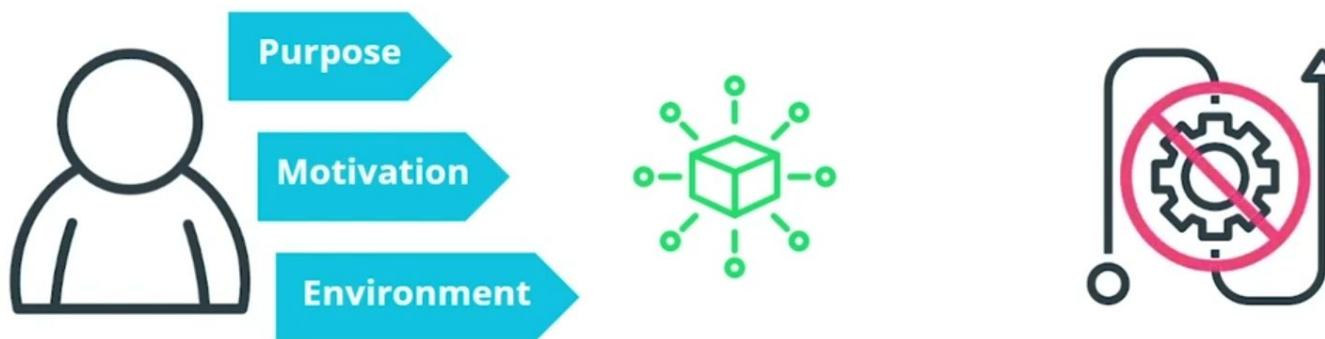
Working **software** over comprehensive **documentation**

Responding to **change** over following a **plan**

Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

## Four Paired Core Values

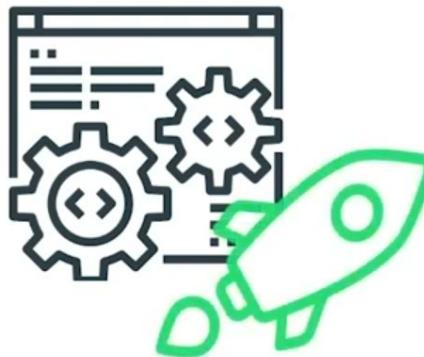
**Individuals and interactions over processes and tools**



Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

## Four Paired Core Values

Working **software** over comprehensive **documentation**

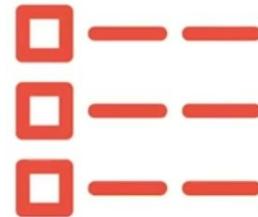


Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

## Four Paired Core Values

---

Customer **collaboration** over contract **negotiations**



## Four Paired Core Values

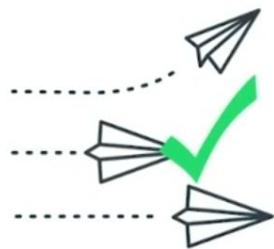
---

Customer **collaboration** over contract **negotiations**



## Four Paired Core Values

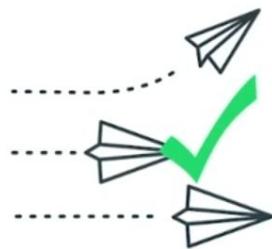
Responding to **change** over following a **plan**



Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

## Four Paired Core Values

Responding to **change** over following a **plan**



Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

# Four Paired Core Values Drive 12 Agile Principles



Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

# The Agile Manifesto Principles

1 Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.



2 Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.



3 Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.



4 Business people and developers must work together daily throughout the project.



Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

# The Agile Manifesto Principles

- 5** Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.  

- 6** The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.  

- 7** Working software is the primary measure of progress.  

- 8** Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.  


Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

# The Agile Manifesto Principles

**9** Continuous attention to technical excellence and good design enhances agility.



**10** Simplicity--the art of maximizing the amount of work not done--is essential.



**11** The best architectures, requirements, and designs emerge from self-organizing teams.



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



Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

# Four Paired Core Values Drive 12 Agile Principles



Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

## Agile Manifesto Exercise

---

- Which two values are absent at SocialKare.gov?
- Which Agile Principles should SocialKare.gov adopt?



# Agile Manifesto: Missing Paired Core Value 1

Individuals and interactions over processes and tools



## Evidence:

- Lack of team cohesion
- Siloed approach
- Autocratic management style

## Results:

- Quality of the results was disappointing
- Demotivated and fatigued team

## Agile Manifesto: Missing Paired Core Value 2

Working software over comprehensive documentation



### Evidence:

- Lots of ineffective planning documents and paper trails
- No demos for a working prototype
- No opportunity for early feedback

### Results:

- Tangible results at the end inadequate
- Feedback on the final deliverables was dismaying

## Agile Manifesto: Missing Principle 1

Simplicity--the art of maximizing the amount of work not done -- is essential



### Evidence:

- Heavy requirements upfront
- Unprioritized requirements

### Result:

- Non-value added (NVA) requirements that were not so useful

## Agile Manifesto: Missing Principle 2

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software



### Evidence:

- No delivery of functionality incrementally
- No periodic product demos

### Result:

- Feedback loop delayed

## Agile Manifesto: Missing Principle 3

Welcome changing requirements, even late in development.  
Agile processes harness change for the competitive advantage



### Evidence:

- Changing requirements at the tail end
- Stringent change control review

### Result:

- Delays in responding to key functional change requests

## Agile Manifesto: Missing Principle 4

Business people and developers must work together daily throughout the project

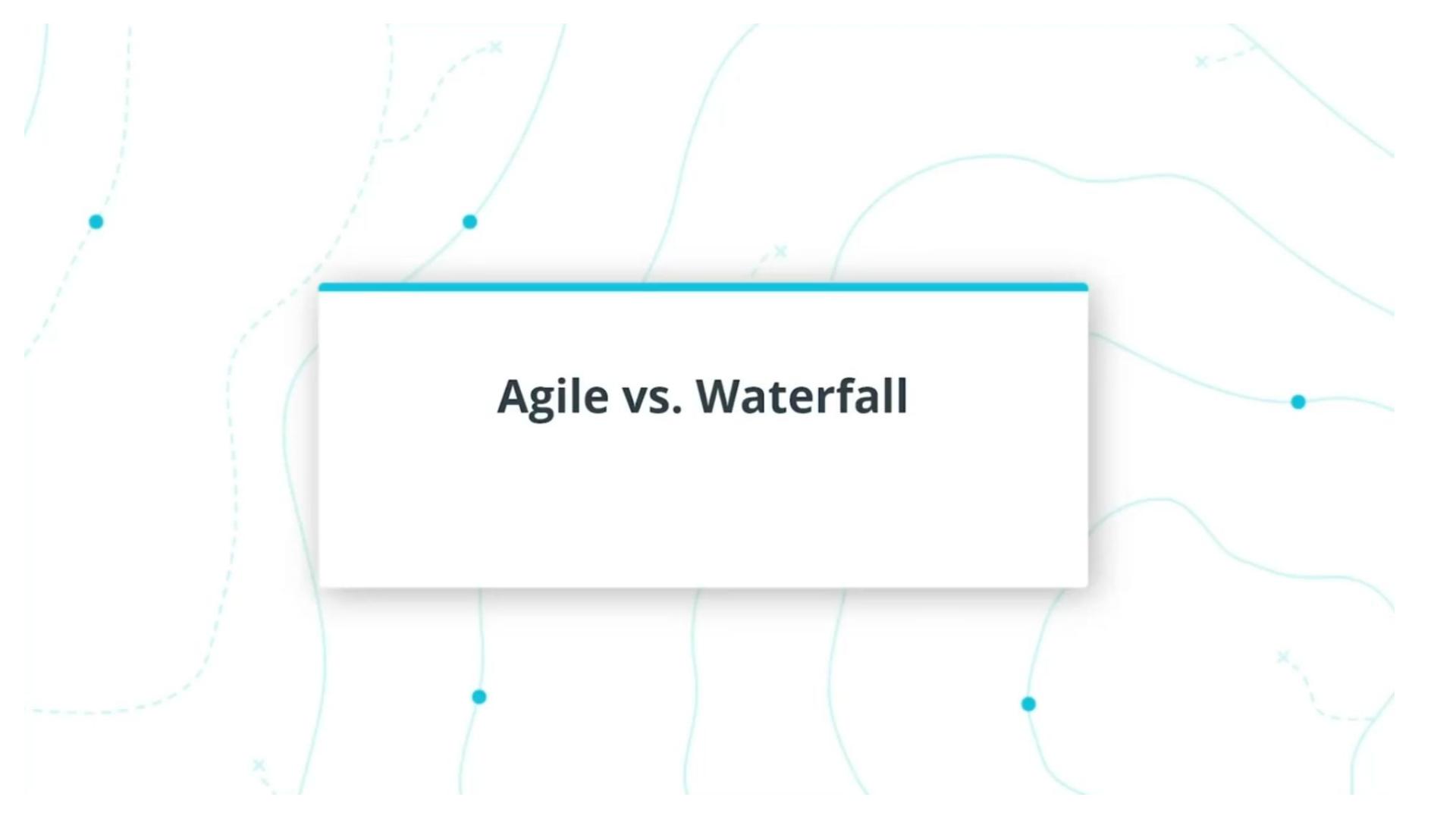


### Evidence:

- Development team did not actively engage business users

### Result:

- Delayed feedback loop
- Lack of confidence



## Agile vs. Waterfall

## The Waterfall Model

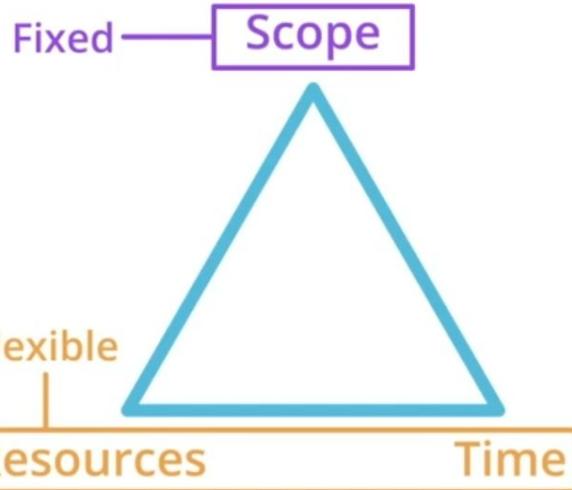
---

- Process-driven
- Each phase depends on deliverables from the previous phase
- Focus on work to be done

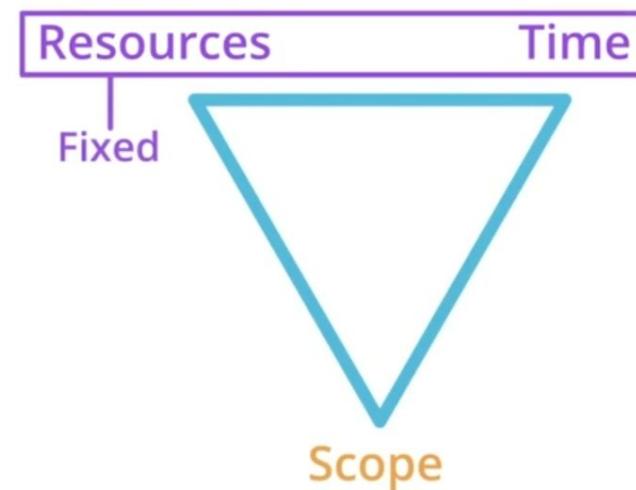


## Waterfall vs. Agile

### Waterfall



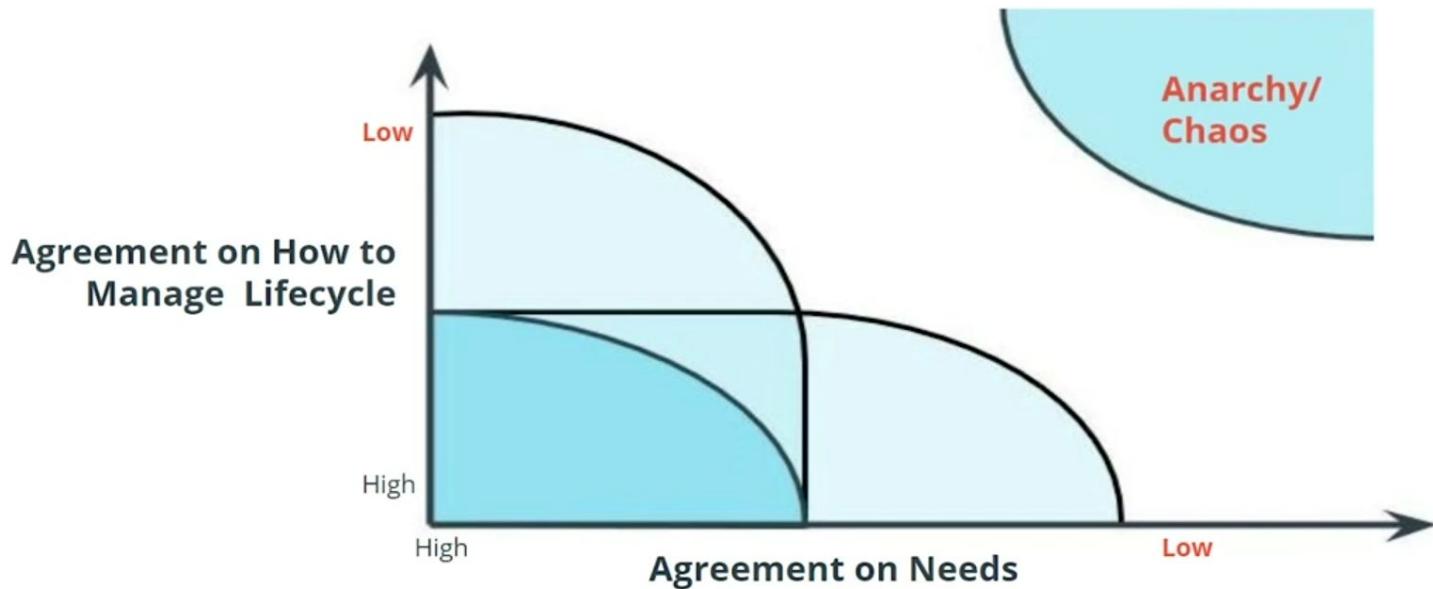
### Agile



## Agile vs. Waterfall

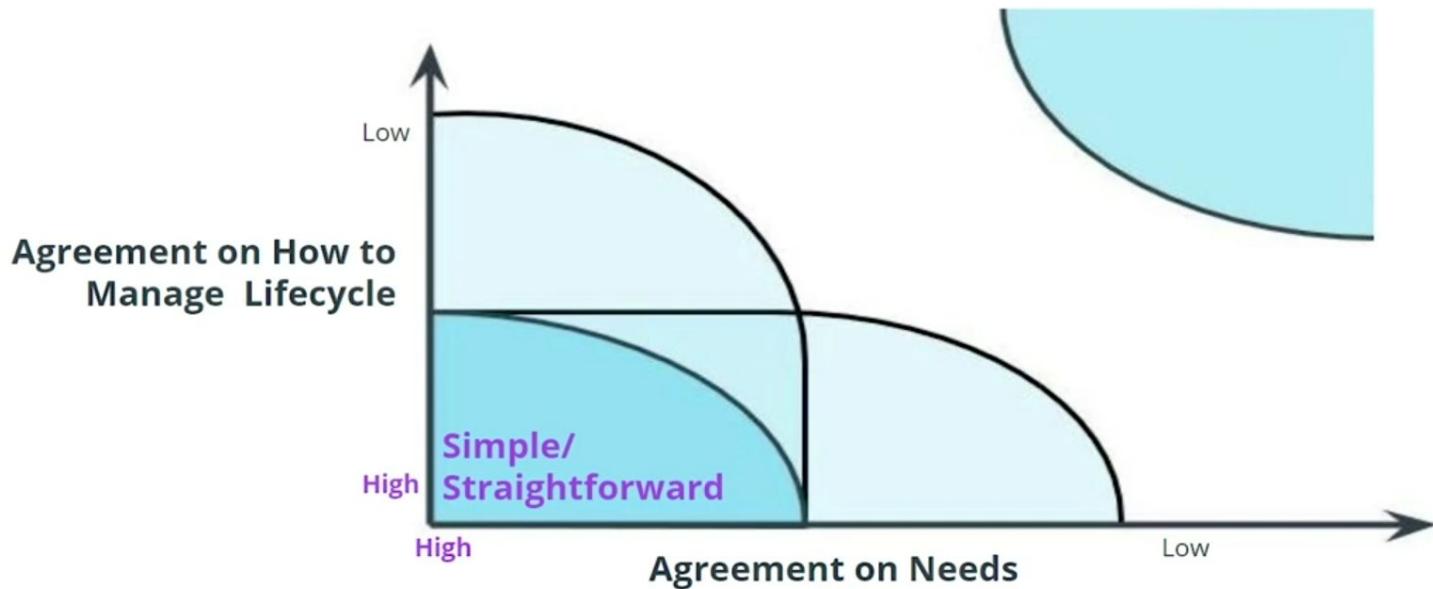
	Agile	Waterfall
Product Lifecycle	Iterative	Distinct phases
Design Process	Incremental	Sequential
Change	Change is embraced	Change is a challenge
Mindset/Focus	Customer needs	Completing the project
Collaboration	Significant collaboration	Limited Synchronization
Requirements	Prepared incrementally	Prepared at start of project
Organization	Self-directed team	Led by Product Manager

## Stacey Diagram



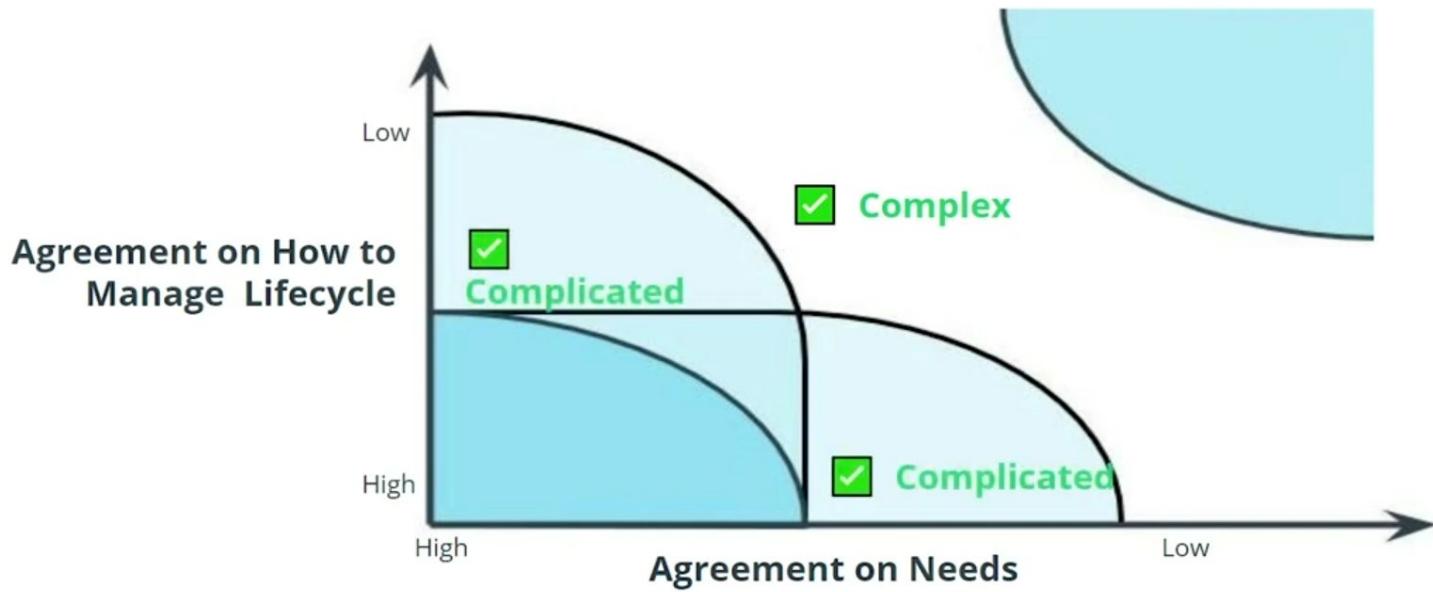
Adapted from: Stacey RD. *Strategic management and organisational dynamics: the challenge of complexity*. 3rd ed. Harlow: Prentice Hall, 2002

## Stacey Diagram



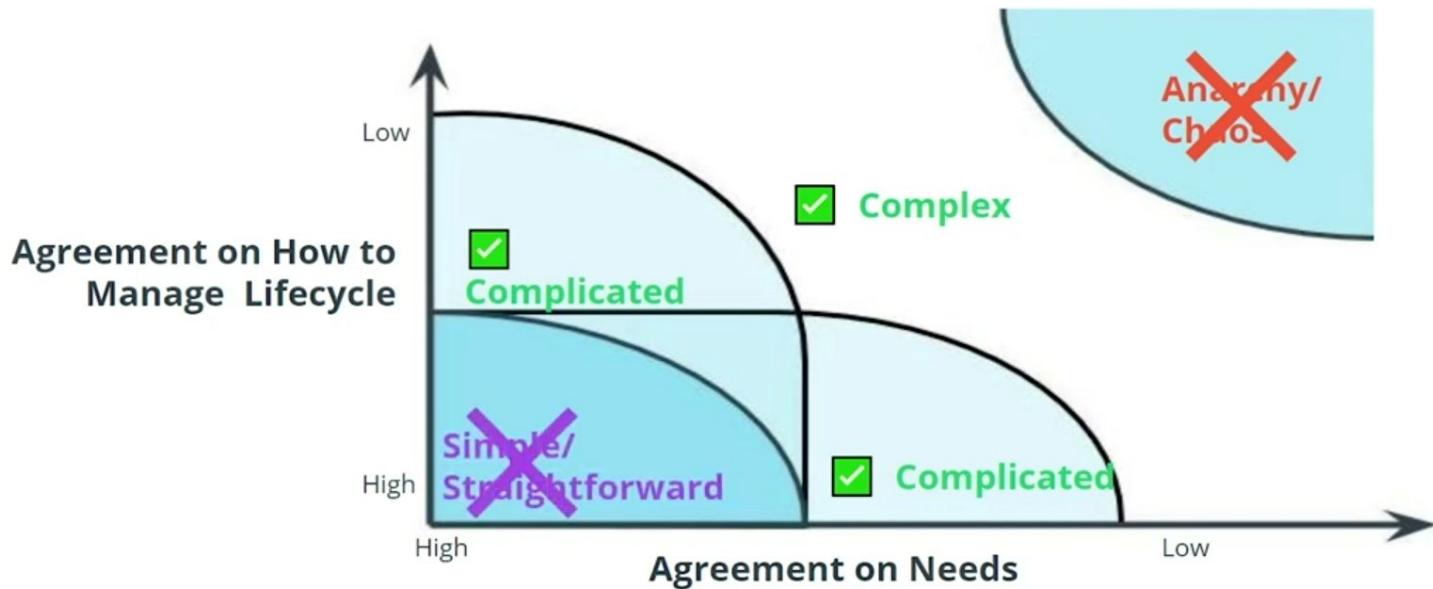
Adapted from: Stacey RD. *Strategic management and organisational dynamics: the challenge of complexity*. 3rd ed. Harlow: Prentice Hall, 2002

## Stacey Diagram



Adapted from: Stacey RD. *Strategic management and organisational dynamics: the challenge of complexity*. 3rd ed. Harlow: Prentice Hall, 2002

## Stacey Diagram



Adapted from: Stacey RD. *Strategic management and organisational dynamics: the challenge of complexity*. 3rd ed. Harlow: Prentice Hall, 2002

## Weaknesses of Waterfall

---

- Identify at least two weaknesses
- Explain why this is a weakness when compared to the Agile approach



## Waterfall Behavior 1

Thorough documentation and paper trails

### Evidence:

- Significant amount of time spent on creating and maintaining documents

### Result:

- Some focus was lost on working solutions
- Time taken away from valuable product development

## Waterfall Behavior 2

The testing process started once development was over



### Evidence:

- All testing was planned for and conducted after full development

### Result:

- Bugs found later in development cycle
- Expensive fixes



## Misconceptions of Agile

## Misconceptions of Agile #1



- Holding ceremonies for the sake of holding ceremonies does not deliver benefits
- Ceremonies must adhere to core principles of Agile

*"Scheduling Agile Ceremonies  
Makes a Team Agile"*

## Misconceptions of Agile #2



The Facilitator Role is Filled by a Traditional Project Manager

- Facilitator role is different from the Project Manager role
- Requires a shift to Agile Mindset
- Should be a Servant Leader who assists with whatever the team needs to forward

## Misconceptions of Agile #3



- Agile recommends small cross-functional teams
- Allows the team to be nimble and powerful
- Challenging to track work when team is larger than 15

Team Size is Larger than Agile Frameworks Recommend

## Misconceptions of Agile #4



The Product Backlog is  
Managed Like a Traditional  
Requirements Document

- Backlog should not be a process-driven document
- Should focus on customer value and strategic objectives
- Product Owner plays a critical role

## Misconceptions of Agile #5



*"Documentation and Audit Trails  
for Compliance are not  
Important in Agile"*

- Agile does not mean no documentation
- Remove low value documents
- Document what is value to maintain and sustain the product

## Final Exercise

---

- Evaluate the situation at SocialKare.gov
- Outline three recommendations to set an Agile Mindset
- Consider the Agile Manifesto in your recommendations

## Success Strategy #1



*Ensure SocialKare.gov's  
Management Commitment*

- Ensures strategic input and support
- Continuing clear direction as product development evolves
- Resource support
- Removing significant roadblocks



## Success Strategy #2



*Provide autonomy to the team by empowering them*



- Ensures timely decisions to move the product development forward
- Team becomes more self-managed

## Success Strategy #3



*Incorporate Adaptive Planning with relevant buy-in*

- Inspiring product vision
- Roadmap that progressively evolves over time
- Release Plan and Iteration Plan sets expectations
- Daily updates to ensure proactive stance



## Lesson Overview



Why Agile?

Agile Teams

Agile  
Frameworks

- High Performing Teams
- Size, Structure and Skills
- Agile Governance

## Learning Objectives

---

By the End of the Lesson You Will Be Able To...

- Differentiate the 3 core roles (**Product Owner, Facilitator and Team Member**).
- Identify the characteristics of a **high-performing team**
- Identify and implement the approaches for high performing **team development and management**
- Implement methods for **conflict resolution** & psychological safety.

## Learning Objectives

---

By the End of the Lesson You Will Be Able To...

- Apply Agile principles to create teams with **optimal size and structure**.
- Develop effective **cross-functional skills** in a team.
- Use best practices to manage and work with **remote teams**
- Apply governance to enable **collaboration** and **decentralized decision making**



## Why Agile Teams?

## Why Agile Teams?

Focus on Value



### Focus on Value

- Business Value remains focus
- Aligns with organization's vision



# Why Agile Teams?

Focus on Value



Self-Organized Teams



## Self-Organized

- Team Buy-in
- Shared Ownership
- Motivation



# Why Agile Teams?

Focus on Value



Self-Organized Teams



Constructive Conflict Resolution



## Constructive Conflict Resolution

- Driven by shared vision
- Focus on the team, not tasks
- Transparently yet respectful



# Why Agile Teams?

Focus on Value



Self-Organized Teams



Constructive Conflict Resolution



Innovation Hub



## Innovation Hub

- Solutions for new requirements
- Diverse viewpoints
- Innovative solution options



# Why Agile Teams?

Focus on Value



Self-Organized Teams



Constructive Conflict Resolution



Innovation Hub



Cultivate Mastery



## Cultivate Mastery

- Time is put aside for learning and growth
- Continual improvement is built-in alongside built-in quality





## How Experts Approach Agile Teams

## Teamwork is Agile's Secret Sauce



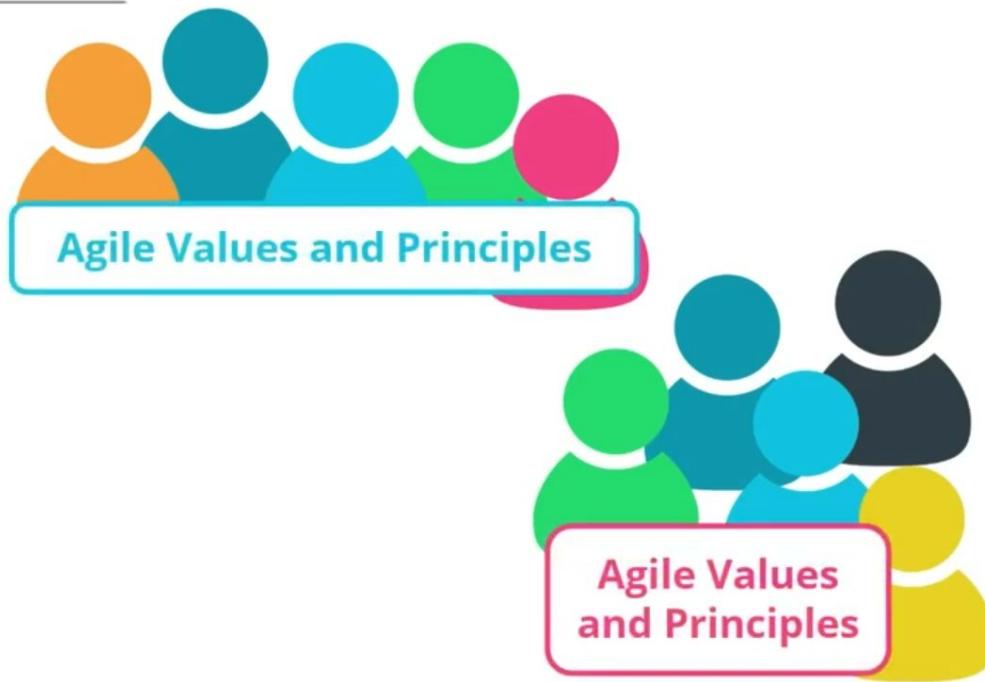
## Teamwork is Agile's Secret Sauce



## Teamwork is Agile's Secret Sauce



# Teamwork is Agile's Secret Sauce



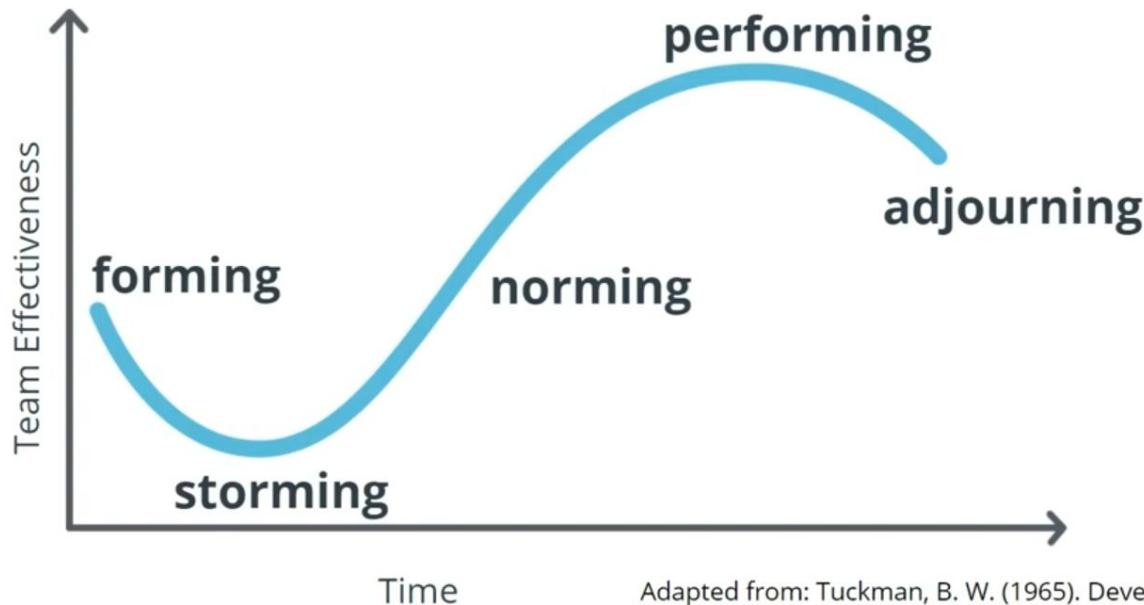
## Give it Time

---



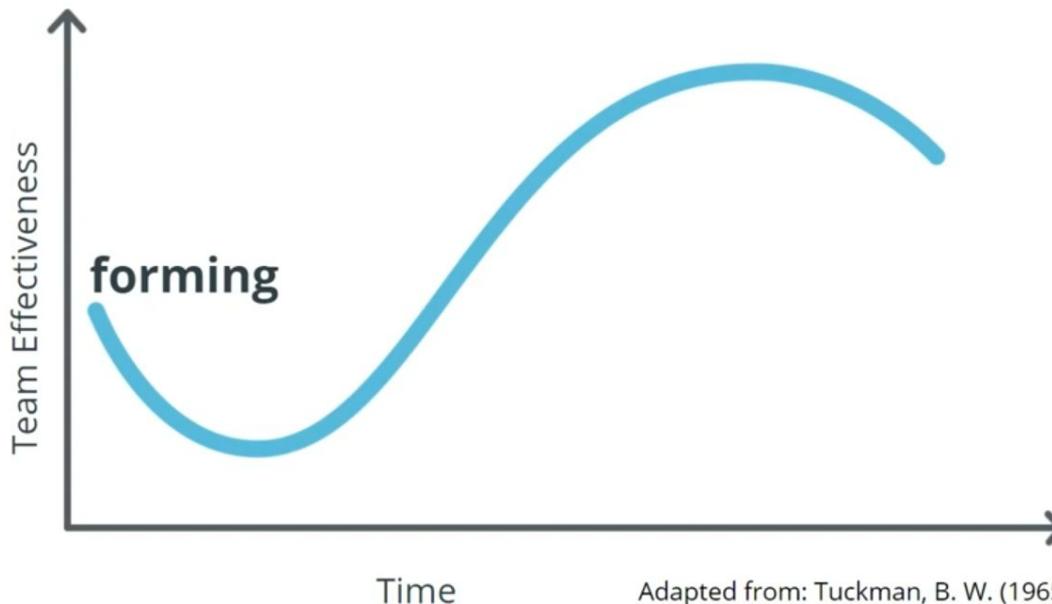
- Agile Teams need time to become awesome
- Patient guidance

## Tuckman Model



Adapted from: Tuckman, B. W. (1965). Developmental sequence in small groups.  
*Psychological Bulletin*, 63(6), 384–399.

## Tuckman Model

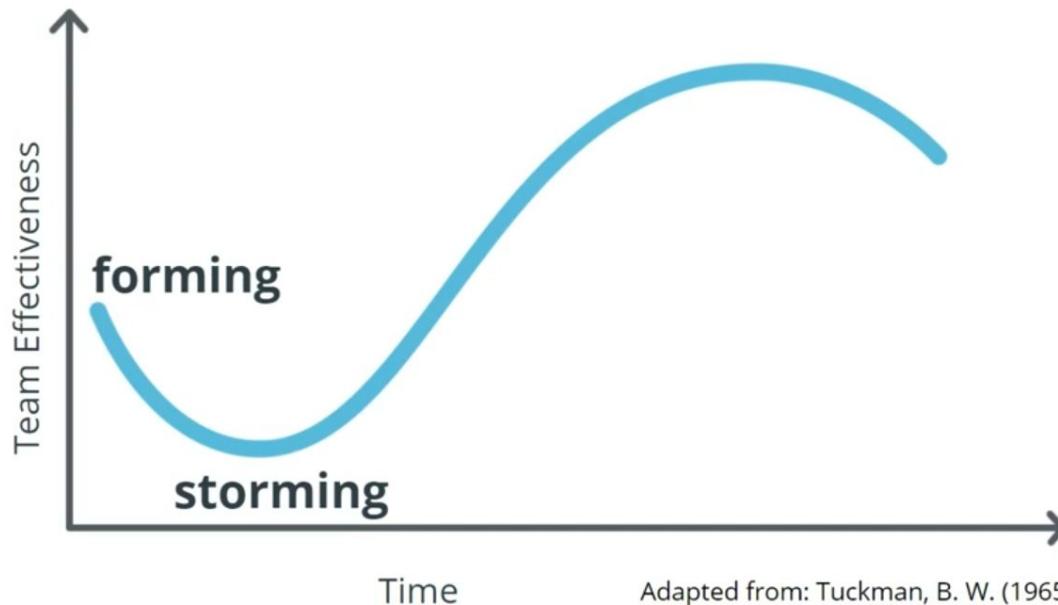


### Forming

- Team comes together
- Facilitator explains the ground rules, product vision and expected outcomes

Adapted from: Tuckman, B. W. (1965). Developmental sequence in small groups. *Psychological Bulletin*, 63(6), 384–399.

## Tuckman Model

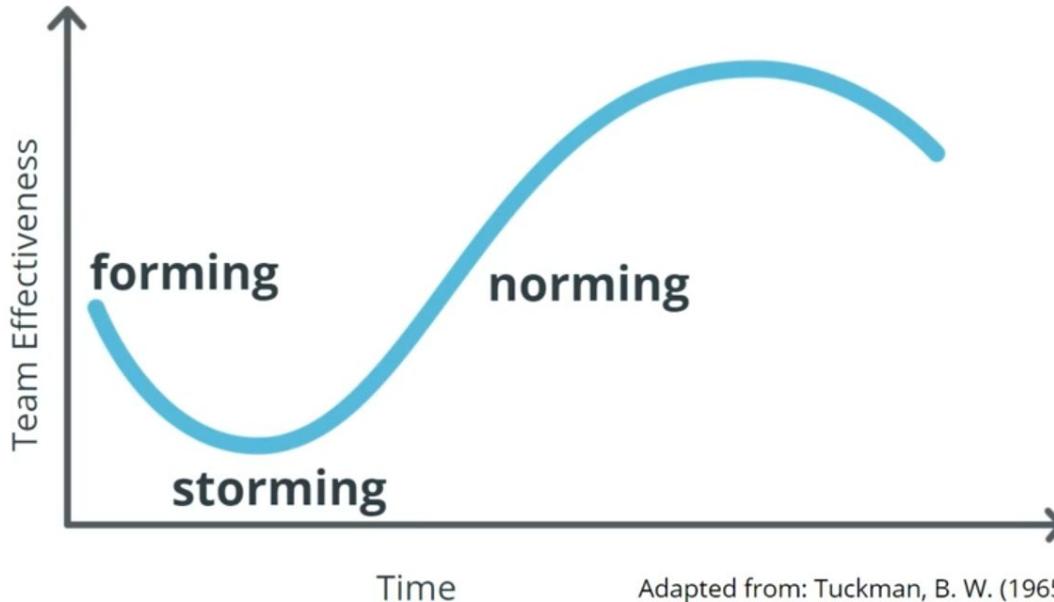


### Storming

- Velocity is diminished
- Ambiguity work, roles and product questions

Adapted from: Tuckman, B. W. (1965). Developmental sequence in small groups. *Psychological Bulletin*, 63(6), 384–399.

## Tuckman Model

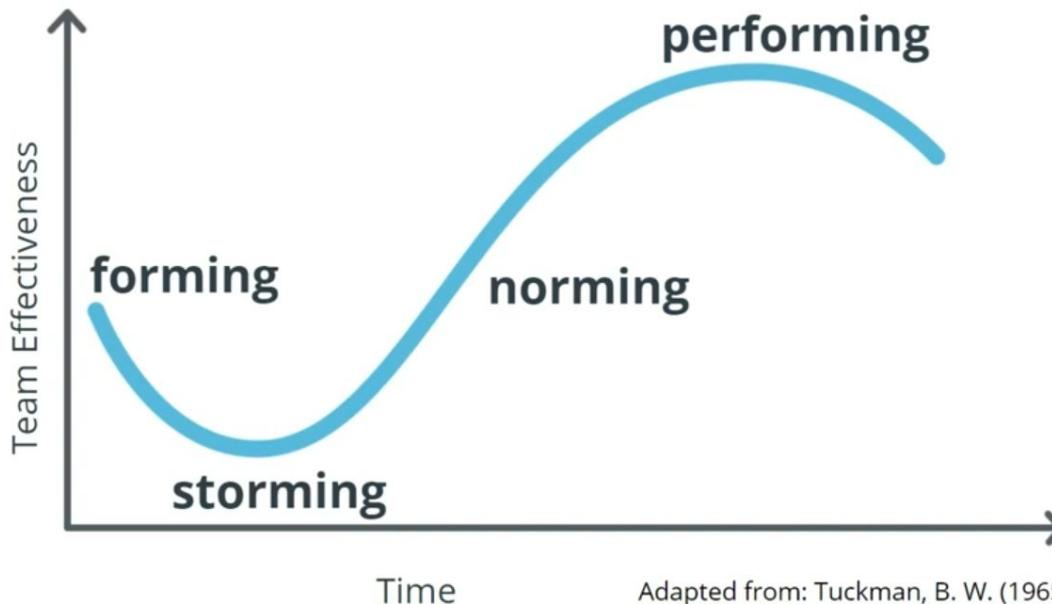


### Norming

- Trust and camaraderie have developed
- Team members focus on collective objectives

Adapted from: Tuckman, B. W. (1965). Developmental sequence in small groups. *Psychological Bulletin*, 63(6), 384–399.

## Tuckman Model

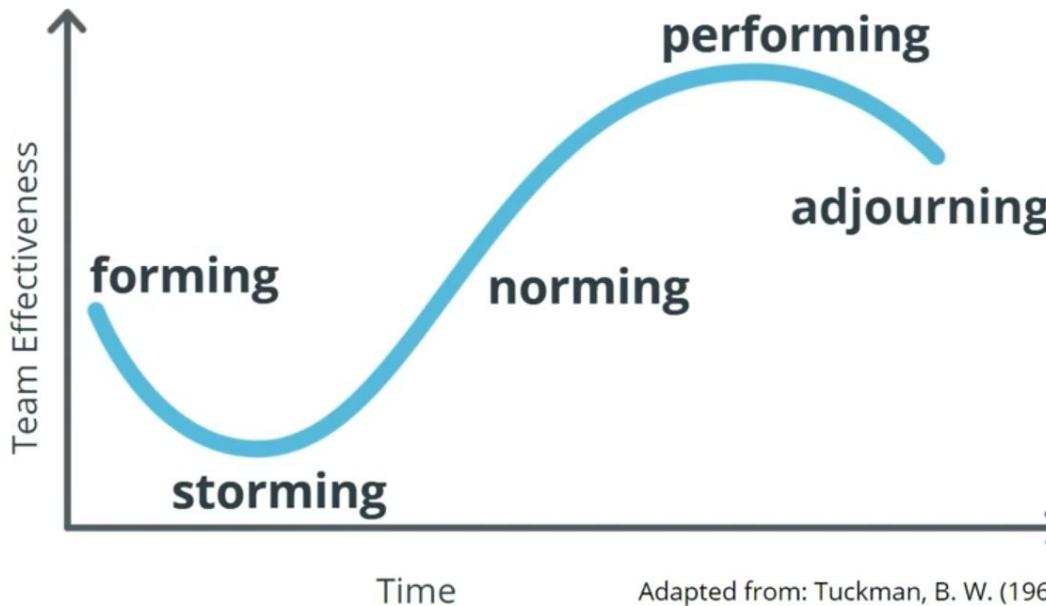


### Performing

- Team has worked together successfully
- Team has developed a cross-functional work style

Adapted from: Tuckman, B. W. (1965). Developmental sequence in small groups. *Psychological Bulletin*, 63(6), 384–399.

## Tuckman Model



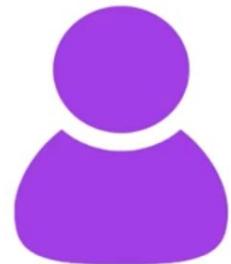
### Adjourning

- Product is no longer needed
- Reallocate resources to more valuable work

Adapted from: Tuckman, B. W. (1965). Developmental sequence in small groups. *Psychological Bulletin*, 63(6), 384–399.

## Core Roles

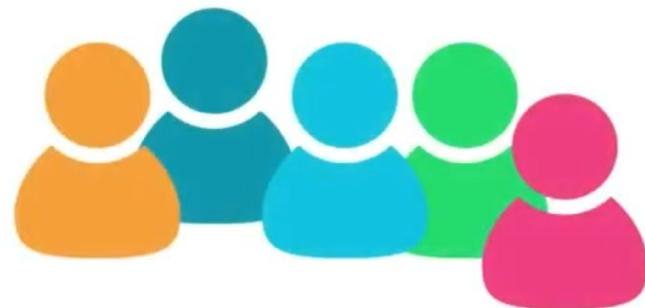
---



Product Owner

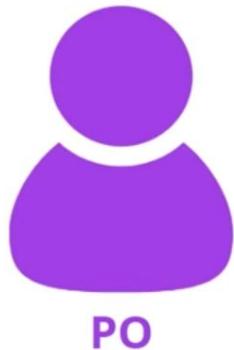


Facilitator



Agile Team

## The Role of The Product Owner



- Maximize value delivery
- Prioritize and manage the Product Backlog

## Managing The Backlog

---

- Backlog items refined with necessary details
- Prioritized for business value
- Visible and Transparent



## The Role of The Agile Team



**Agile Team**

Self-directed and self-organized

- Team manages work after business based on prioritization by the PO
- Boosts Team's confidence and ability to deliver
- **Team** is responsible for delivering shippable product after every iteration

## Team Members Have T-Shaped Skills



- Skill depth in at least one area
- Capabilities in areas outside their expertise
- Allows for role-blending

## The Role of The Facilitator



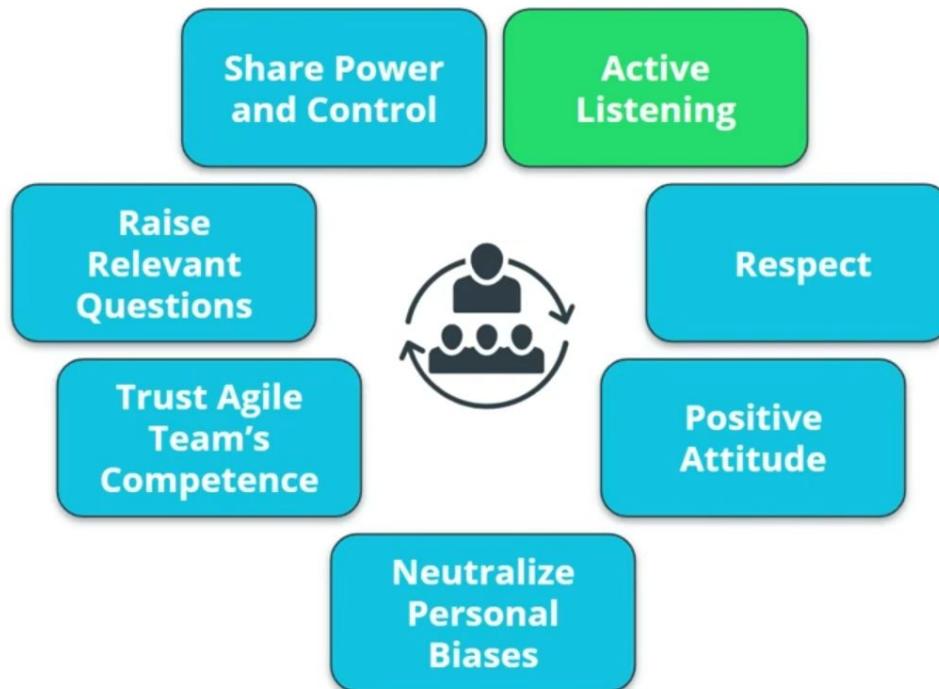
- Keeps Agile Team moving towards business objectives
- Removes blockages
- Agile Coach

## Servant Leadership



*To serve  
and  
support*

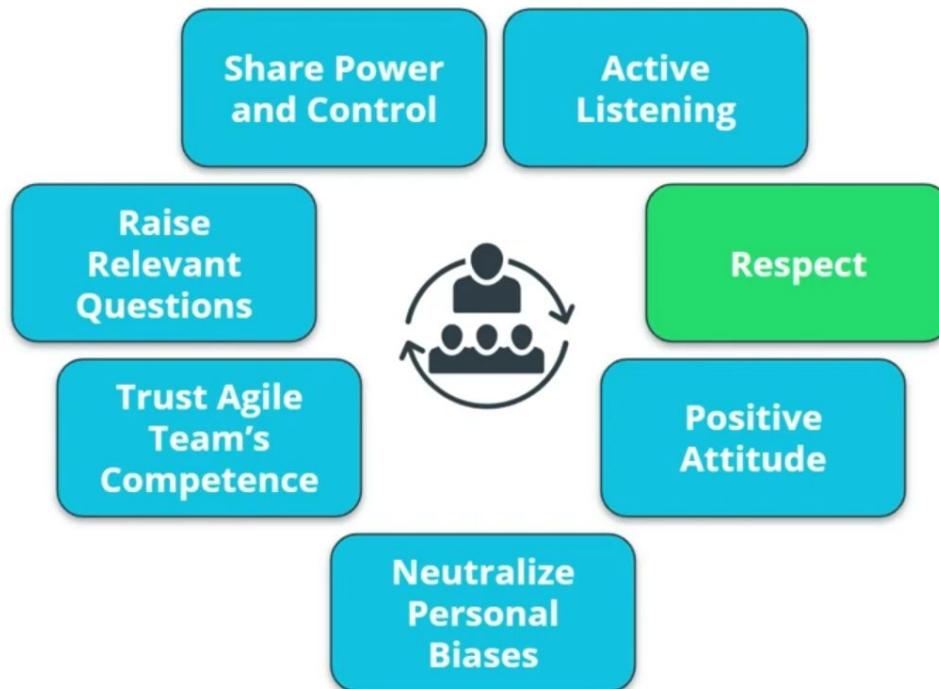
# Servant Leadership



## Active Listening

- Fully concentrate with other's perspective
- Dispute resolution and decision making
- Non-verbal cues to convey understanding

# Servant Leadership



## Respect

- Aligns with Agile Mindset
- Treat everyone with integrity and honor

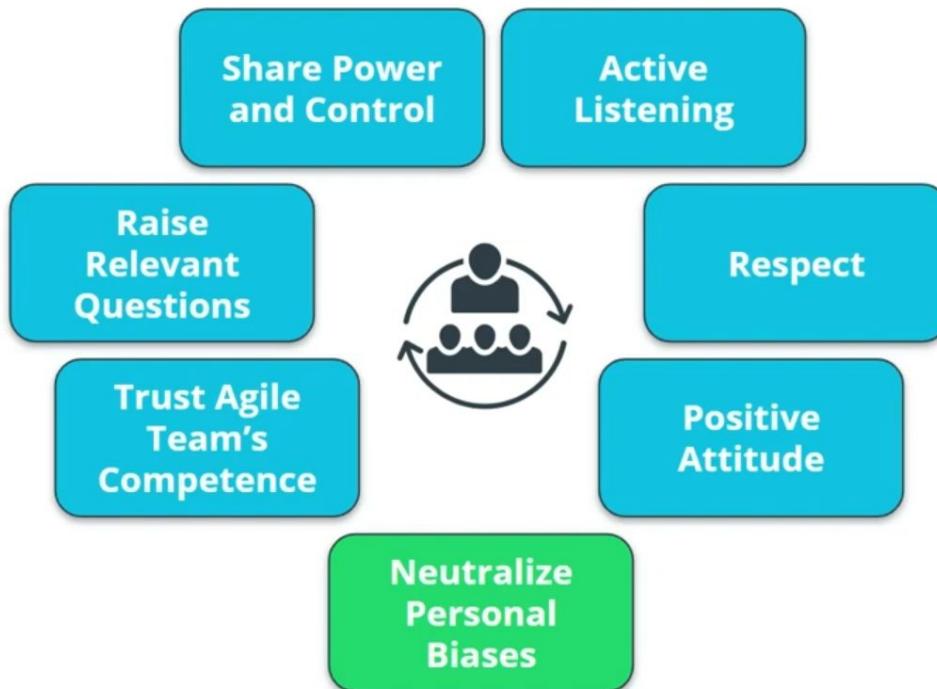
# Servant Leadership



## Positive Attitude

- Essential during times of crisis or confusion
- Model desired behavior
- Contagious and positive energy spreads

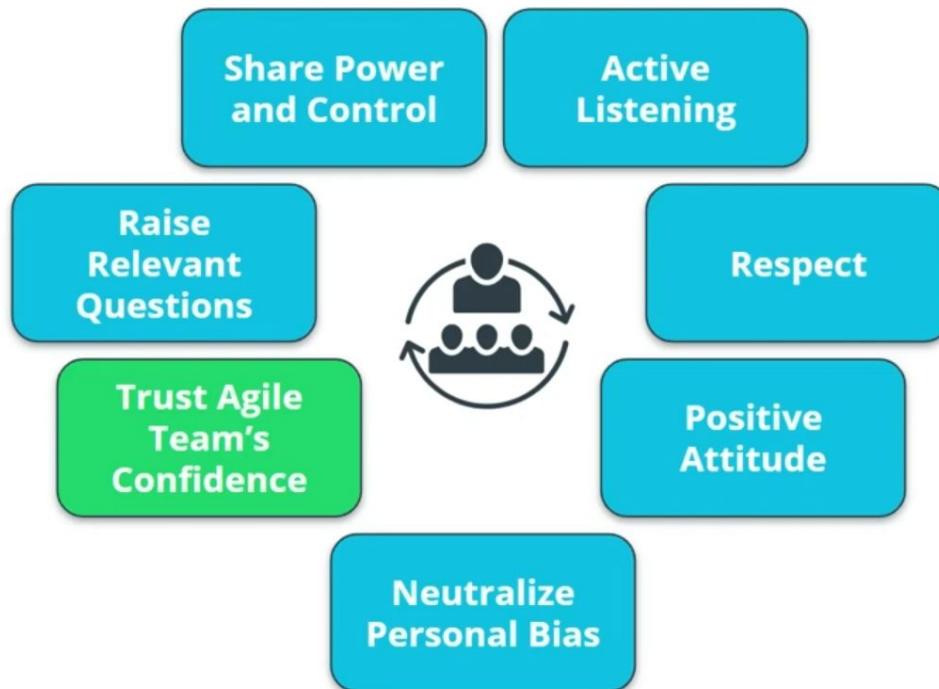
# Servant Leadership



## Neutralize Personal Bias

- Establish a Working Agreement
- Reduce risk of bias through Role Blending
- Establish transparent measurable goals

# Servant Leadership



## Trust

- Between Facilitator, PO and Agile Team
- Between Agile Team members

# Servant Leadership



## Raise Relevant Questions

- Learning Mode
- Ask unbiased open ended questions
- Honest and Collaborative

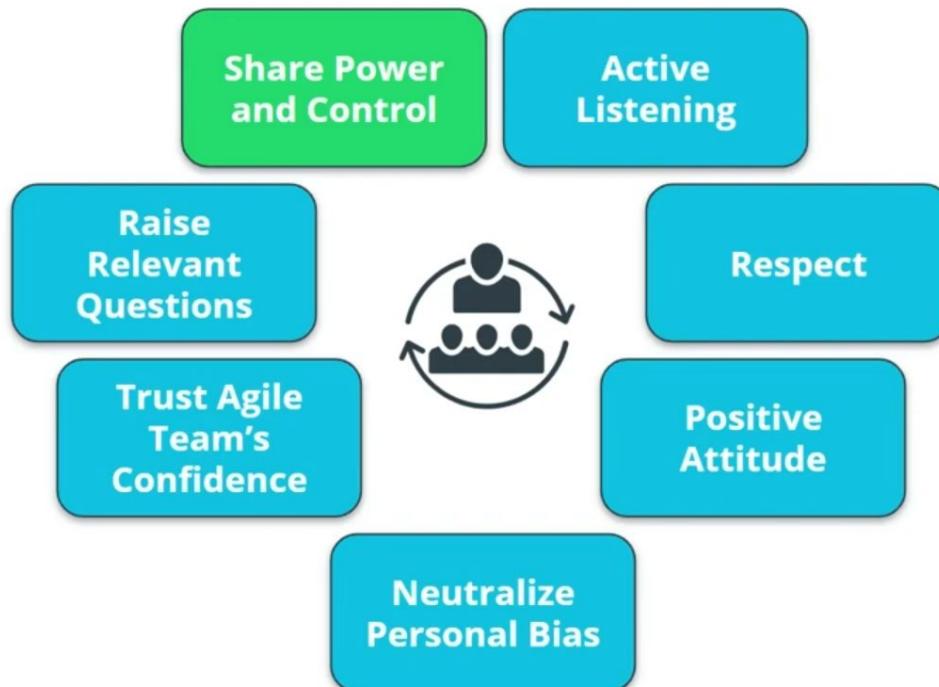
# Servant Leadership



## Raise Relevant Questions

- Learning Mode
- Ask unbiased open ended questions
- Honest and Collaborative

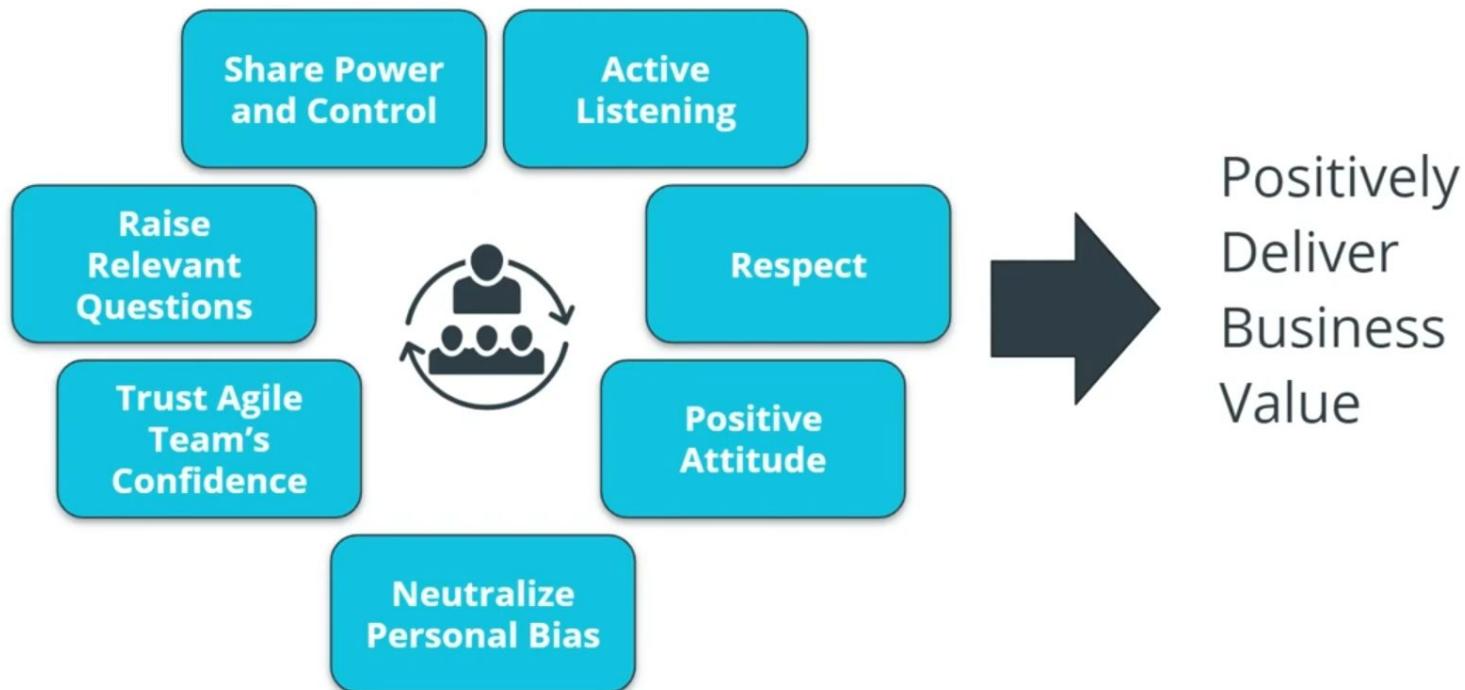
# Servant Leadership



## Share Power and Control

- Information
- Knowledge
- Authority

# Servant Leadership



## **Three Core Roles Exercise**

---

- Identify three core roles in the SocialKare.gov context
- Assign resources to the three core roles



# SocialKare.gov Employees

*This information will be helpful for the Three Core Roles Exercise*

- **John Details:**

John is a detail oriented individual who likes to plan everything out in tremendous detail. He has several years of experience as a Technical Project Manager. Prior to stepping into the role of Project Manager, John was a Developer. He is known to have good communication skills.

- **Sarah Tenure:**

Sarah has been with the Agency responsible for launching SocialKare.gov for over 2 decades. Sarah has a lot of business knowledge and carries a certain amount of credibility due to this tenure. Sarah is strong headed and likes to make decisions based on how she sees things because she believes she has better insight than others who may have been with the organization for as long as she has.

- **Tim Devs:**

Tim has excellent technical skills. He prefers working as an individual contributor where he can spend time diving into the technical challenges on his own. He is well respected for his technical expertise. Tim maintains a strong focus on technical delivery and prefers the business users to stay out of his way while he is working through the development work. Tim leads a team of 5 developers with specialized vertical skills who are also on this project team

- **Jane Dollars:**

Jane has sponsored this project. She is a senior executive and likes to stay involved with the day to day activities of projects that she sponsors.

## Three Core Roles Exercise



Role	Responsibilities	Employee	Justification
Facilitator	Doubles up as Agile Coach  Facilitating the Agile Team's work  Protecting from problems	John Details	Good communication skills  Experience as a Project Manager helpful

## Three Core Roles Exercise



Role	Responsibilities	Employee	Justification
Product Owner	Voice of the Customer (VOC)  Prioritize Backlog  Share Business Decisions & Provide Information	Sarah Tenure	Business Acumen  Credibility  Relationship with Business Users

## Three Core Roles Exercise



Role	Responsibilities	Employee	Justification
Agile Team Member	Responsible for Creation and Delivery of solutions	Tim Devs	Important Technical Skills

## High Performing Agile Teams



- Deliver consistent results
- Model desired behavior and characteristics
- Surpass expectations

## High Performing Agile Teams



## **Self-Directed and Independent**



- Unleashing team ambition
- Cutting unnecessary red tape

## T-Shaped Small Teams

---

- Small teams deliver Agile, product-centric value
- T-shaped skills allow role blending



## Focused on A Mission

---



- Unleashing team ambition
- Cutting unnecessary red tape

## **Supported By Executives**

---

- Create the right environment
- Push for cultural transformation
- Breakdown silos
- Step back and allow teams to flourish



## Agile Teams Make Decisions On Their Own

---

- Minimal dependence on other teams
- Highly motivated and skilled
- Shared sense of purpose



## Agile Teams Develop Team Norms and Ground Rules

- Mutually agreed upon by the team
- Useful for guiding group behavior
- Enabler for discussing the 'undiscussable'



# Building A High Performing Agile Team



Committed to:

- Delivering value
- Continuous improvement
- Agile mindset

## Inspire the Agile Team



- Senior Leadership should inspire
- Creates enthusiasm
- Team understands their contribution to the organization

## Build Cross Functional Diverse Team

- Diverse perspectives leads to richer options for solutions
- Team benefits from different viewpoints



## Maintain Focus As Team Evolves



- Team needs breathing room to grow
- Culture needs to guide evolution so the team stays focused
- Unsatisfied team delivers unsatisfactory results

## Encourage Innovation and Free Thinking

---

- Talk openly constructively
- Discuss challenging questions
- Drives better results



## Communicate Clearly

---



- Clarity in direction for the Team
- Lack of clarity frustrates the Team
- Clear communication benefits everyone

## Work Collaboratively To Solve Problems

- Allows Team to take advantage of different skill sets and perspectives
- Stimulates creativity
- Small Agile Team size helps



## Timebox Meetings

---



- No hostages
- Break away from process centric mindset
- Minimize unproductive meetings

## Recognize and Appreciate

---

- Built into the Agile frameworks
- Simple steps can be a motivator
- Periodic and consistent reminders of the Agile Team's contributions



## **Include Uncommitted Objectives**



- Identifies work that is planned but without a full commitment
- Allows for flexibility while driving team to accomplish the more
- Builds and fuels the team's confidence

## **Take Time Out to Connect and Have Fun**

---

- Encourage team members to be themselves
- Removes toxicity
- Having fun is a key strategy for driven teams



## High Performing Agile Team Development and Management

- Inspire the Agile Team
- Build cross functional diverse team
- Maintain focus as the team evolves
- Encourage innovation and free thinking
- Communicate clearly
- Work collaboratively to solve problems
- Timebox meetings
- Recognize and Appreciate
- Include uncommitted objectives
- Take time out to connect and have fun

# High Performing Agile Teams Never Disappoint



- Delivers consistent results despite challenges
- Models desired behavior
- Surpasses expectations

*Leadership allows the Team to be self-directed, independent and ambitious in delivering value*

## The Need For Remote Agile Teams

- Business is globally interconnected
- COVID pandemic necessitated remote work at an unprecedented level
- Permanent change to team management norms?



## Team Communication



- ✖ Limited communication can impact Team bonding
- ✓ Use technology to creatively connect as a Team

## Time Zone Differences



Teams find it hard to sync up calendars



Use up-front planning to ensure the Team connects

## Work From Home



Work from home adds distractions and can impact well-being

## Work From Home



Work from home adds distractions and can impact well-being



Set a clearly defined schedule and home office location

## Fewer Catch-ups



Reduces collaborative moments



Use virtual communication tools

## Lack of Team Cohesion



Reduces collaborative moments



Ensure a clear vision and clarity on responsibilities

## Remote Agile Teams



- Remote is here to stay!
- Covid-19 pandemic has shown the value of remote Agile teams
- Remote should be a high priority for successful organizations

## High Performing Agile Teams Exercise

---

- Identify three Agile Team members who you can work with to build a high performing team
- Identify three recommendations to move to a high performing model



# High Performing Agile Teams



Facilitator	Coaching Opportunity	Justification
John Details	Focus on facilitation and not technical solutioning; let the SocialKare.gov team be self-organized and self-directed	Former technical background might inhibit ability to allow the team to be more autonomous  Should not stifle the Agile Team's innovative and creative side

# High Performing Agile Teams



Product Owner	Coaching Opportunity	Justification
Sarah Tenure	<p>Listen to the Agile Team for input</p> <p>Make final priority decisions</p> <p>Do not interfere in the Agile Team's technical decisions</p>	<p>Needs to be more collaborative</p> <p>Let Facilitator run the Ceremonies</p>

# High Performing Agile Teams



Team Member	Coaching Opportunity	Justification
Tim Devs	Allow cross-functional skills development  Collaborate with business	Avoid micromanagement of developers  Give room to innovate

## Optimal Agile Team Has Between 5-12 Members

### < 5 Members

- Difficult to have all of the required competencies
- Team may not be multidisciplinary



### > 12 Members

- Difficult to self-direct and self-organize
- Increasing noise/communications challenges



## Agile Team Size and Structure



$n$  = number of people

$$\text{Number of Communications Channels} = \frac{n * (n - 1)}{2}$$

More channels = more noise and more distractions



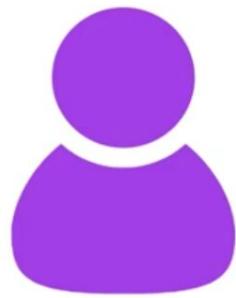
## Clarity and Respect for Core Roles

---

- Role clarity is important
- Lack of clarity can cause stress and confusion



## Clarity and Respect for Core Roles



**Product Owner**

- Voice of the Customer
- Product Champion



**Facilitator**

- Servant Leader
- Ensures the team runs smoothly



**Agile Team**

- Determines how the work gets done

## **Autonomous, Self-Organizing And Cross-Functional**

---

- T-shaped Skills
- Ignite innovation
- Break silos
- Consistent flow of value delivery



## Cross-Functional Agile Teams

- Everyone works towards a common goal
- All team members expected to exhibit T-shaped skills
- Small self-directed teams



## Organizing an Agile Team Exercise

---

- Recommend three non-core roles
- Their value to the Agile Team



# High Performing Agile Teams



Non-Core Role	Justification
Sponsor	<p>Senior sponsorship is critical</p> <p>Sponsor provides formal authorization and funding</p>

## High Performing Agile Teams



Non-Core Role	Justification
Enterprise Architect	<p>Valuable technical resource</p> <p>Can seek guidance on alignment with SocialKare.gov enterprise wide architecture model</p>

# High Performing Agile Teams

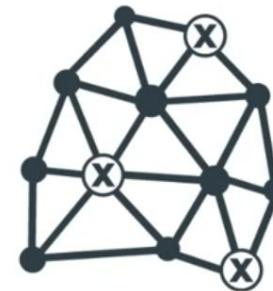


Non-Core Role	Justification
Domain Subject Matter Expert	<p>Valuable resource for Product Owner</p> <p>Can provide domain knowledge and advice</p>

## Agile Governance

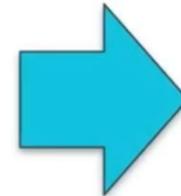
---

*Decentralized  
Decision Making*



## Tolerances Are Essential For Agile Governance

- Especially true for cost and time
- When in danger, de-scoping is recommended
- Timeboxing is critical



## Decentralized Decision Making Empowers The Team



- Open communication
- Constructive debate
- Quick resolution of issues
- All Team members commit to decisions

## Agile Governance Delivers Goodwill and Trust

- Clarity about roles
- Thresholds and tolerances  
for decentralized decisions



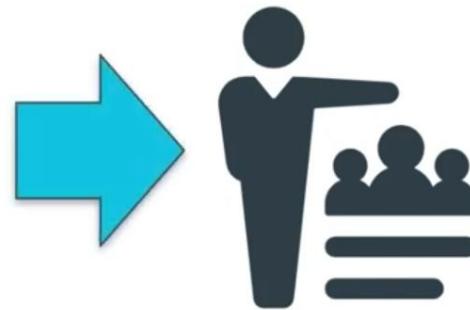
## Agile Promotes Light and Simple Governance

- Teams self-monitor progress
- Use transparent metrics
- Assessment relies on observation and engagement



## **Some Decisions Remain Centralized**

- Decisions that span a longer time frame
- Decisions that have an organization-wide impact
- Rare decisions that require deeper analysis



## Agile Governance

---

*It is essentially about  
Decision Making*



## Agile Governance Exercise

---

- Three centralized decisions
- Three decentralized decisions



# High Performing Agile Teams



Centralized Decisions	Justification
Span a longer time frame	Unlikely to alter in the short term
Have a larger organization wide impact	Large and organization wide economic benefits
Rare decisions	Not urgent and a deeper more detailed analysis is needed

# High Performing Agile Teams



Decentralized Decisions	Justification
Regular or frequent decisions	A centralized decision here would be of limited value
Time critical decisions	Should not be delayed because such delays will have a significant cost of delay for SocialKare.gov.
Local or team level information is needed	These de-centralized decisions need specific local or team level context



# What Kills High Performance Teams

## Pitfall #1



*"Conflicts not managed effectively"*

- Agile Teams sometimes spiral into an adversarial approach
- Agile Team must be reminded of the value of diversity in thoughts
- Ultimate goal is a win-win scenario

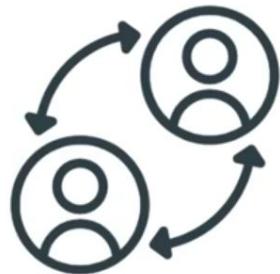
## Pitfall #2



*"Blame game begins"*

- The blame game and negative criticism quickly escalates to unhealthy conflict
- Disengagement kicks in
- Learning mindset encourages curiosity and solution oriented thinking

## Pitfall #3



*"Rules of Engagement Not Followed"*

- Open and honest communication requires adherence to Team rules of engagement
- Otherwise it kills trust
- Important to slow down when necessary to re-group

## Pitfall #4

---



*"Spark or passion is dimmed"*

- If Agile Team does not see value in their work, it kills motivation
- Product Owner must continually remind Agile team of alignment with vision

## Lesson Recap

---

- High Performing Teams
- Size, Structure and Skills
- Agile Governance



**Agile Teams**

# Lesson Overview



Why Agile?

Agile Teams

Agile  
Frameworks

- Why Frameworks?
- Scrum
- Kanban
- XP

## Learning Objectives

---

By the End of the Lesson You Will Be Able To...

- Identify the advantages of using a framework
- Identify the specific and shared characteristics of Scrum, Kanban and XP frameworks
- Determine which frameworks are appropriate for a given project.
- Identify the core team roles assign team members to each role
- Implement best practices for each framework, including ceremonies and artifacts

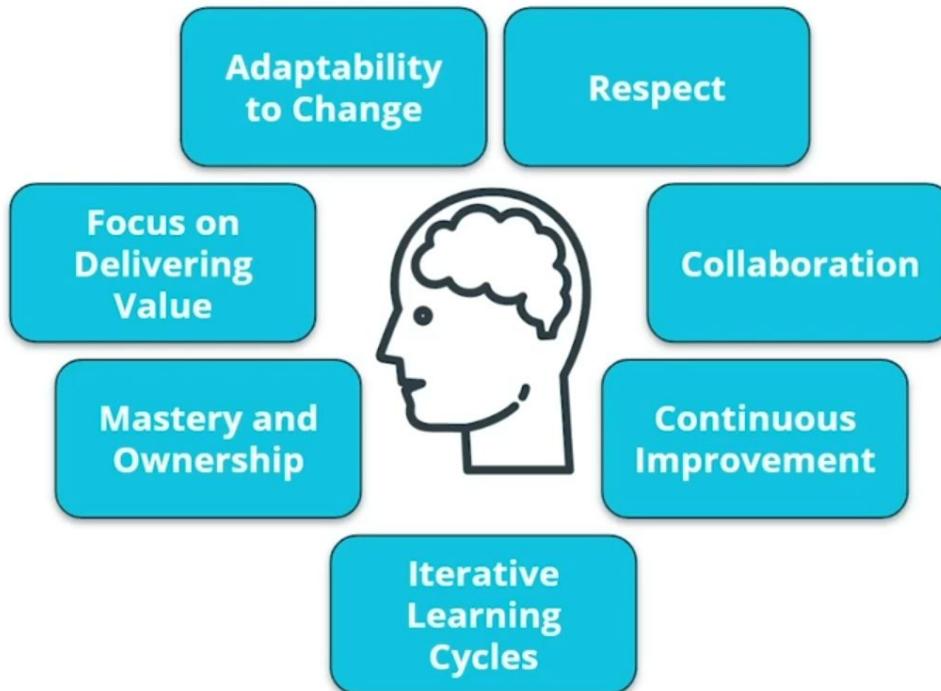
## Why Agile Frameworks?



AGILE  
FRAMEWORKS

A large, light blue arrow pointing to the right. Inside the arrow, the words "AGILE FRAMEWORKS" are written in white capital letters.

## The Agile Mindset



*The set of attitudes supporting an Agile working environment*

## Doing Agile

---

- Adopting practices
- No commitment to principles and values



## Being Agile

---

- Agile Mindset
- Agile Manifesto
- Agile Principles

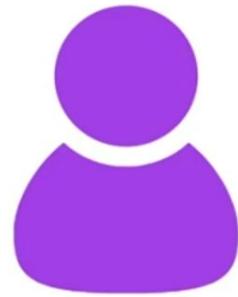


# Four Paired Core Values Drive 12 Agile Principles



Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

## Clarity and Respect for Core Roles



### Product Owner

- Voice of the Customer
- Product Champion



### Facilitator

- Servant Leader
- Ensures the team runs smoothly



### Agile Team

- Determines how the work gets done

## Practice-Oriented

---

- Makes the abstract more practical
- Particular guidelines are provided
- Can execute on Agile's promise of adaptability with value

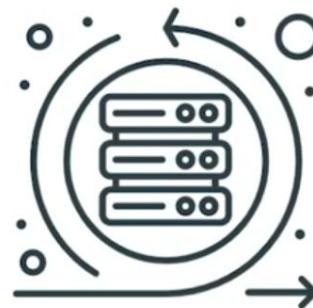




## How Experts Approach Agile Frameworks

## Thinking About Agile Frameworks

- Frameworks bring good intentions to fruition
- Tried and tested Framework brings comfort to the process



## Focus on the Right Individuals and Interactions

**Individuals** and interactions over **processes** and tools



Skilled Team members have the most significant impact on success

## **Focus on the Right Individuals and Interactions**

---

- Recruit and retain the right skilled resources
- Cultivate a collaborative and value-driven environment
- Team-oriented mindset



## Don't Force Agile Unnecessarily



- Do not impose Agile on a Team that is already performing
- High performing Teams may already be Agile
- Additional practices can be gradually introduced

## Refined Users Stories are Important

- PO must ensure User Stories are refined
- PO's job is not to identify technical solutions
- Excessive PO intrusion kills fire for innovation



## Respect the Backlog

---

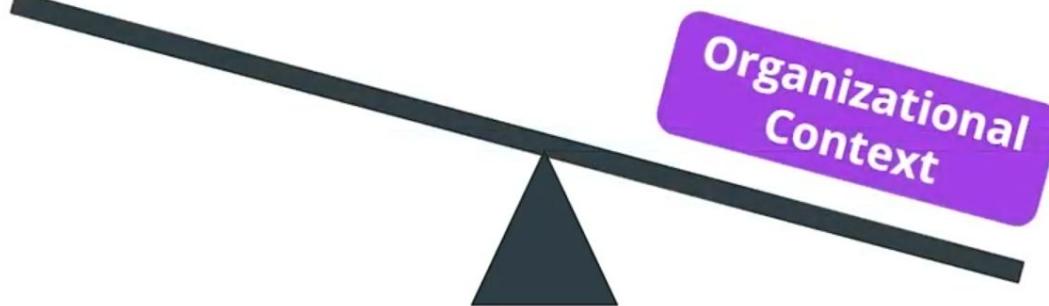


- Backlog's priorities must be respected
- Agile Team must not over-commit
- Tracking and Collaboration tools like JIRA, ADO and Rally should be kept up to date

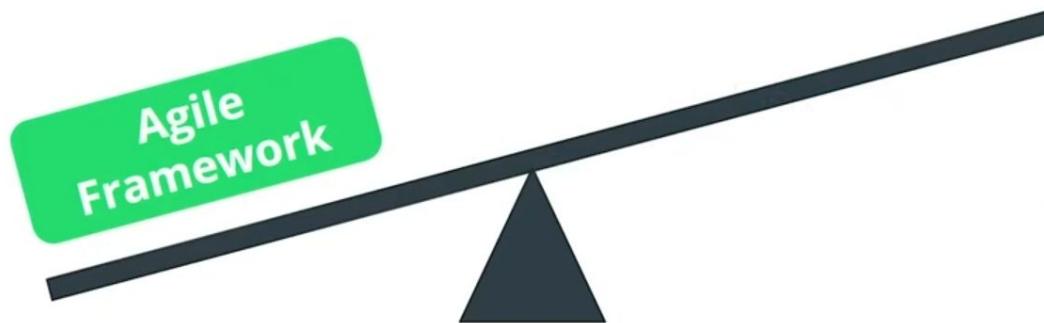
## Framework “Balanced” with Organizational Context



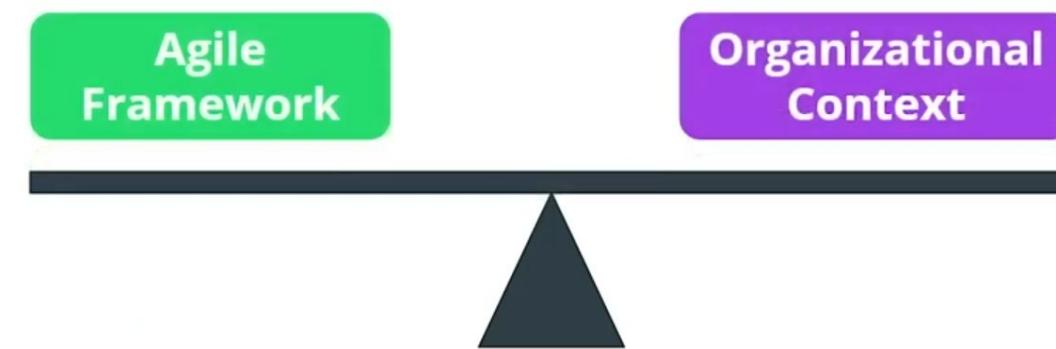
## Framework “Balanced” with Organizational Context



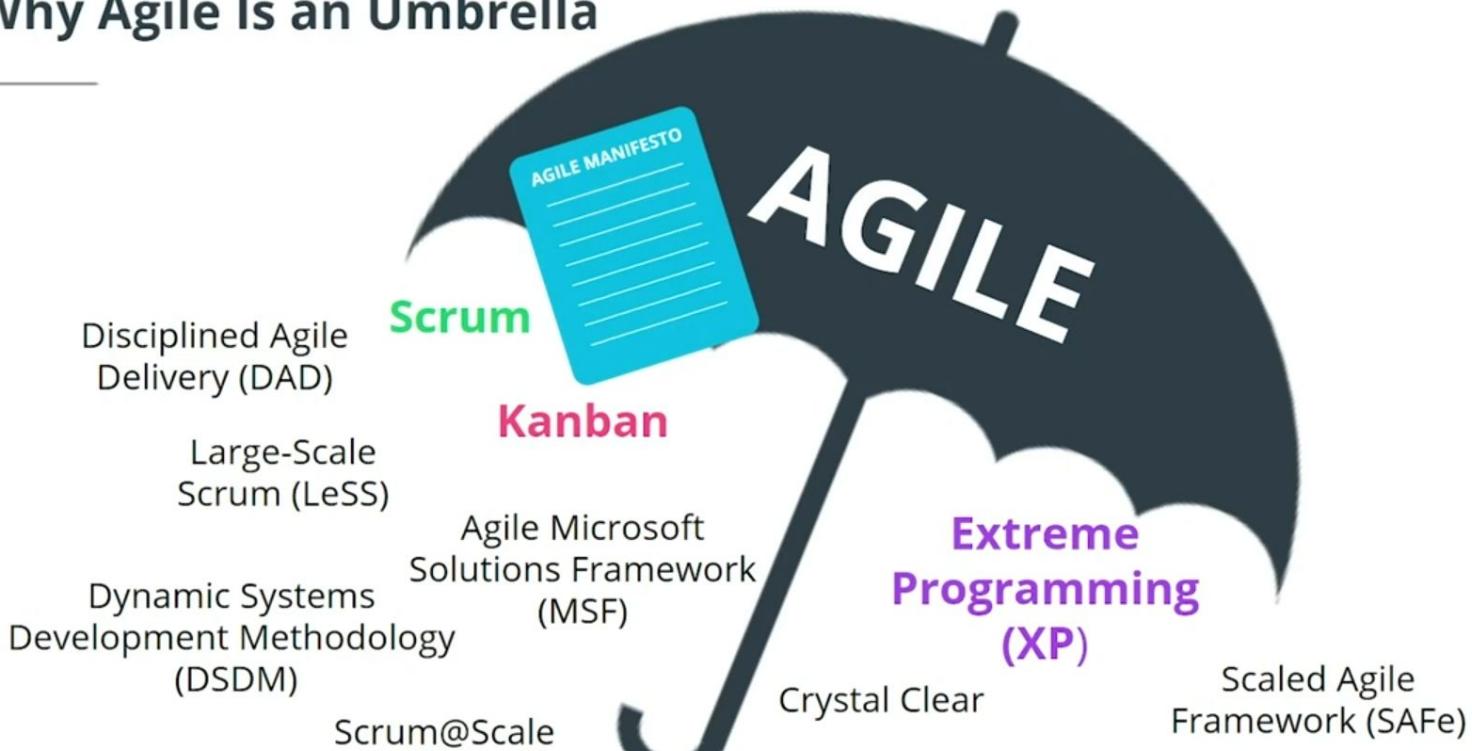
## Framework “Balanced” with Organizational Context



## Framework “Balanced” with Organizational Context

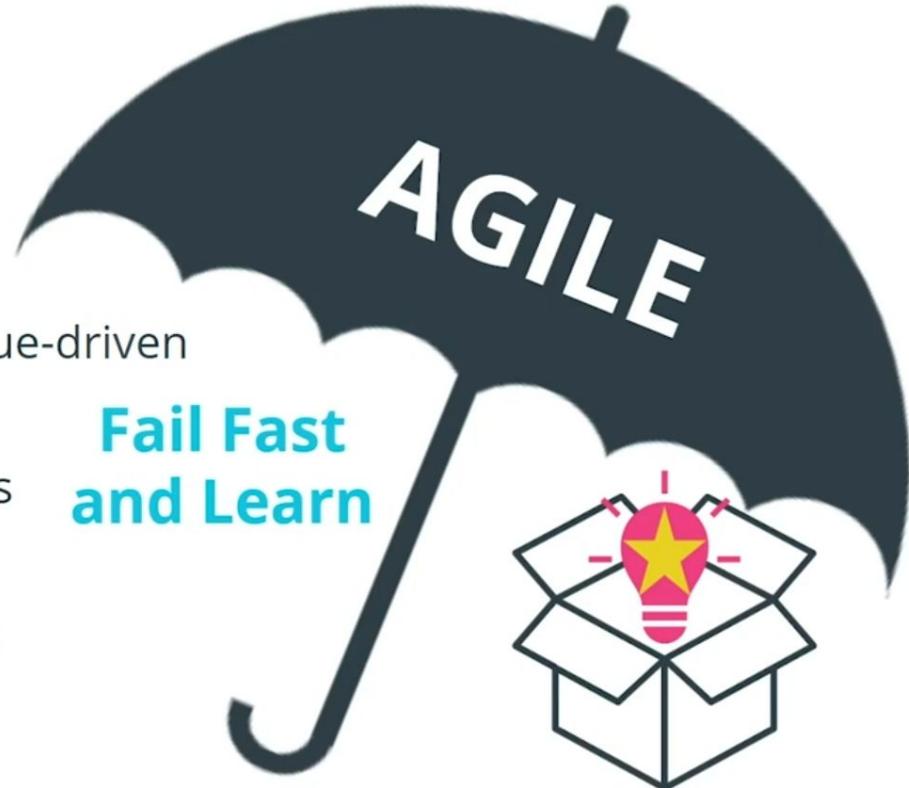


## Why Agile Is an Umbrella



## Why Agile is an Umbrella

- Features need to be adjusted along the way
- Agile Team focused on value-driven functionality with quality
- Fail Fast and Learn reduces waste
- Growth Mindset promotes directed trial and error



## The Agile Umbrella

### **Deliver Value**

- Business Value remains focus
- Aligns with organization's vision



# The Agile Umbrella

## Continual Improvement

- Ongoing effort
- Improve Product Features and Processes
- Incremental improvements



# The Agile Umbrella

## Flexibility

- Foundation for all Agile Frameworks
- Flexibility in processes and adaptive planning
- Adaptable to changes



# The Agile Umbrella

## Trust and Respect

- Motivates the team
- Builds commitment
- Foundation for all frameworks



## The Agile Umbrella

### Cultivate Mastery

- Time is put aside for learning and growth
- Continual improvement is built-in alongside built-in quality



## Agile Frameworks Exercise¶

- List at least 3 specific benefits of using an Agile Framework
- Explain how those benefits will help SocialKare.gov achieve its goals



## Agile Frameworks Exercise Solution

Benefit from Using an Agile Framework	How The Benefit will Help SocialKare.gov
Based on real-life experiences	SocialKare.gov would gain from a proven Framework
There's a higher level of predictability	SocialKare.gov will benefit from defined Ceremonies and Agile Team Roles
Organizations benefit from higher quality delivery	SocialKare.gov will deliver solutions in a timely manner
Frameworks can be customized	Best done once SocialKare.gov has tried out a Framework in its original form

## The Scrum Framework

---

- Most common Agile Framework
- Team creatively manages work
- Focus is on rapid delivery of features in response to market demand



## The Scrum Framework



- Work is delivered in **Sprints**
- Quality is built in
- Team is self-directed and self-organized

## The Scrum Framework

---

- Provide updates daily
- Deliver in short Sprints
- Opportunities for improvement identified through Retrospectives

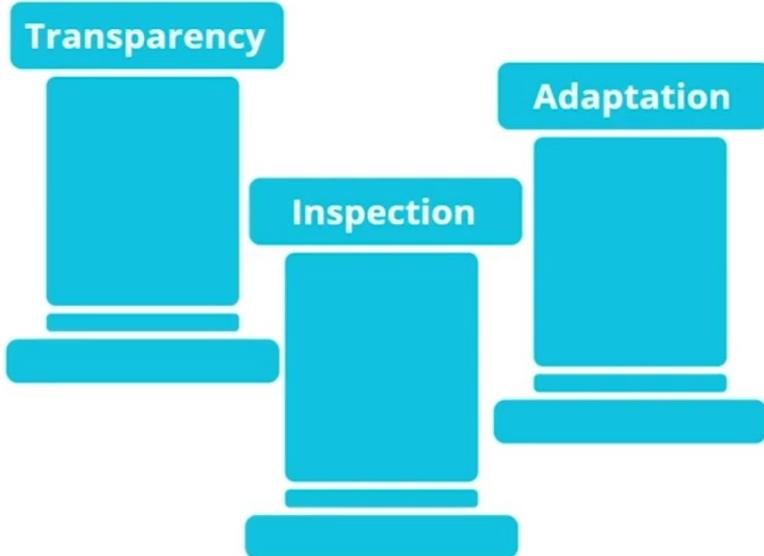


## The Scrum Framework

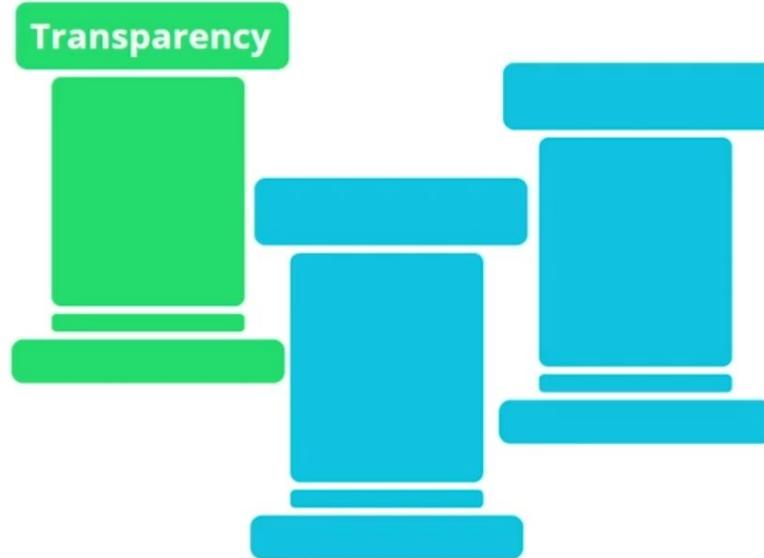


- Specific Roles
- Ceremonies
- Artifacts

## Scrum Pillars and Values



## Scrum Pillars



### Transparency

- Explicitly give visibility
- Define common standards
- Have a consistent understanding
- Builds trust and openness

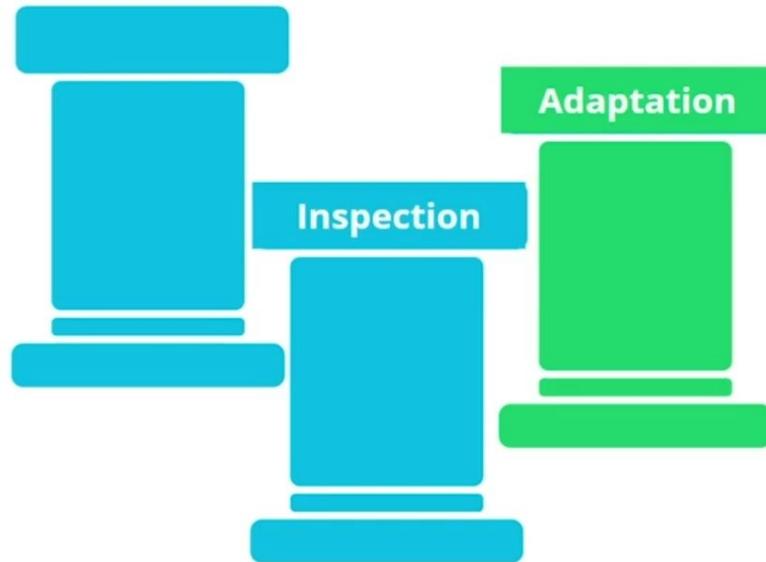
## Scrum Pillars



### Inspection

- Business users review Agile Team's progress
- Should do it at least once per Sprint
- Identifies unwanted variances from Sprint Goals

## Scrum Pillars



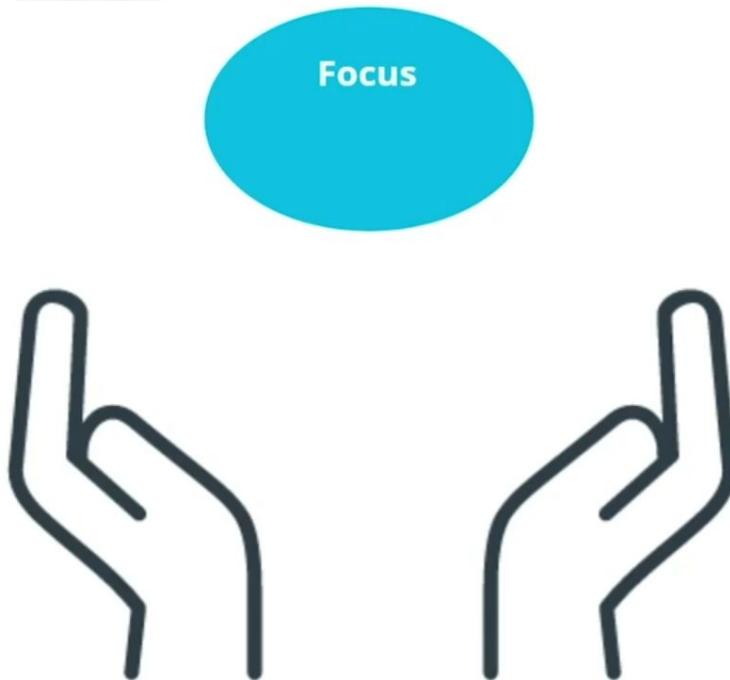
### Adaptation

- Adjust based on opportunities for improvement
- Include process improvements
- Proactively deal with such adjustments

## Scrum Values



## Scrum Values



### Focus

- Collaborative effort in Scrum
- Requires everyone to focus on the work to be completed

## Scrum Values

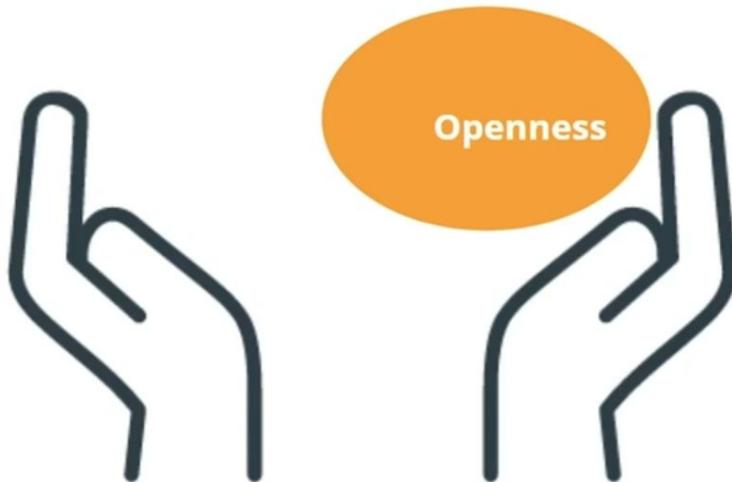
---



### Courage

- Have the courage to do what is right
- Dig deep into challenging problems

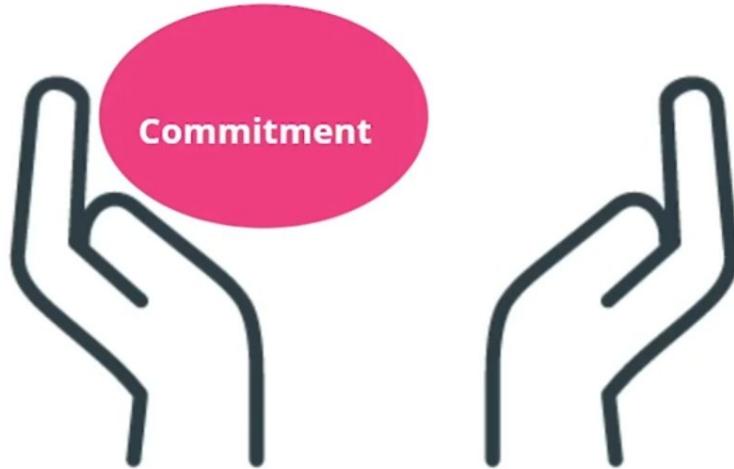
## Scrum Values



### Openness

- Key to Agile Team performance
- Agile Team, as well as Stakeholders, must agree to openness
- Ensures everyone will be comfortable to openly discuss work

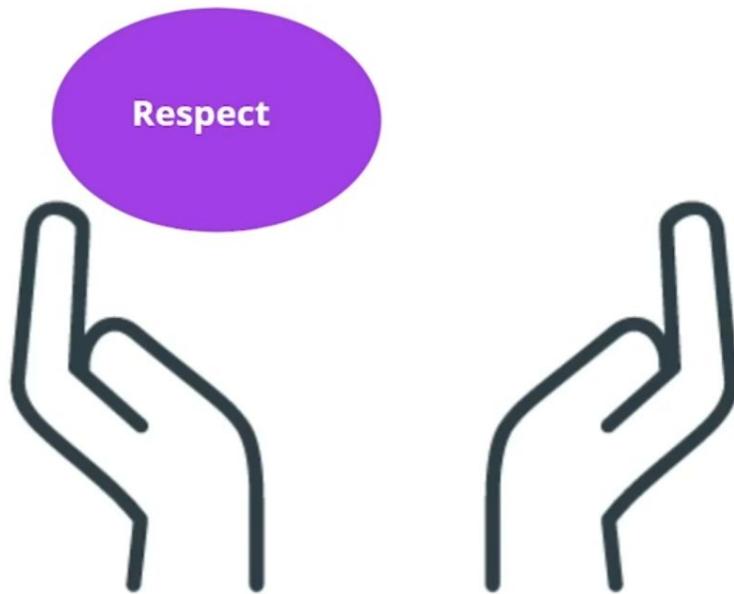
## Scrum Values



### Commitment

- Commitments build trust
- Scrum encourages reasonable and good faith approach to Sprint goals
- Every Team member personally commits

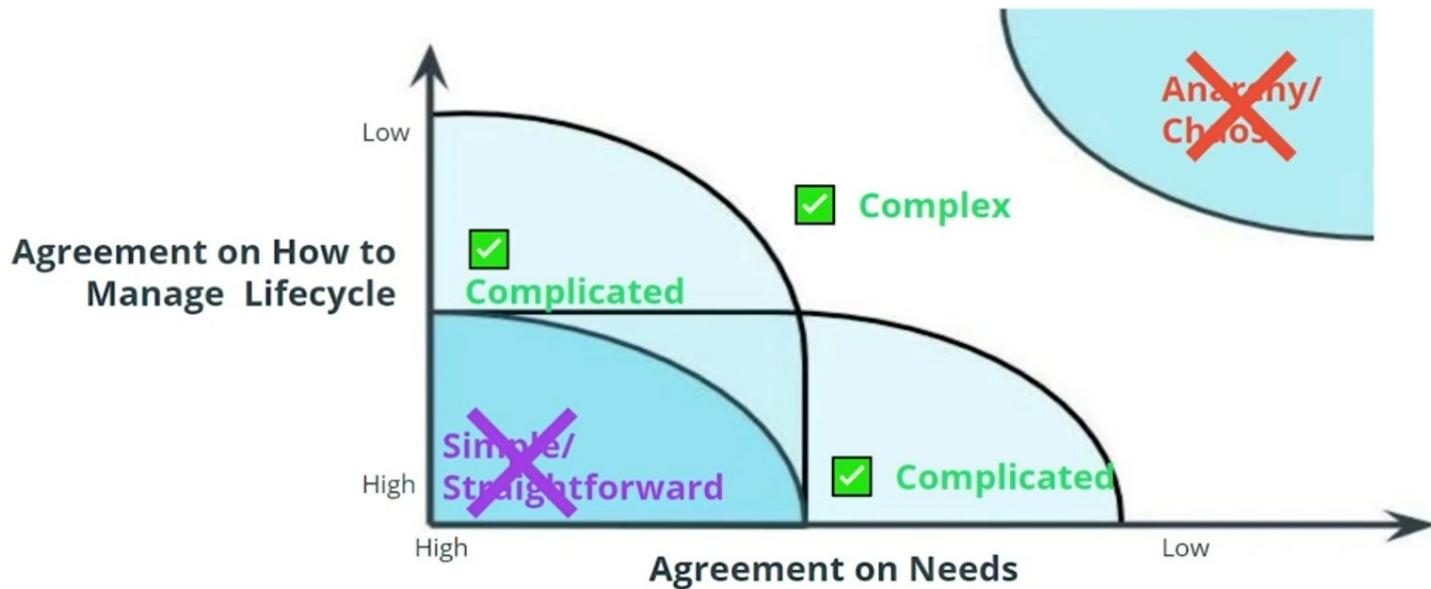
## Scrum Values



### Respect

- Foundational element for a high achieving Agile Team
- Elevates Agile Team's participation by including them in ceremonies and planning
- Increases Agile Team's motivation

## Stacey Diagram



Adapted from: Stacey RD. *Strategic management and organisational dynamics: the challenge of complexity*. 3rd ed. Harlow: Prentice Hall, 2002

# Scrum Ceremony: Sprint Review/Demo

## Purpose

Generate feedback and nurture collaboration

## Agenda

- Demonstrate the Sprint's work
- Discuss accomplishments
- Identify remaining items in the backlog
- 

## Timing

- Once at the end of each Sprint
- Timeboxed to 4 hours for a 1 month Sprint
- 2 hours for a 2 week Sprint

## Expected Outcome

- PO determines if deliverables are acceptable based on Acceptance Criteria and definition of Done
- Marketplace changes are considered
- PO and Team make adjustments

# Scrum Ceremony: Sprint Retrospective

## Purpose

Agile Team self-inspects and identifies opportunities for improvement

## Agenda

- Create a plan for improvement
- Discuss what went well and what to continue doing

## Timing

- After Sprint Review
- Timeboxed to 3 hours - but may be shorter

## Expected Outcome

- PO's feedback from Sprint Review is considered
- Improvements to process, tools and people are discussed

# Scrum Ceremony: Release Planning

## Purpose

Agile Team develops a Release Plan

## Agenda

- Discuss critical dates and milestones
- Coordinate with other teams
- Balance business value vs. quality

## Timing

- Only when there is a release
- Timeboxed to 20 minutes

## Expected Outcome

- Release date for delivery of feature to the customer base
- Based on input from the business units or users

## Scrum Ceremonies

---

- Focused meetings
- Specific purpose



Ceremony/Event

## Scrum Ceremonies



# Scrum Ceremony: Project Vision

## Purpose

Business Leaders identify the business need

## Agenda

- Discuss Project Goals
- Identify Sponsor
- Share Success Criteria
- Identify Assumptions, Constraints and Risks

## Timing

- Once per project
- Timeboxed to one hour

## Expected Outcome

- Craft a desired future state statement
- Clarify business value
- Lay out a high level path to accomplish objectives

# Scrum Ceremony: Daily Stand Up

## Purpose

Synchronize activities and create a short term plan for the next 24 hours.

## Agenda

- What did I do since the last time we met?
- What do I plan to accomplish today?
- What impediments am I encountering?

## Timing

- Timeboxed to 15 minutes
- Held every day at the same time
- Preferably in the morning and in the same location

## Expected Outcome

- Cultivates transparency
- Conversation remains focused
- No Problem Solving!

# Scrum Ceremony: Sprint Planning

## Purpose

Agile Team self-organizes and plans out the work to be performed in the Sprint.

## Agenda

- What are we committing to deliver?
- How will we complete the required work?
- What are the estimates?

## Timing

- Timeboxed to 8 hours or less
- More than one session
- Scrum Master ensures timebox is not exceeded

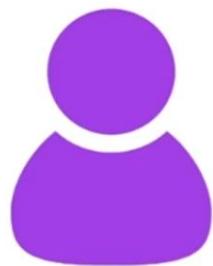
## Expected Outcome

- PO prioritizes backlog items in the backlog
- Agile Team decides how to carry out the work



## Scrum Roles and Artifacts

## Core Roles in The Scrum Framework



**Product Owner**

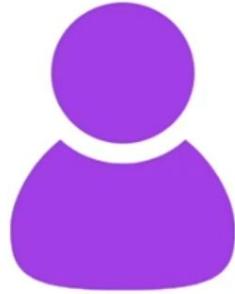


**Scrum Master**



**Agile Team**

## **Product Owner**



**PO**

- Maximize value delivery
- Prioritize and manage the Product Backlog

## Agile Team

- T-Shaped Skills
- Self-organized
- Accomplish the work to complete each iteration



Agile Team

## Scrum Master



Scrum Master

- Keeps Agile Team moving towards business objectives
- Removes blockages
- Facilitates Ceremonies

## Scrum Master



Scrum Master

- Servant Leader to Agile Team and PO
- Supports PO with keeping the Product Backlog refined
- Coach in the adoption of Scrum practices

## Scrum Artifacts



- Transparently communicate progress and value being delivered
- Allows for constructive feedback
- Easily understood to ensure everyone is on the same page

## Scrum Artifacts



Product  
Increment



Product  
Backlog



Sprint  
Backlog

## Scrum Artifact: Product Increment



Product  
Increment

- Clarifies what is included in a Sprint
- PO and Agile Team agree on Definition of Done

## Scrum Artifact: Product Backlog

- Prioritized list of work to be delivered
- PO prioritizes the Product Backlog
- Backlog items are dynamic and evolving



## Scrum Artifact: Sprint Backlog



Sprint  
Backlog

- Prioritized items for the Sprint
- A portion of the Product Backlog Items are included
- Agile Team is provided a Sprint Goal

## Scrum Artifact: Burndown Chart



- Communicates if Agile Team is on target
- Alerts Agile team about bottlenecks and problems
- Shows Agile Team's success

## Scrum Artifact: Burnup Chart

- Shows how much work has been completed
- Clearly illustrates the amount of work for a product or Sprint



## Scrum Artifact: Velocity Chart



- Displays the amount of value delivered in each Sprint
- Enables PO to predict future Sprint capacity

## Exercise: Set Up the Scrum Framework

- Core roles to fill
- Ceremonies to conduct
- Schedules, duration and agendas of ceremonies



## Three Core Roles in Scrum Exercise



Role	Responsibilities
Scrum Master	<ul style="list-style-type: none"><li>• Provides Scrum Framework coaching and guidance</li><li>• Facilitating the Agile Team's work</li><li>• Protecting from problems</li><li>• Servant Leader</li></ul>



## Three Core Roles in Scrum Exercise

Role	Responsibilities
Product Owner	<ul style="list-style-type: none"><li>• Voice of the Customer (VOC)</li><li>• Prioritize Backlog</li><li>• Share Business Decisions</li><li>• Provide Information</li></ul>

## Three Core Roles in Scrum Exercise



Role	Responsibilities	
Agile Team Member	<ul style="list-style-type: none"><li>• Creation and delivery of solutions:</li></ul> <p>Modeling                      Business Analysis Programming                 Release Activities Testing Front-end design</p>	

# Scrum Ceremonies



Scrum Ceremony	Purpose	Agenda	Schedule and Frequency
Retrospective	Scrum Team inspects itself and creates a plan for improvements	Discuss what went well during the Sprint, as well as opportunities for improvement	Occurrence: once per Sprint  Time-boxed to 3 hours per Sprint
Project Vision	Articulate the business need that is intended to be achieved	Discuss Goals, Sponsor, Vision, Success Criteria, Assumptions & Risks	Occurrence: once per project  Time-boxed to 1 hour

# Scrum Ceremonies



Scrum Ceremony	Purpose	Agenda	Schedule and Frequency
Release Planning	Develop a Plan that defines when sets of functionality or products will be delivered to the customer	Discuss critical dates and milestones, coordinate with dependent systems and balance business value versus quality	Occurrence: once per Release after initial creation Time-boxed to 20 minutes
Sprint Planning	Plan work to be performed in the Sprint	Create Sprint Backlog Task planning Task level estimation	Occurrence: a few times per Sprint Time-boxed to 8 hours per Sprint

# Scrum Ceremonies



Scrum Ceremony	Purpose	Agenda	Schedule and Frequency
Daily Stand Up	Synchronize activities and create a plan for next 24 hours	Each team member answers 3 questions:  What did I do since the last time we met?  What do I plan to accomplish today?  What impediments am I encountering?	Held daily (preferably in the morning)  Time-boxed to 15 minutes

# Scrum Ceremonies

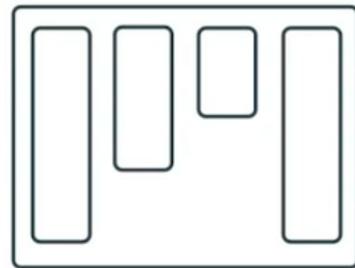


Scrum Ceremony	Purpose	Agenda	Schedule and Frequency
Sprint Review and Demo	Demonstrate product and discuss marketplace changes and backlog	Team presents completed Sprint deliverables to Product Owner, based on Acceptance Criteria and Definition of Done	Occurrence: once per Sprint  Time-boxed to 4 hours per Sprint

## The Kanban Framework

---

- Originated from lean product development
- Lean focuses on seven fundamental principles



## Lean Principles



*Set the  
foundation  
for Kanban*

# Lean Principles



## Eliminate Waste

- Does not add value to customer
- Does not improve quality
- Unnecessarily increases time or effort

# Lean Principles



## Build Quality In

- Good practices to prevent defects

# Lean Principles



## Create Knowledge

- Training and peer-to-peer knowledge transfer
- 'Do a little, show a little, learn a little'

# Lean Principles



## Defer Commitment

- Wait until the 'last responsible moment'
- Allows Agile Team to have time for innovation

# Lean Principles



## Deliver Fast

- Create product incrementally
- Build only what is needed
- Release product earlier to get feedback

# Lean Principles



## Respect People

- Feel trusted and valuable
- Solve problems for themselves
- Authority to effect outcomes

# Lean Principles



## Optimize the Whole

- Decrease barriers
- Decrease hand-offs
- Decrease Work in Process (WIP)

## Lean Principles



## Kanban

---

- Refers to a 'signboard'
- Kanban Board is instrumental



## Five Principles of Kanban

Visualize the Flow



### Visualize the Flow

- Important for
  - Organizing
  - Tracking
  - Optimizing



## Five Principles of Kanban

Visualize the Flow



Limit Work in Progress



### Limit Work in Progress (WIP)

- Respect the Team's ideal work capacity
- Limiting WIP helps:
  - Smooth flow
  - Reduce Lead times
  - Improve Quality
  - Delivery more frequently



# Five Principles of Kanban

Visualize the Flow



Limit Work in Progress



Manage Flow



## Manage Flow

- Issues are identified as soon as possible
- Helps to minimize lead times
- Improves delivery predictability



# Five Principles of Kanban

Visualize the Flow



Limit Work in Progress



Manage Flow



Make Policies Explicit



## Make Policies Explicit

- Explicit understanding of the process has benefits
- Discussions around issues take place in an objective manner



# Five Principles of Kanban

Visualize the Flow



Limit Work in Progress



Manage Flow



Make Policies Explicit



Improve Collaboratively



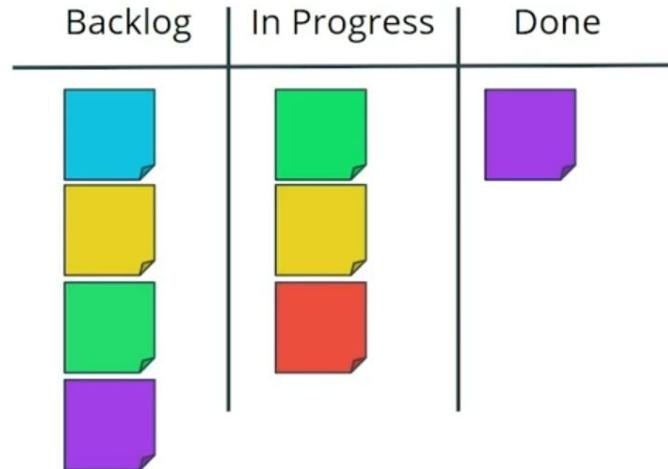
## Improve Collaboratively

- Agile Team must own processes
- Work collaboratively to improve the processes being utilized



## Kanban Board

- Plays a valuable role
- Articulates work items in various stages of product development



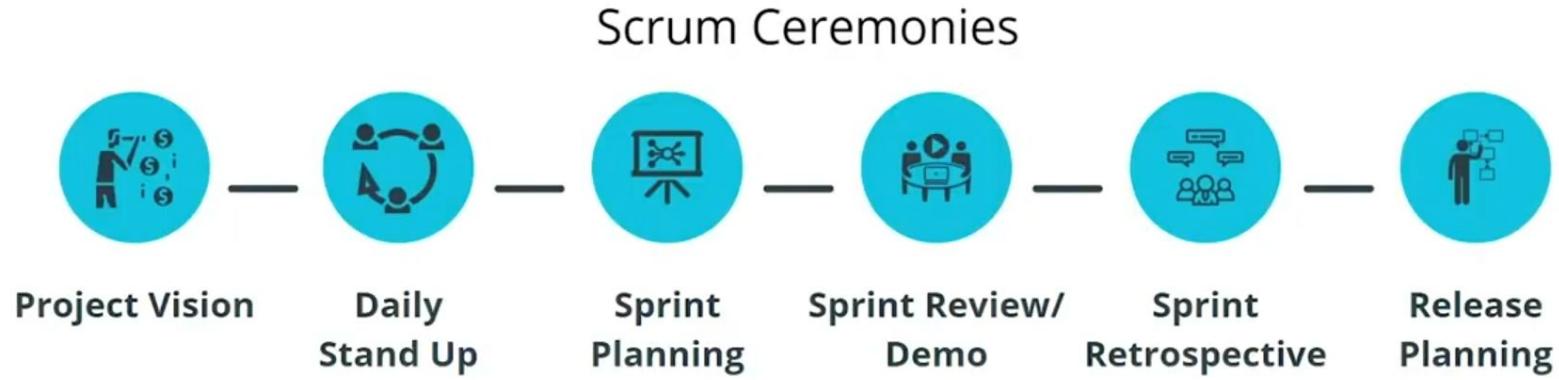
## Kanban Ceremonies

---

- Focused meetings
- Specific purpose



## Kanban vs. Scrum Ceremonies



## Kanban vs. Scrum Ceremonies



## Kanban vs. Scrum Ceremonies

### Kanban Ceremonies



Project Vision



Daily  
Stand Up



Review/  
Demo



Release  
Planning

# Kanban Ceremony: Project Vision

## Purpose

Business Leaders identify the business need

## Agenda

- Discuss Project Goals
- Identify Sponsor
- Share Success Criteria
- Identify Assumptions, Constraints and Risks

## Timing

- Once per project
- Timeboxed to one hour

## Expected Outcome

- Craft a desired future state statement
- Clarify business value
- Lay out a high level path to accomplish objectives

# Kanban Ceremony: Daily Stand Up

## Purpose

Synchronize activities and create a short term plan for the next 24 hours.

## Agenda

- What did I do since the last time we met?
- What do I plan to accomplish today?
- What impediments am I encountering?

## Timing

- Timeboxed to 15 minutes
- Held every day at the same time
- Preferably in the morning and in the same location

## Expected Outcome

- Cultivates transparency
- Conversation remains focused
- No Problem Solving!

# Kanban Ceremony: Review/Demo

## Purpose

Generate feedback and nurture collaboration

## Agenda

- Demonstrate the Iteration's work
- Discuss accomplishments
- Identify remaining items in the backlog
- 

## Timing

- Once at the end of each Iteration
- Timeboxed to 4 hours for a 1 month Iteration
- 2 hours for a 2 week Iteration

## Expected Outcome

- PO determines if deliverables are acceptable based on Acceptance Criteria and definition of Done
- Marketplace changes are considered
- PO and Team make adjustments

# Kanban Ceremony: Release Planning

## Purpose

Agile Team develops a Release Plan

## Agenda

- Discuss critical dates and milestones
- Coordinate with other teams
- Balance business value vs. quality

## Timing

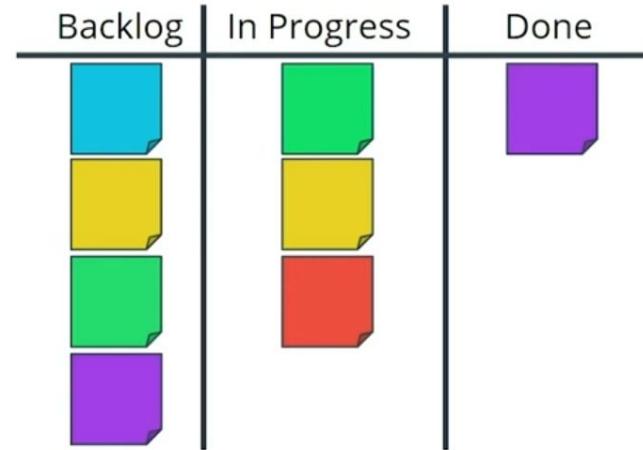
- Only when there is a release
- Timeboxed to 20 minutes

## Expected Outcome

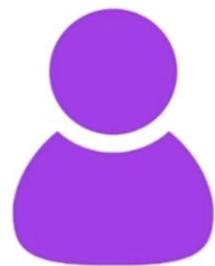
- Release date for delivery of feature to the customer base
- Based on input from the business

## Where Does Kanban Make Sense?

- Projects that involve a large quantity of small activities
- Suitable for ad-hoc work as well



## Core Roles in The Kanban Framework



**Product Owner**



**Facilitator**



**Agile Team**

## Kanban Artifacts



Kanban Board



Cumulative  
Flow Diagram

## Kanban Artifact: Kanban Board

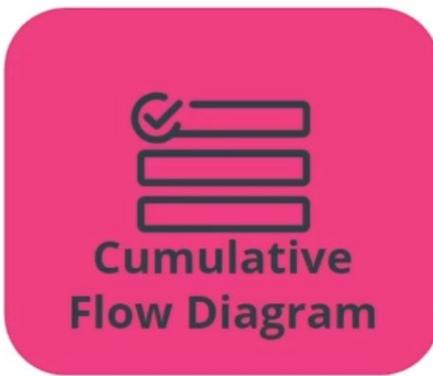


Kanban Board

- Plays a valuable role
- Articulates work items in various stages of product development

## Kanban Artifact: Cumulative Flow Diagram

- Problem areas are easily detected
- Changes can be made so work can continue efficiently
- Shows the total amount of WIP



## The XP Framework

---

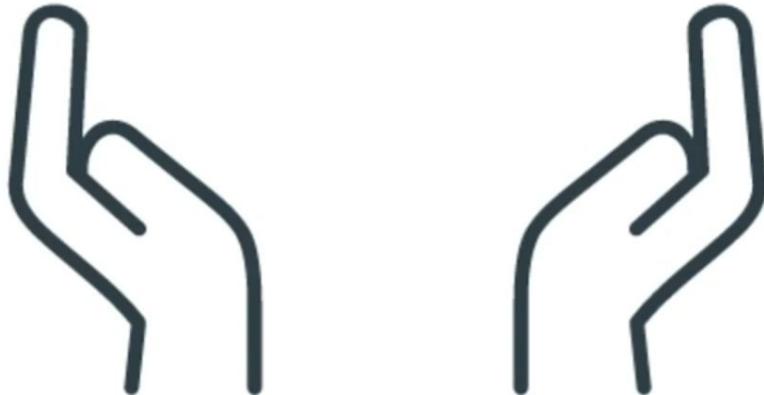
- Extreme Programming (XP) has its roots in Software Development
- Focuses on best practices in software development



## XP Values



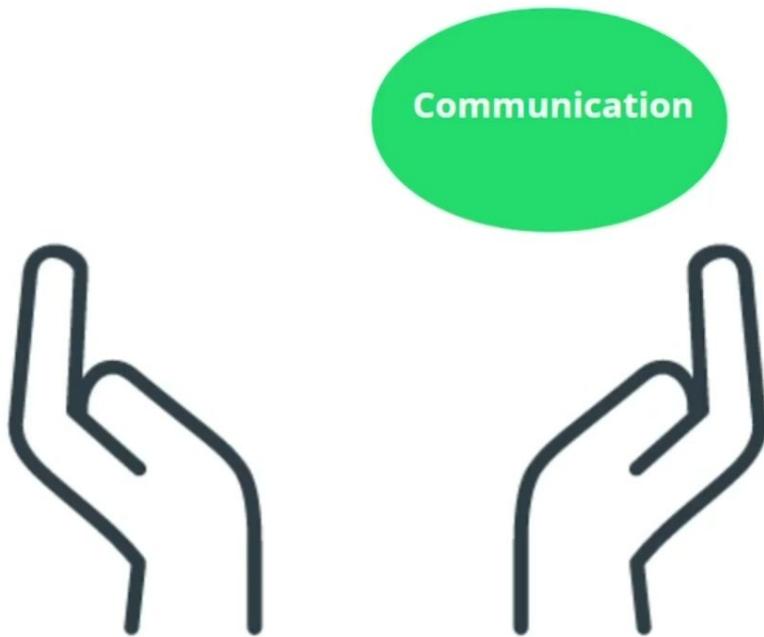
## XP Values



### Simplicity

- Reduce unnecessary complexity
- Minimize extra features
- Maximize value for the investment

## XP Values



### Communication

- Have transparency of the deliverables
- Daily Stand Ups are pivotal

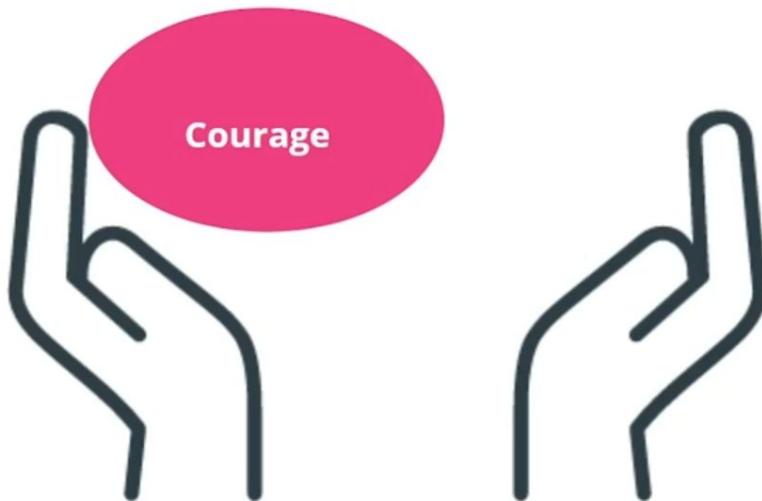
## XP Values



### Feedback

- Taken seriously in each iteration
- An opportunity to demonstrate working results
- Failing fast allows team to make changes proactively

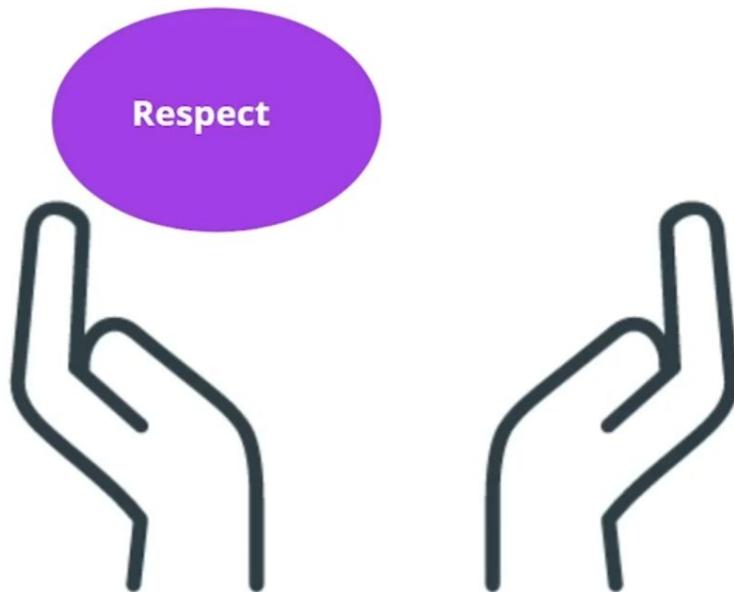
## XP Values



### Courage

- Allows a team to bravely tell the truth
- Paired programming requires courage to share code
- Adapt to shared coding standards

## XP Values



### Respect

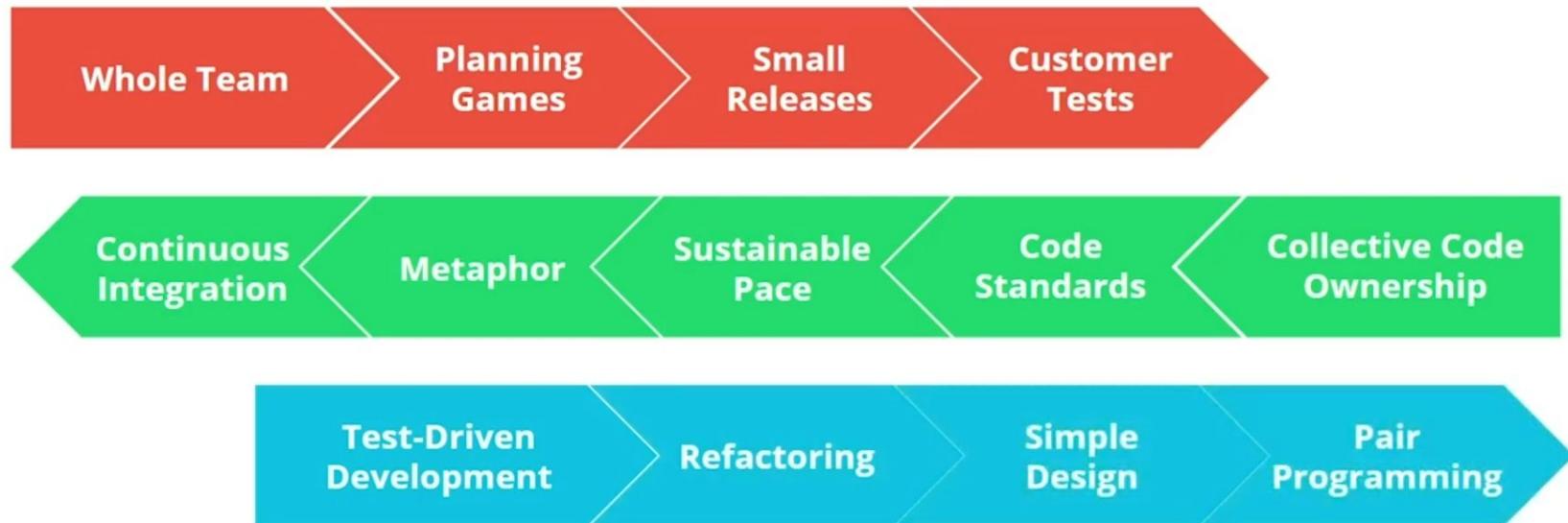
- Foundational element for a high achieving Agile Team
- Each team member is accountable for successful product delivery
- Practiced in paired programming

## XP Values



Set the tone for  
XP's 13 Practices

## XP Practices



## XP Practice: Whole Team



Whole Team

- Encourages co-location, or use of collaboration tools for remote teams
- XP encourages role blending with T-shaped skills
- Allows for efficient flow of information

## XP Practice: Planning Games

- XP has two main planning ceremonies
  - Release Planning
  - Iteration Planning



Planning Games

## XP Practice: Small Releases



Small Releases

- Encouraged in XP at both the iteration and release level
- Built in quality is ensured by leveraging best practices

## XP Practice: Customer Tests

- PO works with customer to describe test criteria
- The team then ideally builds automated tests to confirm that acceptance criteria are met



**Customer Tests**

## XP Practice: Collective Code Ownership



Collective Code  
Ownership

- No single person owns a piece of work
- This strengthens the product
- Quality is enhanced
- Reduces risk of loss of knowledge

## XP Practice: Code Standards

- Everyone codes to a consistent standard
- The code should as if it is written under a uniform standard



Code  
Standards

## XP Practice: Sustainable Pace



Sustainable Pace

- Highest level of productivity maintained at a sustainable pace
- Overtime is counterproductive
- Optimal delivery of long term value requires a sustainable rate

## XP Practice: Metaphor

- Adhere to a set of easy to remember standards for naming convention
- Facilitates communication
- Ensures everyone understands the terms clearly



Metaphor

## XP Practice: Continuous Integration



Continuous  
Integration

- Software builds regularly integrated
- Verify that it compiles and works as a collective code base
- Integration keeps all team members on the same page

## XP Practice: Test-Driven Development

- Team writes tests before coding
- Each small release is tested prior to being released
- Allows for a cleaner application in the long run



Test-Driven  
Development

## XP Practice: Refactoring



Refactoring

- Improve the design of existing code without affecting functionality
- Keeps design cleaner to maintain
- Makes it easier to develop new functionality

## XP Practice: Simple Design

- Optimal design is simplest one
- Recommended design runs all tests and delivers business value
- XP proposes doing what is needed to meet current requirements



Simple Design

## XP Practice: Pair Programming



Pair  
Programming

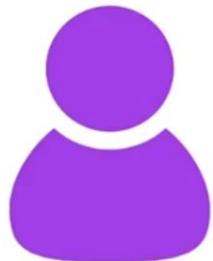
- Team members should work on product deliverables in pairs
- Produces higher quality code and costs less
- Knowledge is shared across team

## XP Is More Stringent Than Other Agile Frameworks

- None of the practices are optional
- Practices are very specific about *how* the work gets done
- Strongly oriented to software development



## Core Roles in The XP Framework



**Customer**



**Testers**

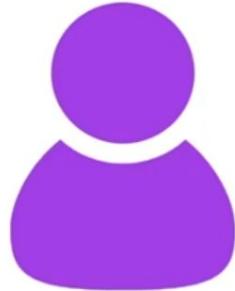


**Coach**



**Programmers**

## **Customer**



**Customer**

- Maximize value delivery
- Prioritize and manage the Product Backlog

## Programmers

- Collaborate in practicing XP Core values
- Includes designer, senior programmer and architect
- T-shaped skills



Programmers

## Coach



Coach

- Plays a pivotal role in team's success
- Helps team learn and follow XP practices

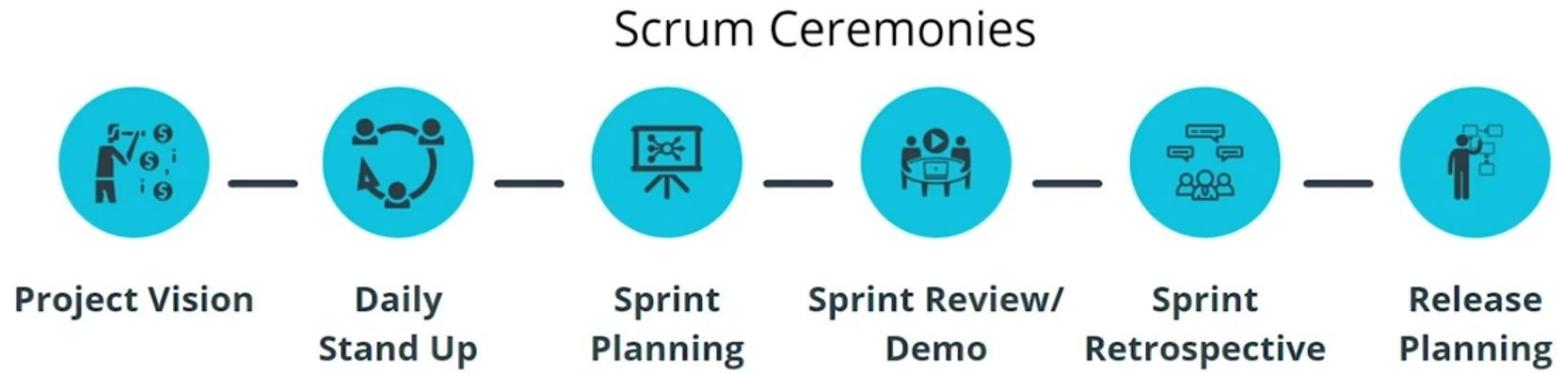
## Testers



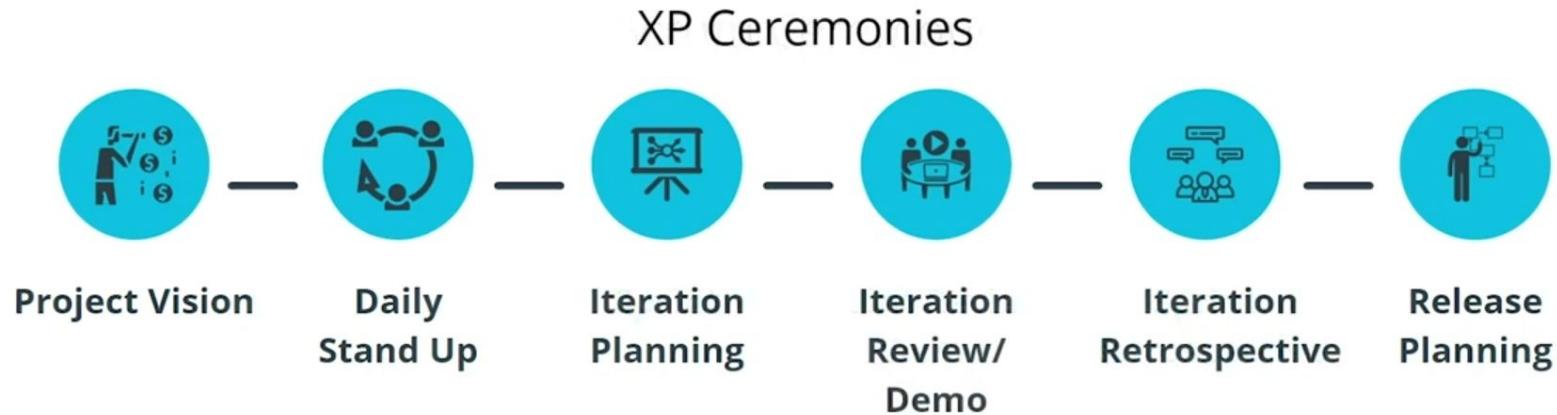
Testers

- Assist customers in creating acceptance criteria
- Execute test cases
- Share outcomes with programmers

## XP vs. Scrum Ceremonies



## XP vs. Scrum Ceremonies



# XP Ceremony: Project Vision

## Purpose

Business Leaders identify the business need

## Agenda

- Discuss Project Goals
- Identify Sponsor
- Share Success Criteria
- Identify Assumptions, Constraints and Risks

## Timing

- Once per project
- Timeboxed to one hour

## Expected Outcome

- Craft a desired future state statement
- Clarify business value
- Lay out a high level path to accomplish objectives

# XP Ceremony: Daily Stand Up

## Purpose

Synchronize activities and create a short term plan for the next 24 hours.

## Agenda

- What did I do since the last time we met?
- What do I plan to accomplish today?
- What impediments am I encountering?

## Timing

- Timeboxed to 15 minutes
- Held every day at the same time
- Preferably in the morning and in the same location

## Expected Outcome

- Cultivates transparency
- Conversation remains focused
- No Problem Solving!

# XP Ceremony: Iteration Review/Demo

## Purpose

Generate feedback and nurture collaboration

## Agenda

- Demonstrate the Iteration's work
- Discuss accomplishments
- Identify remaining items in the backlog
- 

## Timing

- Once at the end of each Iteration
- Timeboxed to 4 hours for a 1 month Iteration
- 2 hours for a 2 week Iteration

## Expected Outcome

- PO determines if deliverables are acceptable based on Acceptance Criteria and definition of Done
- Marketplace changes are considered
- PO and Team make adjustments

# XP Ceremony: Iteration Retrospective

## Purpose

Programmers self-inspect and identify opportunities for improvement

## Agenda

- Create a plan for improvement
- Discuss what went well and what to continue doing

## Timing

- After Iteration Review
- Timeboxed to 3 hours - but may be shorter

## Expected Outcome

- PO's feedback from Sprint Review is considered
- Improvements to process, tools and people are discussed

# XP Ceremony: Release Planning

## Purpose

Programmers develop a Release Plan

## Agenda

- Discuss critical dates and milestones
- Coordinate with other teams
- Balance business value vs. quality

## Timing

- Only when there is a release
- Timeboxed to 20 minutes

## Expected Outcome

- Release date for delivery of feature to the customer base
- Based on input from the business units or users

## XP Artifacts

---

- Product Increment
- Product Backlog
- Iteration Backlog
- Burndown Chart
- Burnup Chart
- Velocity Chart
- Cumulative Flow Diagram



## Exercise: Differences Between Kanban and XP

---

- Identify key differences, including:
  - Roles
  - Ceremonies
  - Practices

## Three Core Roles in Kanban

Role	Responsibilities
Facilitator	<p>Provides Kanban Framework coaching and guidance</p> <p>Facilitating the Agile Team's work</p> <p>Protecting from problems</p> <p>Servant Leader</p>

## Three Core Roles in Kanban

Role	Responsibilities
Product Owner	<ul style="list-style-type: none"><li>Voice of the Customer (VOC)</li><li>Prioritize Backlog</li><li>Share Business Decisions &amp; Provide Information</li></ul>

## Three Core Roles in Kanban Exercise

Role	Responsibilities		
Agile Team Member	<p>Responsible for Creation and Delivery of solutions</p> <p>Modeling                      Business Analysis</p> <p>Programming                  Release Activities</p> <p>Testing                        Others</p> <p>Front-end design</p>		

## Four Core Roles in XP

Role	Responsibilities
Programmer	<ul style="list-style-type: none"><li>• Pair programming, test-driven development and continuous integration</li><li>• Designer, Senior Programmer, Architect</li><li>• Assist with testing when time permits</li></ul>

## Four Core Roles in XP

Role	Responsibilities
Tester	<ul style="list-style-type: none"><li>• Assist customer in creating acceptance criteria</li><li>• Execute test cases</li><li>• Communicate outcomes</li></ul>

## Four Core Roles in XP

---

Role	Responsibilities
Coach	<ul style="list-style-type: none"><li>• Servant Leader</li><li>• Coaches team on XP practices</li></ul>

## Four Core Roles in XP

Role	Responsibilities
Customer	<ul style="list-style-type: none"><li>• Works with business representatives to maximize value</li><li>• Responsible for prioritizing backlog</li></ul>



## XP Ceremonies

Scrum Ceremony	Purpose	Agenda	Schedule and Frequency
Retrospective	XP Team inspects itself and creates a plan for improvements	Discuss what went well during the Iteration, as well as opportunities for improvement	Occurrence: once per Sprint Time-boxed to 3 hours per Iteration
Iteration Planning	Plan work to be performed in the Iteration	Create Iteration Backlog  Task planning  Task level estimation	Occurrence: a few times per Iteration Time-boxed to 8 hours per Iteration

## XP is More Stringent

- XP has 13 detailed practices
- Kanban just focuses on flow and WIP



## Challenges With XP



- Some organizations get excited about the rigor, transparency and collaboration in XP

## Challenges With XP



- Some organizations get excited about the rigor, transparency and collaboration in XP
- Others struggle and get frustrated with XP

## Challenge #1



*It is software centric*

- Different from other Agile Frameworks
- Other Agile Frameworks feel like a leaner version of project management
- XP requires a comfort level with stringent software centric processes

## Challenge #2



- Coaches are familiar with software development
- Have the right personality to facilitate
- Such a combination is rare and expensive

*XP Coaches are tough to recruit*

## Challenge #3



*XP is complex*

- XP can feel anti-Agile
- Takes patience to develop and maintain a successful XP team
- More practices than many other Agile Frameworks

## Exercise: Recommend an Agile Framework

- Characteristics
- Justification



## Final Exercise

Characteristic	Justification
Need for time-boxed Iterations/Sprints	This is a complex project and time-boxed Sprints will allow for regular demos and proactive feedback from the business to ensure the project is heading in the right direction.

## Final Exercise

---

Characteristic	Justification
Need for Retrospectives	Sprint Retrospectives will ensure the Agile team is working towards guided continual improvement to help such a complex project successfully deliver the anticipated business value.

## Final Exercise

---

Characteristic	Justification
Need for self-directed and self-organized teams	The Scrum Framework insists on the Agile team being given autonomy to work on the prioritized and assigned Features and Functionality.

## Why Not Kanban or XP?



Kanban is too simplistic



XP is too heavy

## Lesson Recap

---

- Why Frameworks?
- Scrum
- Kanban
- XP



**Agile  
Frameworks**

# Course Recap



## Why Agile?

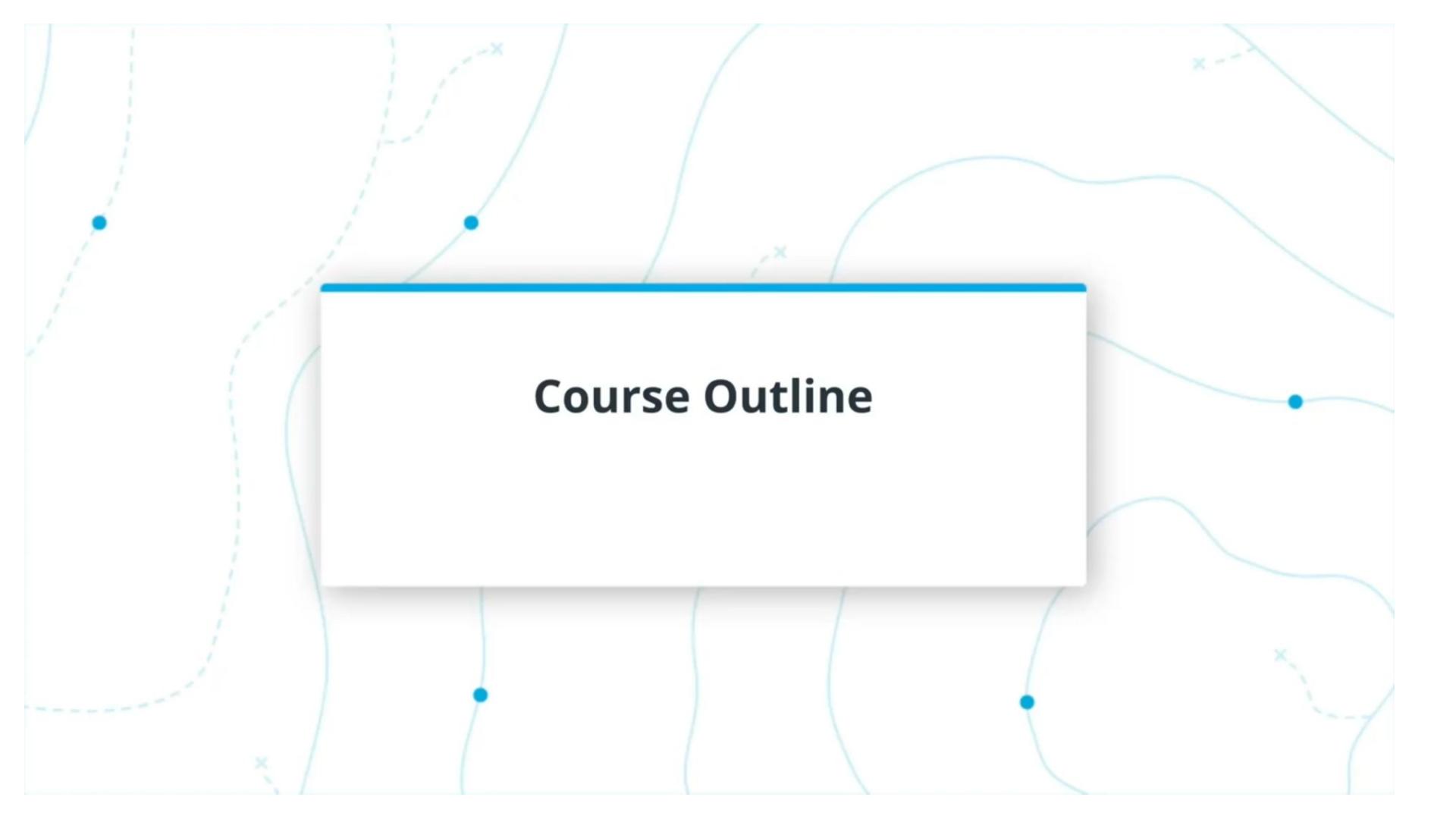
- The Agile Mindset
- The Agile Manifesto
- Agile vs. Waterfall

## Agile Teams

- High Performing Teams
- Size, Structure and Skills
- Agile Governance

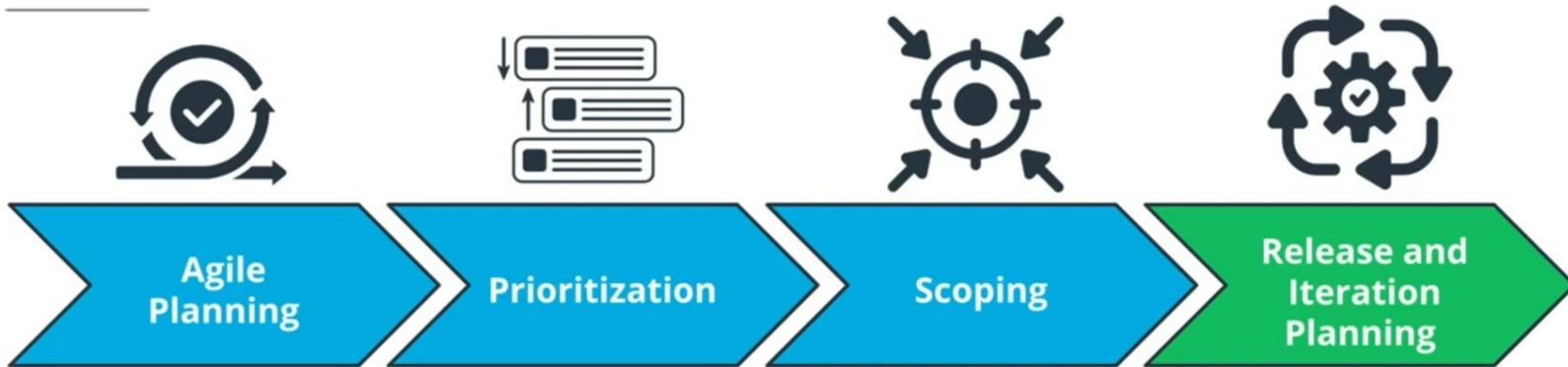
## Agile Frameworks

- Why Frameworks?
- Scrum
- Kanban
- XP



# Course Outline

# What We Will Cover



- Intro to Planning
- Vision and Roadmaps
- Features, Epics, User Stories, and Acceptance Criteria
- Intro to Backlogs
- Managing and Organizing the Backlog
- Prioritization Techniques
- Intro to Scoping and Estimation
- Estimation Techniques
- Defining "Done"
- Intro to Release and Iteration Planning
- Agile Techniques
- Scrum Specifics

## Learning Objectives

---

By the end of the course you will be able to...

- Construct a product roadmap to deliver continuous value
- Create a clear and concise vision statement to guide the development of the MVP
- Develop a release plan using appropriate scoping, and intentionality
- Apply prioritization techniques for effective value-driven delivery
- Write user stories using an Agile mindset to depict the user journey and set acceptance criteria
- Plan a value-driven Minimum Viable Product (MVP)

## Agile Planning Tools

---

- There are several software tools that companies and teams leverage to enable their agility
- I will be teaching this course from a tool agnostic view and you will be able to take what you have learned and apply it in your environment

## Agile Planning Perspective

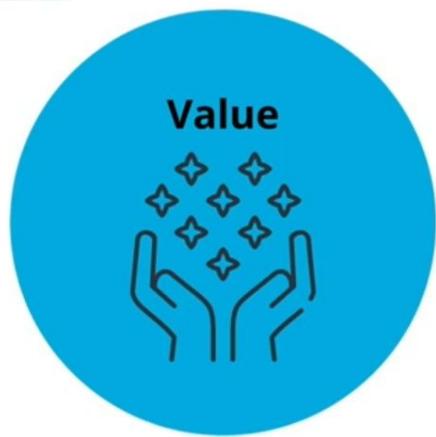
- This course is largely taught from the perspective of software product development
- Even though Agile was initially used for software development teams, today teams across all arenas and industries leverage Agile to deliver value
- At the end of this course you will have the knowledge needed to start applying Agile planning and prioritization practices within your teams or company



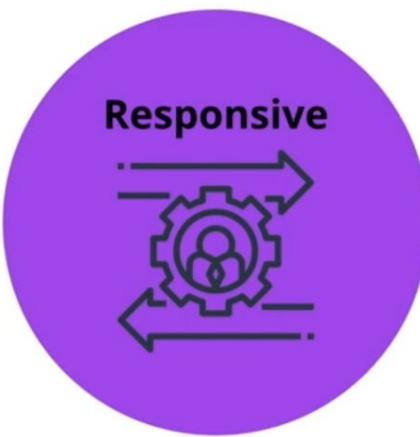
# Introduction to Delivering Value: Agile Planning and Prioritization

## Introduction

---



Value



Responsive



Feedback

## Essential Components

---

- Delivering value keeps customers engaged and satisfied
- Planning is essential because it allows teams to respond to change
- Prioritizing work helps the team focus on delivering the most valuable features

## The Outcomes of Planning and Prioritizing

---

- Increases Transparency
- Increases Quality
- Increases Communication

## Delivering Value: Agile Planning & Prioritization Roles



### Product Owner

Defines the “what” and sets priority



### Scrum Master

Coach and Facilitator



### Development Team

Determines the “how” and builds the solution



### Customer

Receives the product and gives feedback

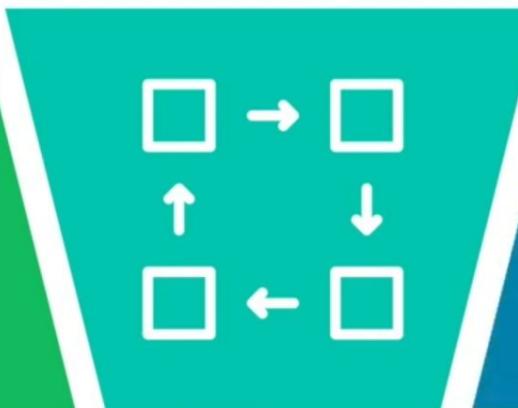
## Using the Agile Approach to Planning

- Agile can be applied in many different situations.
- The Agile approach to planning is most beneficial for teams or organizations that are looking to improve the quality of the products they deliver
- The Agile approach to planning is also helpful for teams or organizations that need to change more efficiently
- Agile is most often applied to software development products and operational processes

# The Evolution of Agile



Avoid Wasting Time



Iterative Process



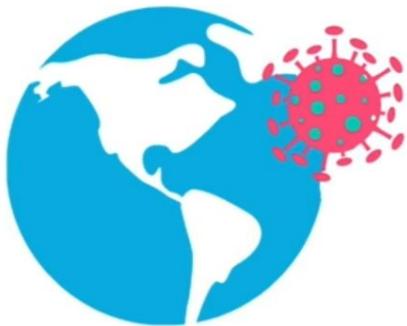
Deliver Value



## Project: Minimum Viable Product

## The Scenario

Worldwide Pandemic



- You work at the Centers for Disease Control (CDC) on infectious disease tracking & prevention
- There is currently a worldwide virus pandemic and there is no vaccine or cure
- The CDC wants to know where to apply funding and where to send test kits
- The CDC also wants to be able to track the status of vaccine to cure creation

## Your Goal

Develop a Minimum Viable Product Plan for a Tracking Product

### Consider The Product Requirements to



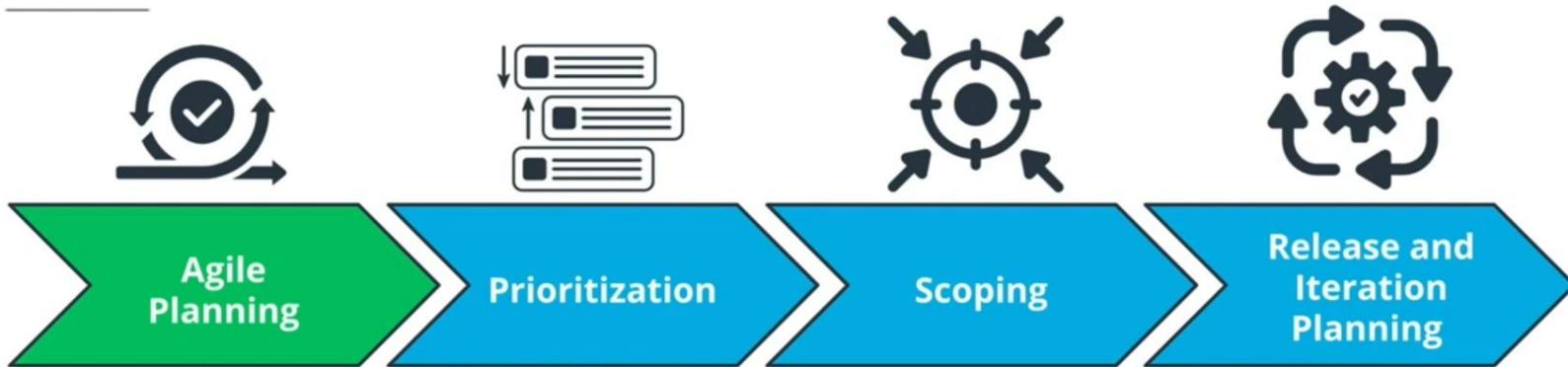
- Create a Vision
- Design Roadmap
- Write User Stories and Develop Acceptance Criteria
- Create a Release Plan and MVP

## Lesson Overview

---

- Planning is iterative, never-ending process that focuses on delivering value
- Agile planning improves product quality and allows teams to manage change more effectively
- You will build a final project that results in the plan for a Minimal Viable Product (MVP)
- You will be able to improve efficiency by applying Agile Planning and Prioritization within your team or organization

## Lesson Overview



- Intro to Planning
- Vision and Roadmaps
- Features, Epics, User Stories, and Acceptance Criteria

## Learning Objectives

---

By the end of the lesson you will be able to...

- Explain the difference between Agile and waterfall planning.
- Explain the advantages of Agile planning in software development.
- Describe a MVP and the importance of the MVP concept.
- Identify and define a vision for a project/product.
- Understand the purpose of a roadmap for each team member's role.

## Learning Objectives

---

By the end of the lesson you will be able to...

- Construct a product roadmap to deliver continuous value.
- Develop and define user stories.
- Differentiate among Features, Epics, and User Stories.
- Define acceptance criteria to identify the completion of a User Story.



## Why Does Agile Planning Matter?

## What are We Working On?

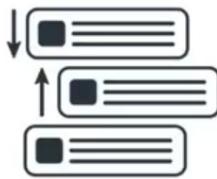
---



Many different requirements, components, pieces, and parts to make the product successful.

## What are We Working On?

---



**WHAT** is the highest priority?

## What are We Working On?

---

*The goal is to always deliver as much value  
as possible to the customer.*

## Agile Manifesto Relationship

Responding to **change**  
over  
following a **plan**

Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

## Agile Manifesto Relationship

Customer  
**collaboration**  
over  
contract **negotiations**

Adapted from: [www.agilemanifesto.org](http://www.agilemanifesto.org)

## **Value: What Does The Customer Need?**

---

- What value does the customer need or require?
- What is important to the customer?
- What problem does the customer have?
- What concerns the customer?
- What is limiting the customer?



## **Iterative**

---

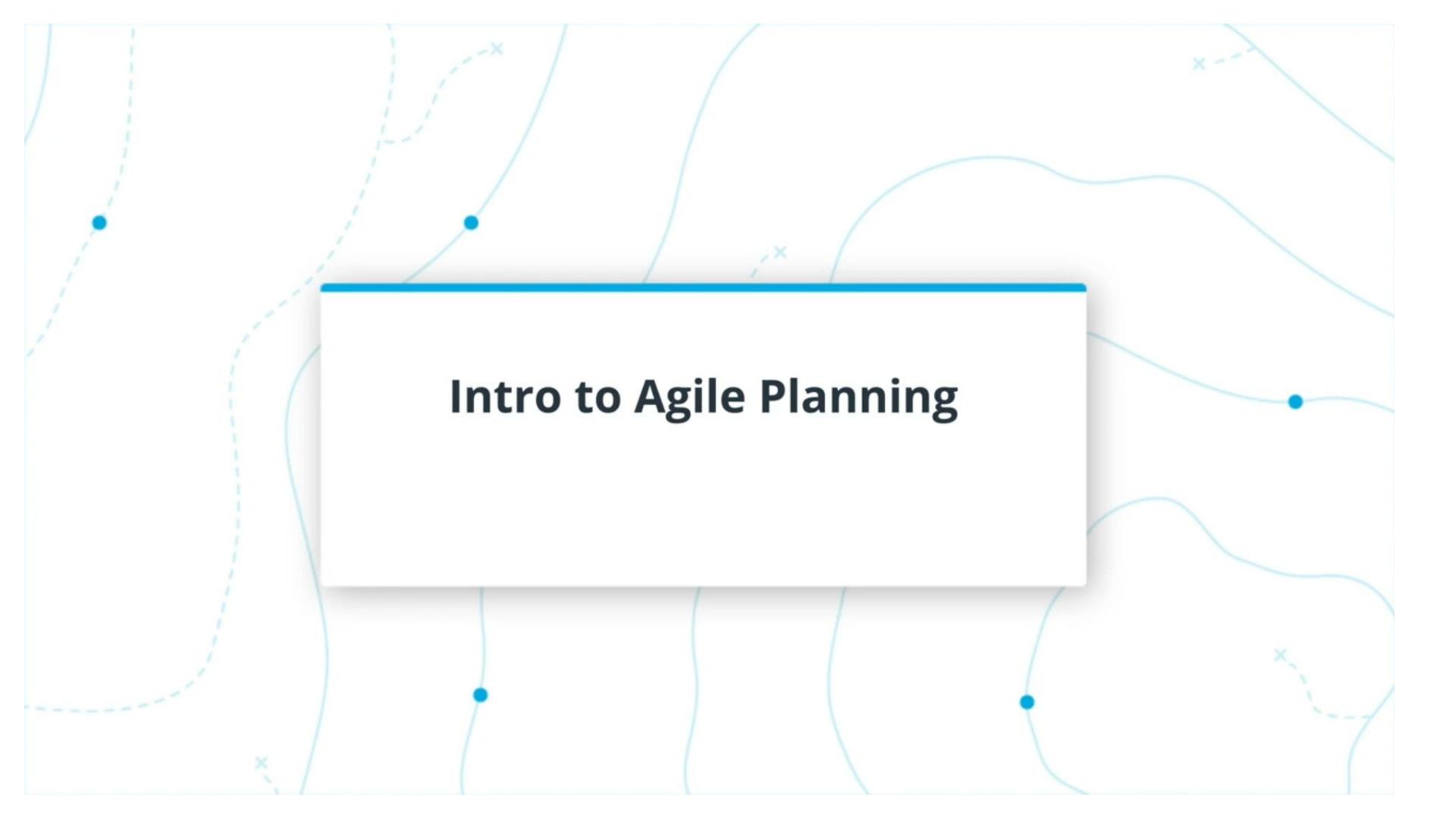
- Planning is iterative
- Done repeatedly

## **Incremental**

- Development is done incrementally.



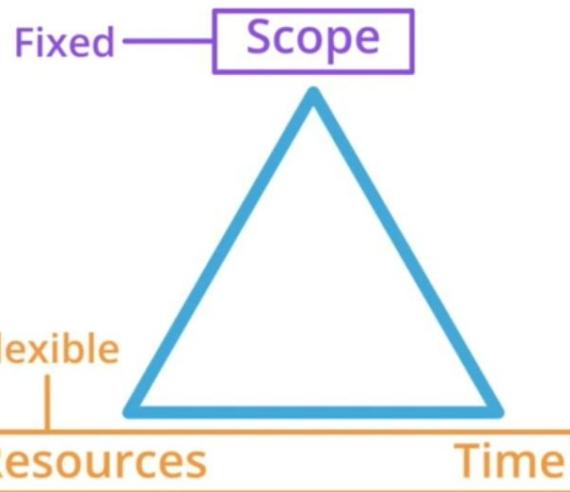
**What value does our customer need?**



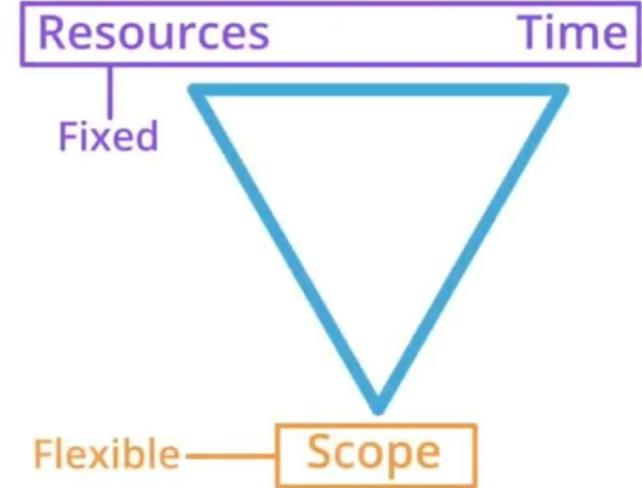
# Intro to Agile Planning

## Waterfall vs. Agile

### Waterfall



### Agile



## Agile vs. Waterfall

	Waterfall	Agile
<b>Requirements</b>	Gathered upfront	Gathered iteratively
<b>Planning</b>	All at once	Iteratively
<b>Delivery</b>	All at once	Iteratively
<b>Time</b>	Flexible	Fixed
<b>Budget</b>	Flexible	Fixed
<b>Customer</b>	The Beginning & End	Actively Engaged

## Waterfall Requirements



### Shall Statements

*Pilot shall be able to fly plane 10,000 feet in the air. Interior cabin pressure between 12 and 11 psi at cruise altitude.*

## Agile Requirements



### User Stories

*As an airline customer I can download the mobile app so that I can change my seat selection.*

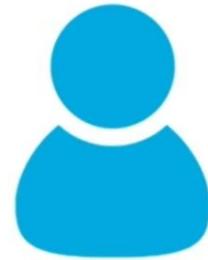
## Roles

---



### Product Owner

Defines the “what” and sets priority



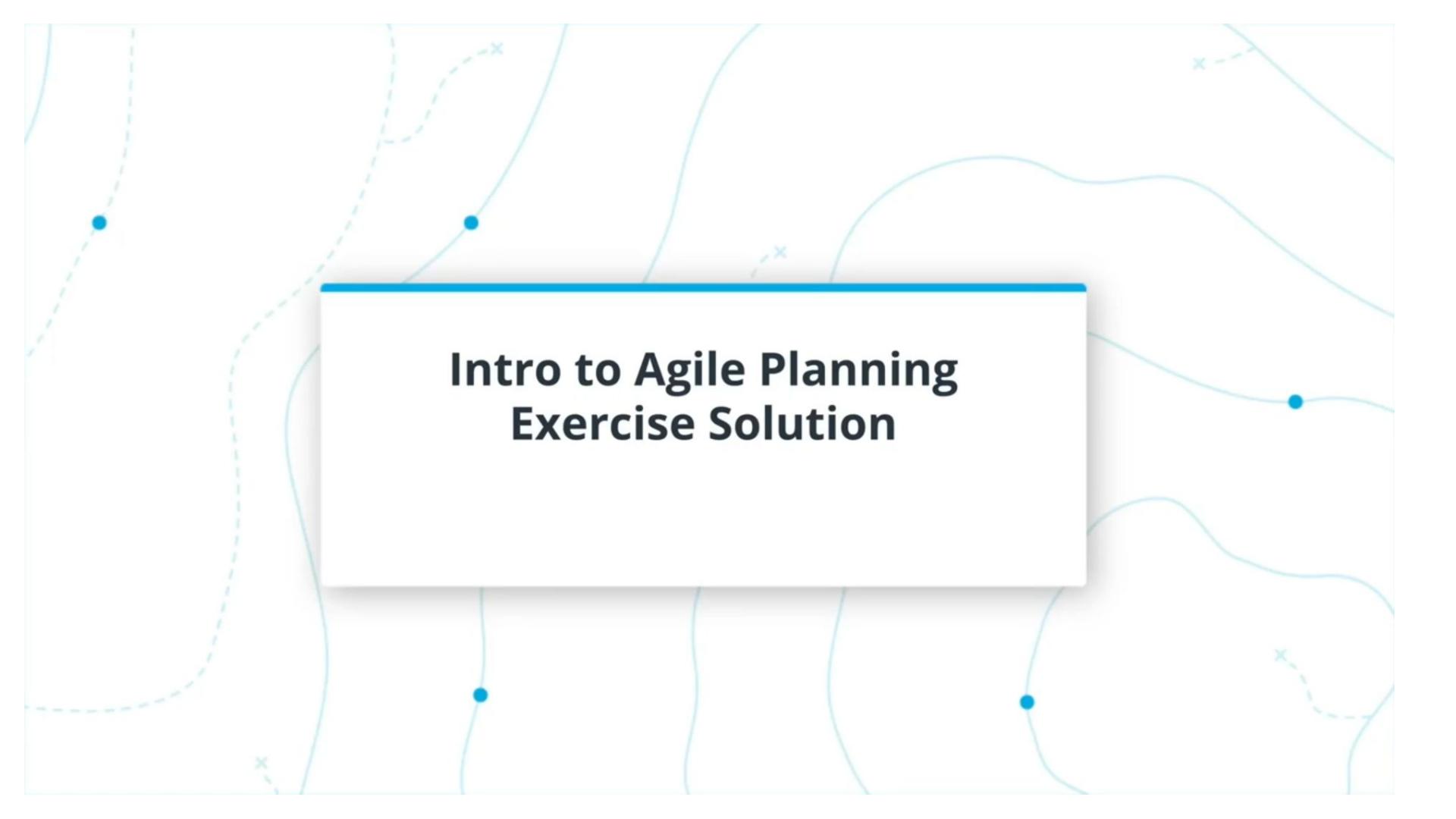
### Scrum Master

Coach and Facilitator



### Development Team

Determines the “how” and builds the solution



## **Intro to Agile Planning Exercise Solution**

## Agile Planning: Dating Product Development Team

You are the Scrum Master for the development team

- You are facilitating a planning session with the team and product owner
- 100 user stories in the backlog for the team to review in the session
- All of the stories have specific details
- Stories tell the development team how to implement
- All of the stories are critical and must be completed within the next 3 weeks

## Example 1

---

*A user story should not tell the development team “how” to implement the solution. The user story and product owner are presenting “what” needs to be done and only the development team (the people doing the work) determine the best way to do the work.*

## Example 2

---

*The product owner did not prioritize the user stories. All of the stories can't be critical. The product owner must prioritize the user stories according to the value they provide the customers. The team and product owner can negotiate how much work can get done within the 3 week timeframe.*

### Example 3

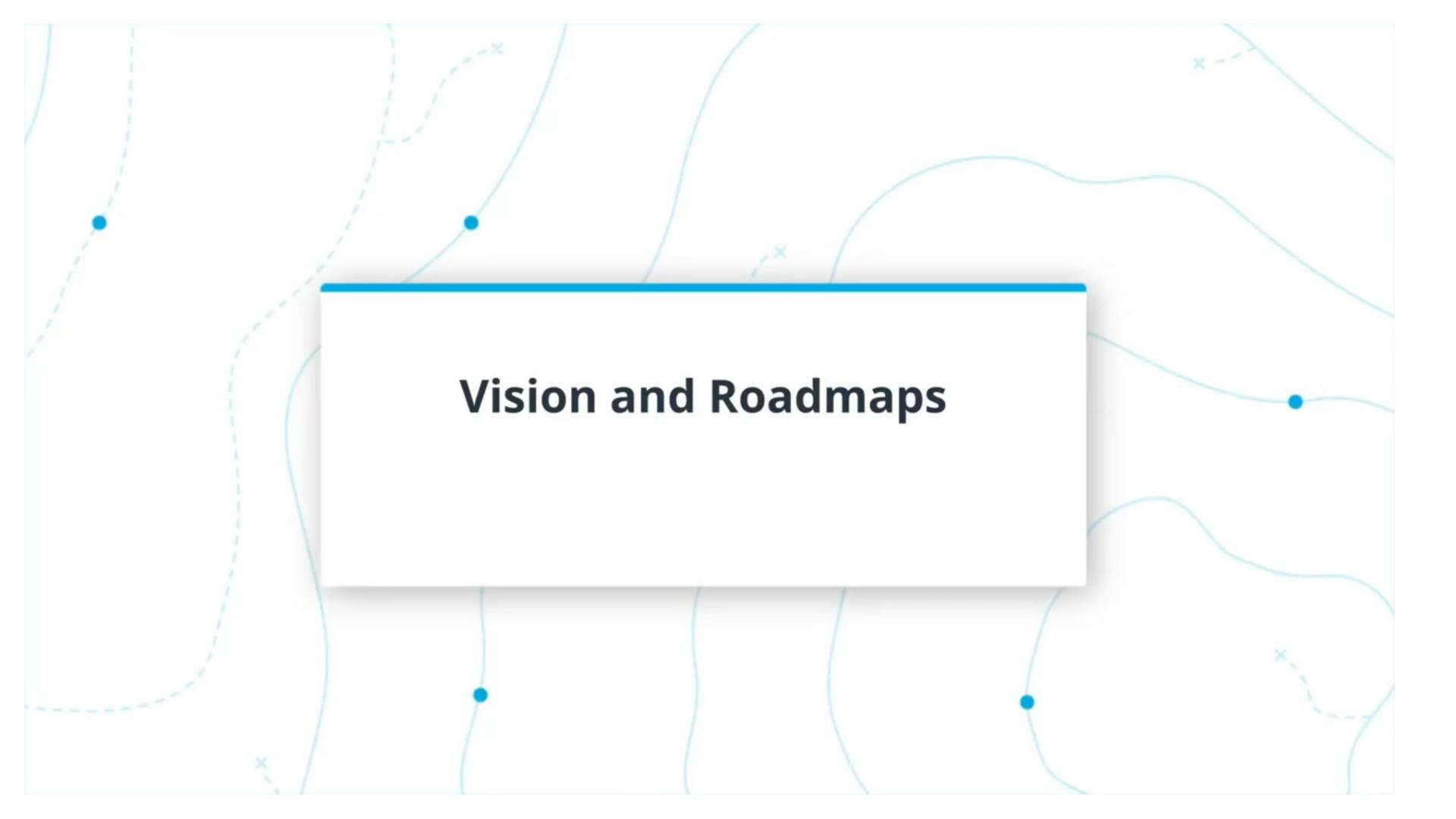
---

*100 stories are too many stories for the team to review in one planning session. The Product Owner should only have the team focus on the stories that will bring the highest near-term value.*

## Example 4

---

*Only the stories that are prioritized at the top of the backlog need to be detailed. The product owner has wasted time by detailing everything in the backlog because there will be change as development progresses.*



## Vision and Roadmaps

## Product Vision

---

- Tells the “why” behind the product
- Guides the direction of the product
- Identifies the outcome of the product
- Identifies who will use the product



## What and Who in the Vision Overview

---

*A good product vision statement tells **what** the product is attempting to achieve and **who** will benefit.*

## Vision Statement Example

---

**IKEA**

*To create a better everyday life for the many people.*

Source: [https://www.ikea.com/ms/en\\_JP/about\\_ikea/the\\_ikea\\_way/our\\_business\\_idea/index.html](https://www.ikea.com/ms/en_JP/about_ikea/the_ikea_way/our_business_idea/index.html)

## Vision Statement Example

---

**IKEA**

*To create a better everyday life for the many people.*

What

Who

Source: [https://www.ikea.com/ms/en\\_JP/about\\_ikea/the\\_ikea\\_way/our\\_business\\_idea/index.html](https://www.ikea.com/ms/en_JP/about_ikea/the_ikea_way/our_business_idea/index.html)

## Vision Statement Example

---

### Honest Tea

*Honest Tea seeks to create and promote great-tasting, healthy, organic beverages. We strive to grow our business with the same honesty and integrity we use to craft our recipes, with sustainability and great taste for all.*

Source: <https://www.honesttea.com/our-mission/>

## Vision Statement Example

### Honest Tea

*Honest Tea seeks to create and promote great-tasting, healthy, What  
organic beverages. We strive to grow our business with the same  
honesty and integrity we use to craft our recipes, with sustainability  
and great taste for all. Who*

Source: <https://www.honesttea.com/our-mission/>

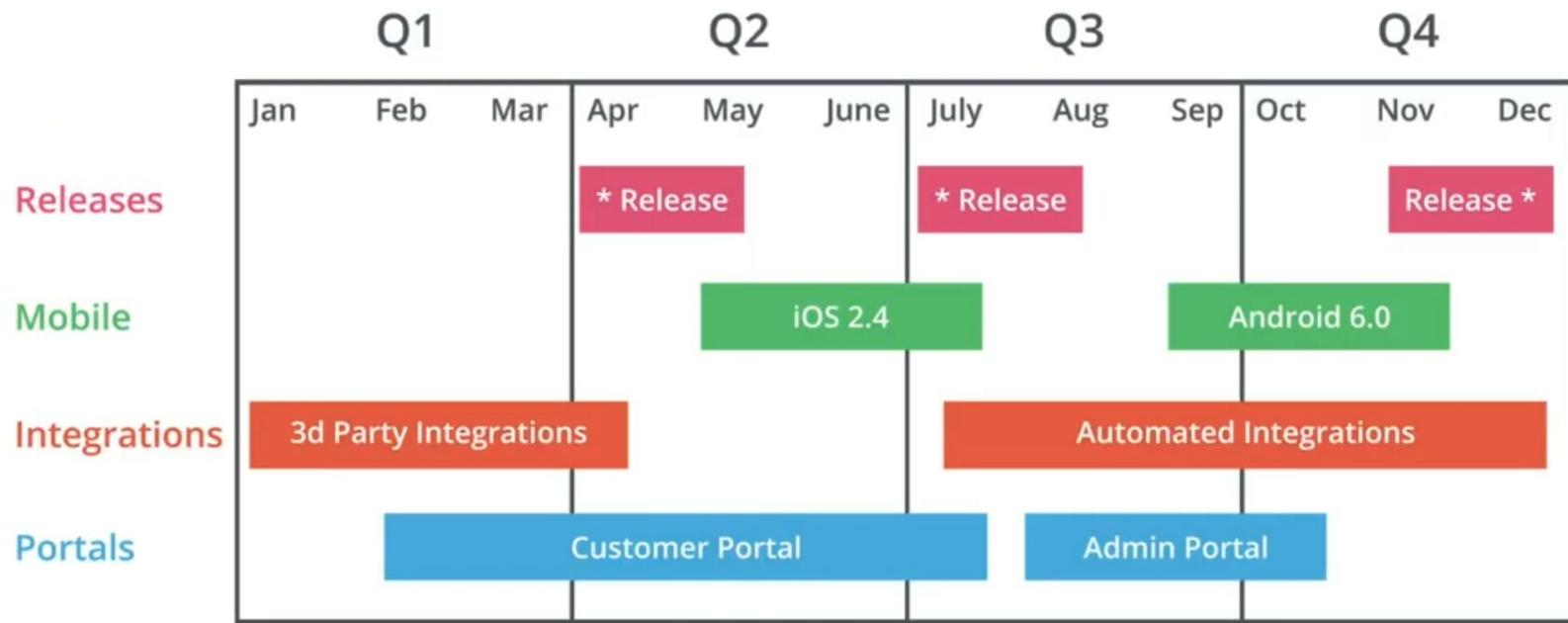
## Product Roadmap

---

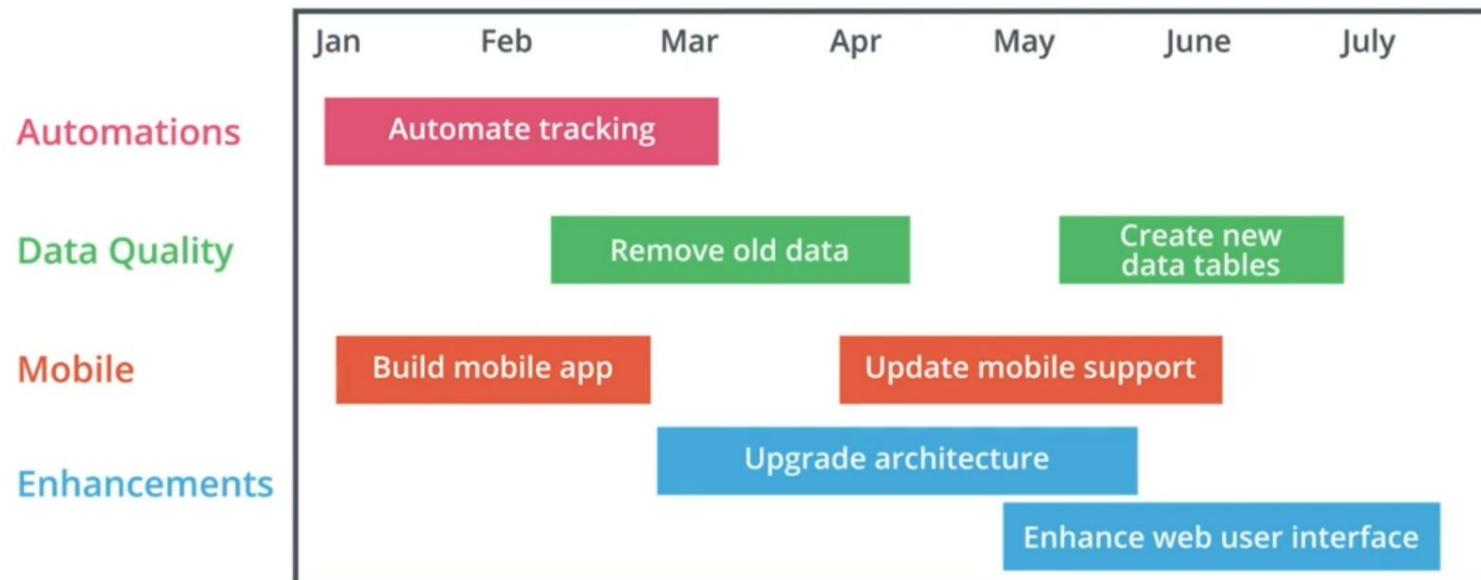
- High-level plan
- Guides the product development
- Communication tool
- Displayed in phases

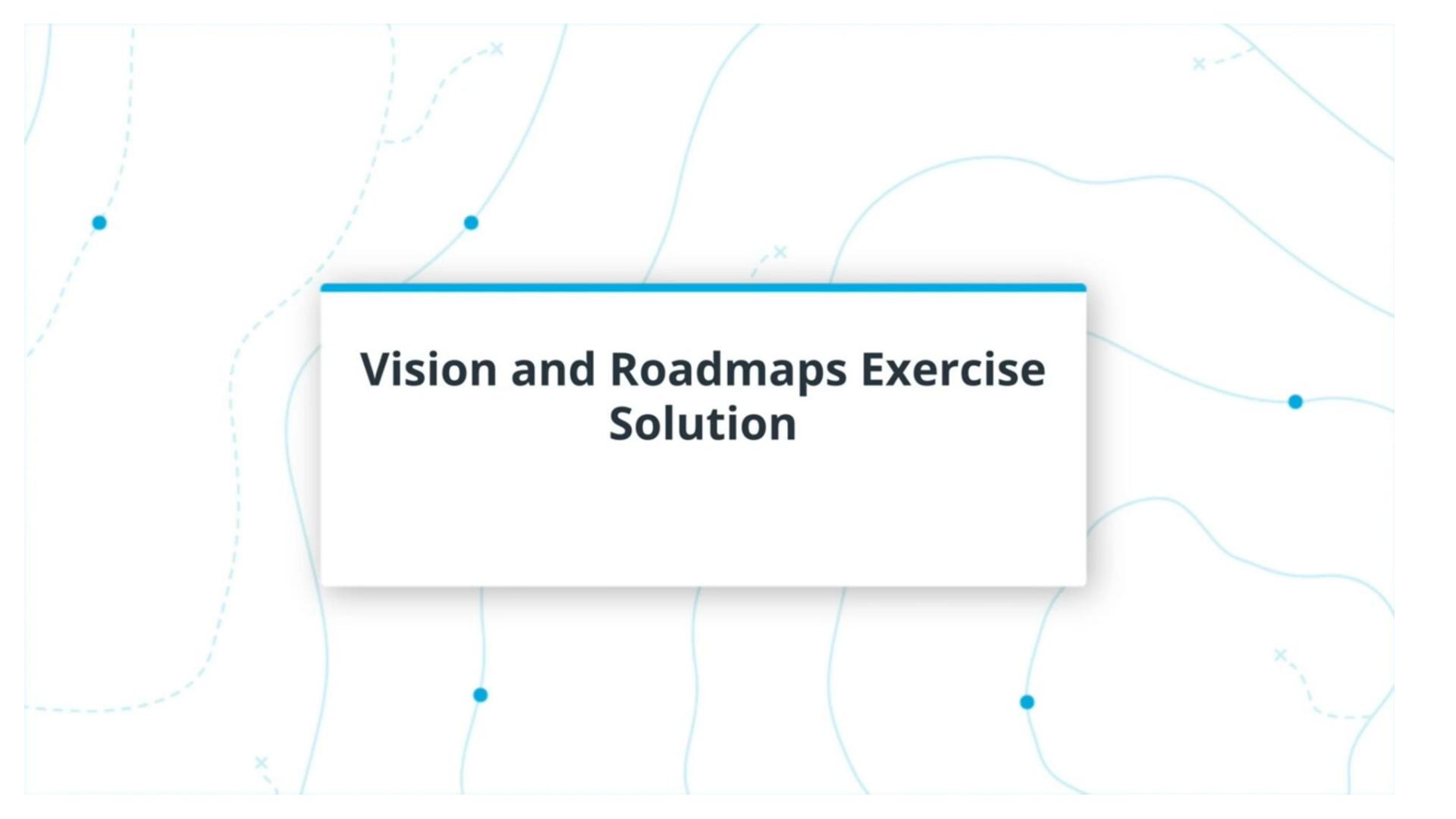


# Product Roadmap Example



# Product Roadmap Example





## **Vision and Roadmaps Exercise Solution**

## Vision and Roadmaps Exercise Part 1

---

Identify the following for each of the given visions

- Target population or market
- The intended outcome

## Vision and Roadmaps Exercise

---

Nike

Bring inspiration and innovation to every athlete\* in the world

What

Who

\*If you have a body, you are an athlete

Source: <https://about.nike.com/>

## Vision and Roadmaps Exercise

---

**LinkedIn**

*To connect the world's professionals to make them more productive  
What #1 Who and successful. What #2*

Source: <https://about.linkedin.com/>

## Vision and Roadmaps Exercise

---

**ASOS**

*To become the number 1 fashion destination for 20-somethings*  
What Who  
*globally.*

Source: <https://www.asosplc.com/>

## Vision and Roadmaps Exercise

---

**Microsoft**

*Our mission is to empower every person and every organization on  
the planet to achieve more.*

Who  
What

Source: <https://www.microsoft.com/en-us/about>

## Visions and Roadmaps Exercise Part 2

---



- You are the Product Owner in a Shoe Company
- The company wants to be a well recognized shoe brand
- The company wants to build a mobile app to sell their shoes
- Create a Vision for the mobile app product

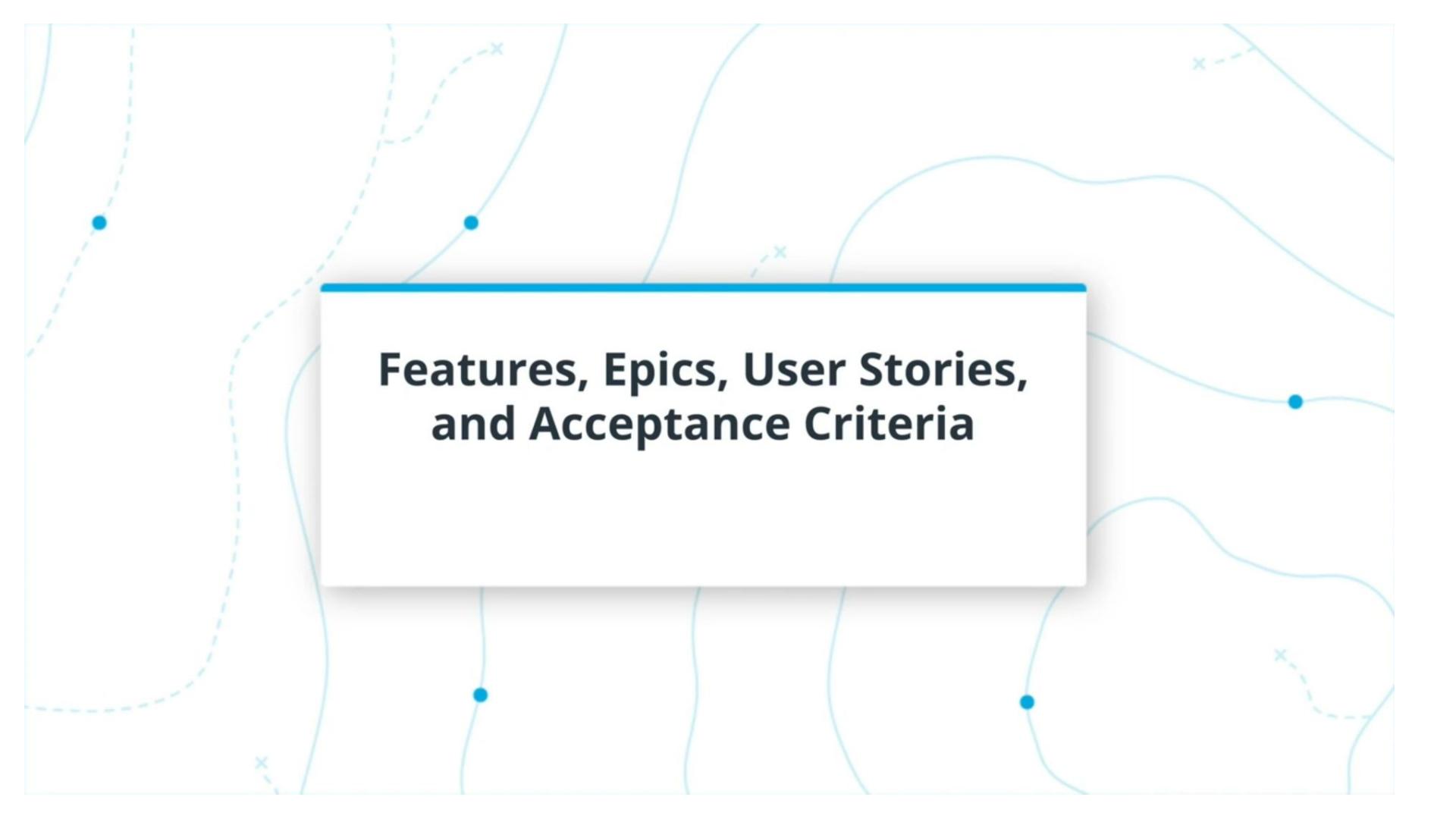
## Example Solution

---

We aspire to provide an effortless mobile experience that is  
easy for all customers to purchase our shoes.

What

Who



## Features, Epics, User Stories, and Acceptance Criteria

## What is a User Story?

---

- User stories are a tool used to help teams plan work
- User stories provide descriptions of requirements that are written based on the user of the system
- User stories clarify the benefit that users are looking to receive

## How user stories are written

---

As a **<role>**, I want to **<function>**, so that **<outcome>**.

As a **customer**, I want to **receive an email**, so that **I know which items are on sale**.

## User Stories and the 3 C's



Card



Conversation



Confirmation

## User Story

### INVEST Criteria

	I	Independent
	N	Negotiable
	V	Valuable
	E	Estimatable
	S	Small
	T	Testable

## Good Acceptance Criteria...

---

- Accompanies each user story
- Provides guidance and parameters for the function or need requested
- Does not limit the team's creativity
- Does not tell the team "HOW" to do it

## Acceptance Criteria

---

As a home buyer, I want a garage so that I don't have to clean snow off my car.

- Ability to park 2 cars
- Enough room to add storage shelves



## Acceptance Criteria

---

As an employee, I can enter my time, so that I can be paid.

- Time entered in hours and minutes
- Time can only be entered Monday - Friday
- Employee can not enter time for another employee

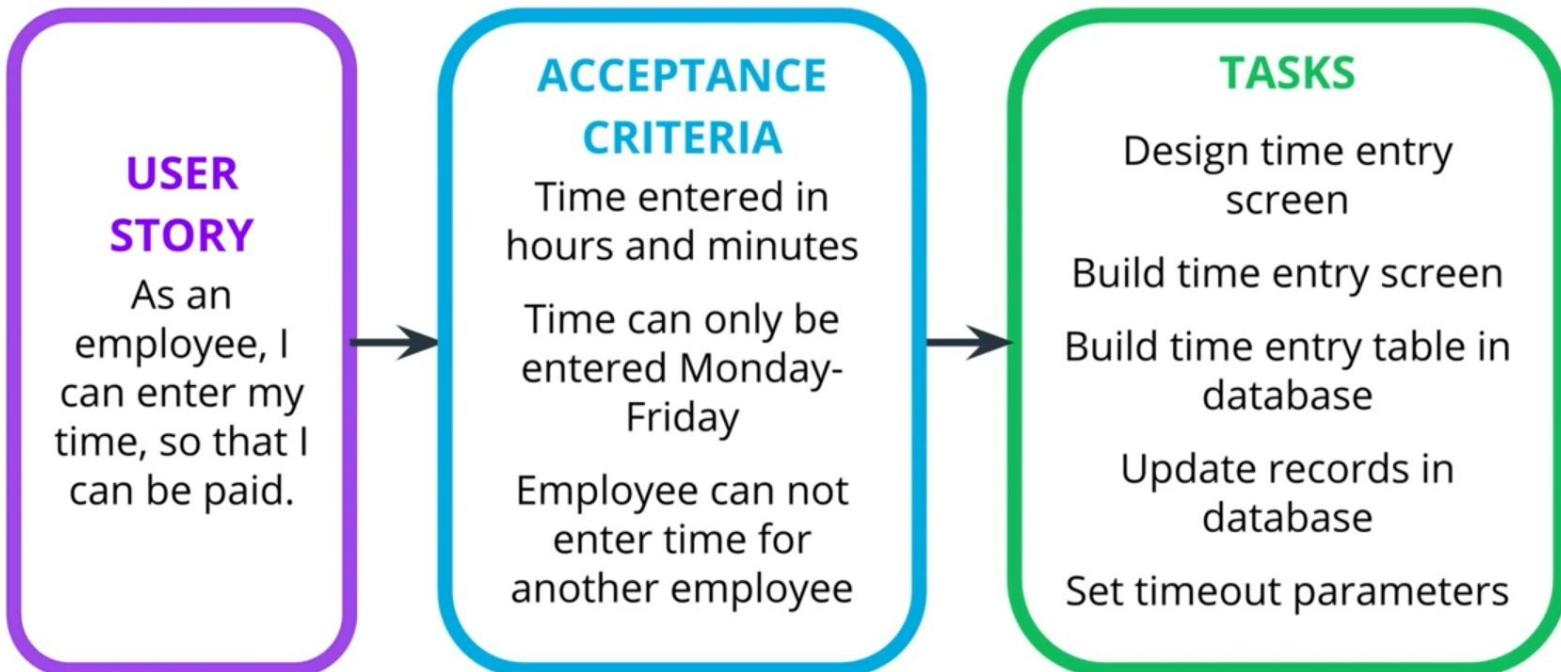
## Tasks

---

The individual steps the developer will do to complete the user story



## Tasks: Example



## What is an Epic?

---

*A large user story that needs  
to be broken down  
in order to be implemented.*



## **What is a Feature?**

---

*Functionality or sets of functionality  
that provide users of a product  
with specific capabilities.*



## Epic vs Feature vs User Story

Epic



A large user story

Feature



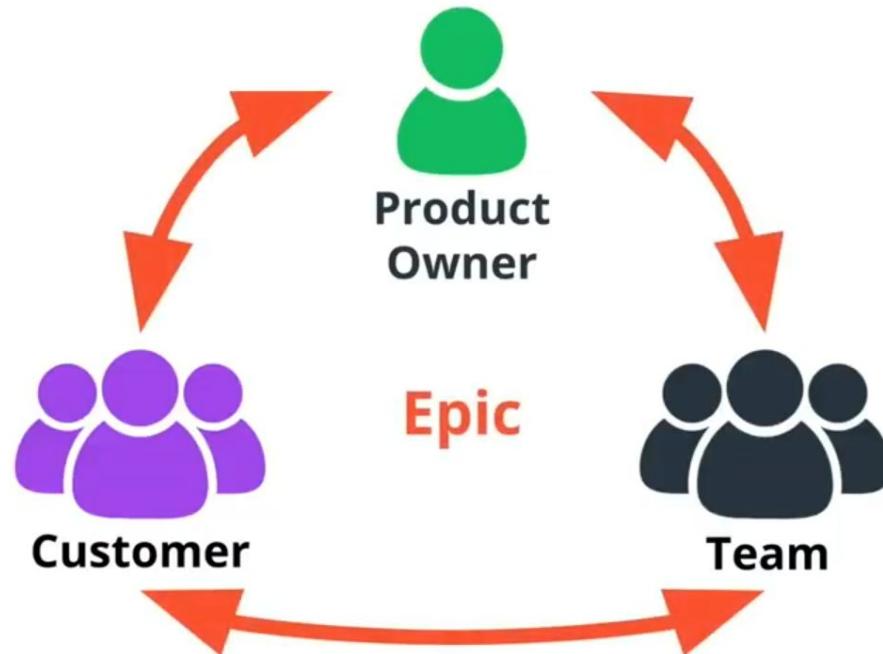
Functionality for users

User Story

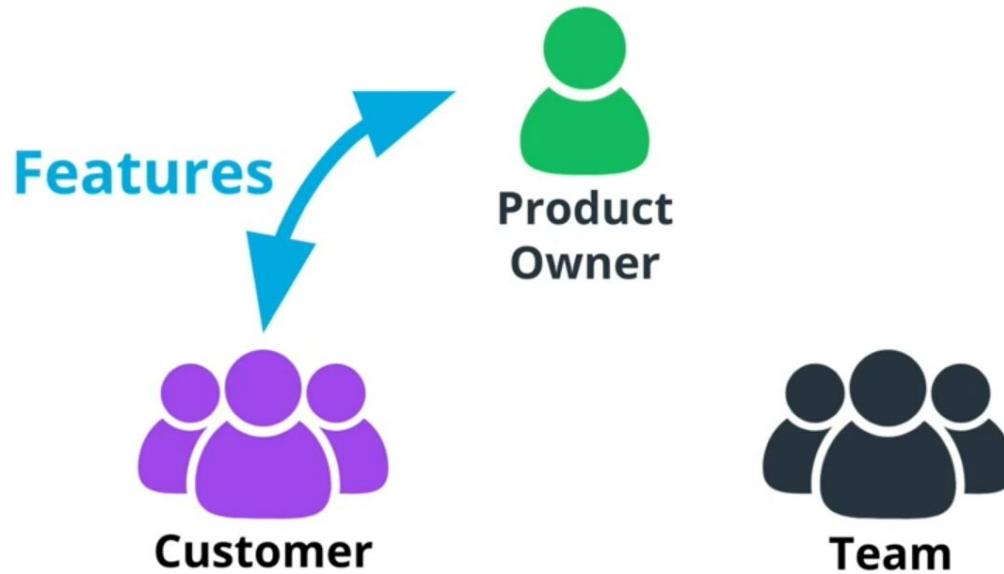


User Story

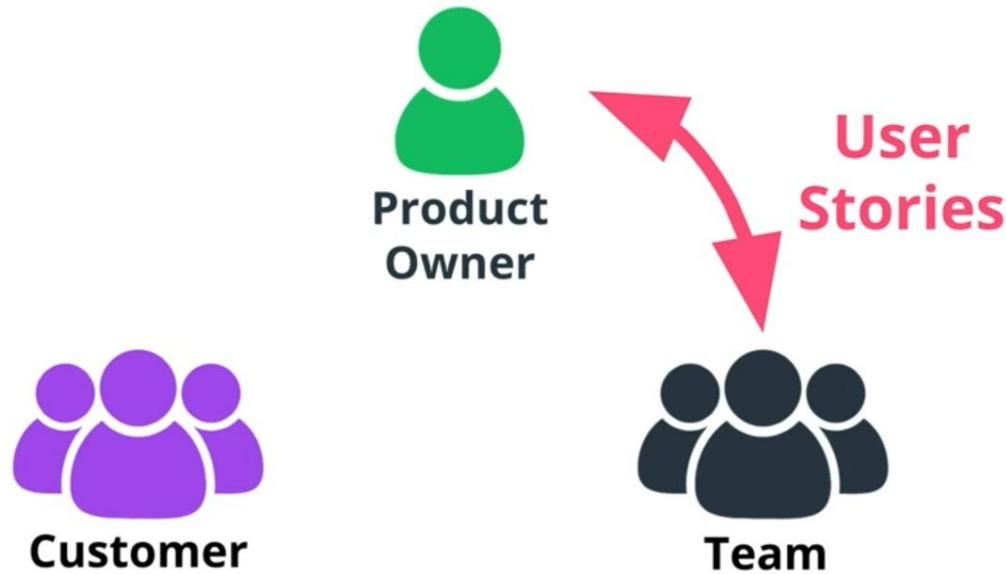
## Epic vs Feature vs User Story



## Epic vs Feature vs User Story



## Epic vs Feature vs User Story



## Epic vs Feature vs User Story

### Epic

As an electric customer, I can make online payments so that I can save time.

### Feature



Payment by Credit Card



Automated Payments



Payment by Checking Account

### User Story

# Epic vs Feature vs User Story

## Epic

As an electric customer, I can make online payments so that I can save time.

## Feature



Payment by Credit Card

## User Story

As a user, I can update my credit card number so that I can make an online payment

As a user, I can add my credit card number so that I can make an online payment

As a user, I can delete my credit card number so that I don't receive incorrect charges

# Epic vs Feature vs User Story

## Epic

As an electric customer, I can make online payments so that I can save time.

As an electric customer, I can setup my online account so that I can see my bill.

## Feature



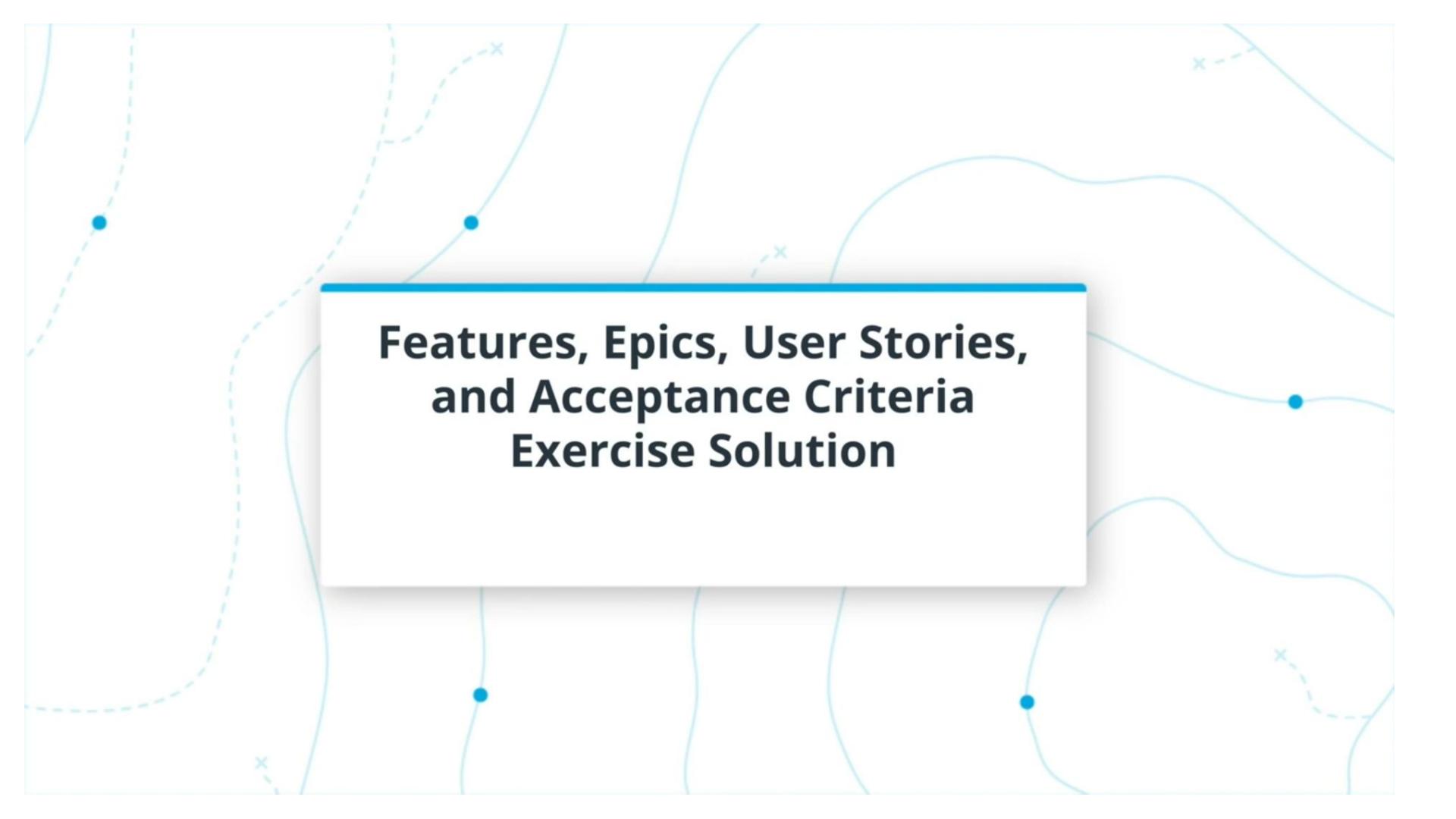
Payment by Credit Card

## User Story

As a user, I can update my credit card number so that I can make an online payment

As a user, I can add my credit card number so that I can make an online payment

As a user, I can delete my credit card number so that I don't receive incorrect charges



## **Features, Epics, User Stories, and Acceptance Criteria Exercise Solution**

## Features, Epics, User Stories, and Acceptance Criteria

---

Change the requirements into user stories



## User Stories Have Three Components

---

1. The person asking
2. The requested functionality
3. The outcome or perceived benefit

## Requirements #1

---

The administrator needs  
to be able to create a  
report to send to the  
executives.



As **an administrator**  
I can **create a report**  
so that **I can send it to the  
executives.**

## Requirements #2

A customer placed an order online and one of the items is missing from the shipment and they want to request a refund without calling the store.



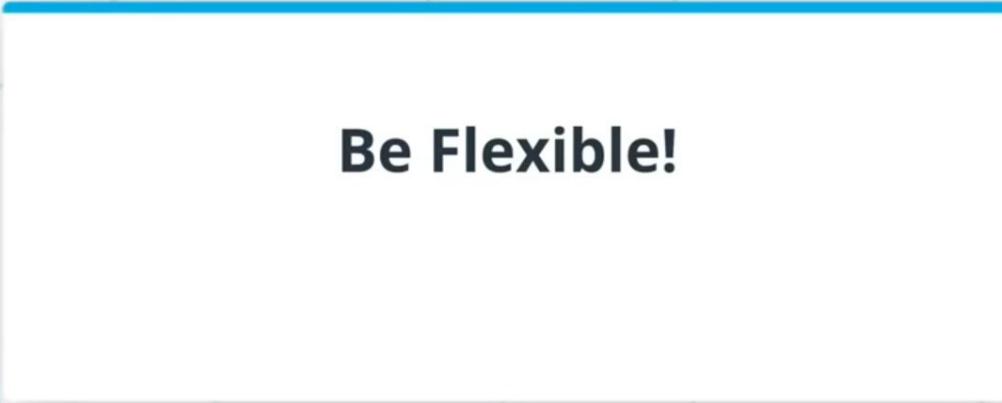
As a customer  
I can **request a refund from my mobile app**  
so that **I don't have to call the store.**

## Requirements #3

The system shall send an email automatically to all employees anytime updates are made.



As a system  
I can send an email when  
changes are made  
so that employees are  
aware of all updates.

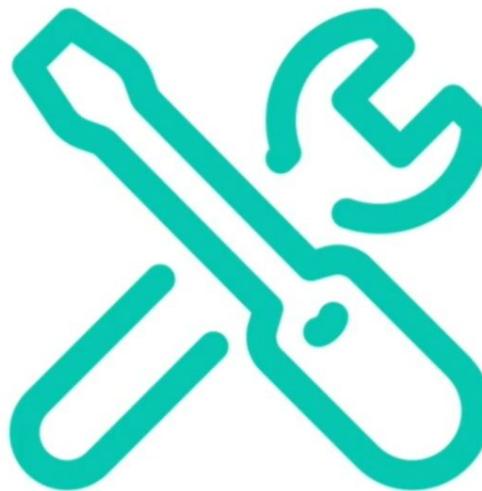


**Be Flexible!**

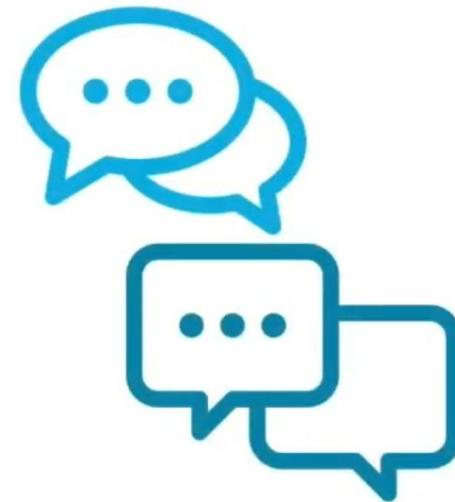
## Adapting Agile



Be Flexible



Tools



Conversations

## Personal Adaptability Example



Software



Compliance

## Personal Adaptability Example



Software



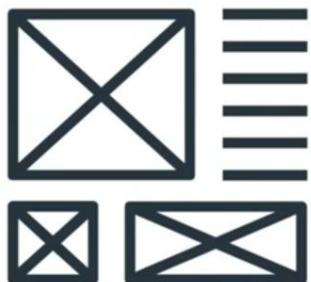
Compliance



## Product Owner Toolkit



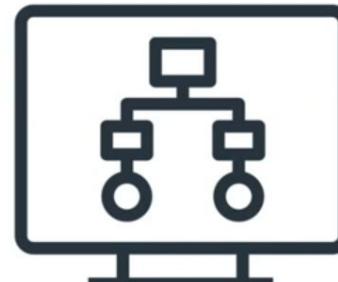
- Words are not always the best method of communication
- Product Owners use the following to communicate to teams and customers the requirements:



Wireframes



Prototype



Diagrams

## The Vision and the Mission

---

- Some companies and products have both a vision and mission statement
- Some have one or the other
- They both drive understanding and direction
- Help teams align and deliver

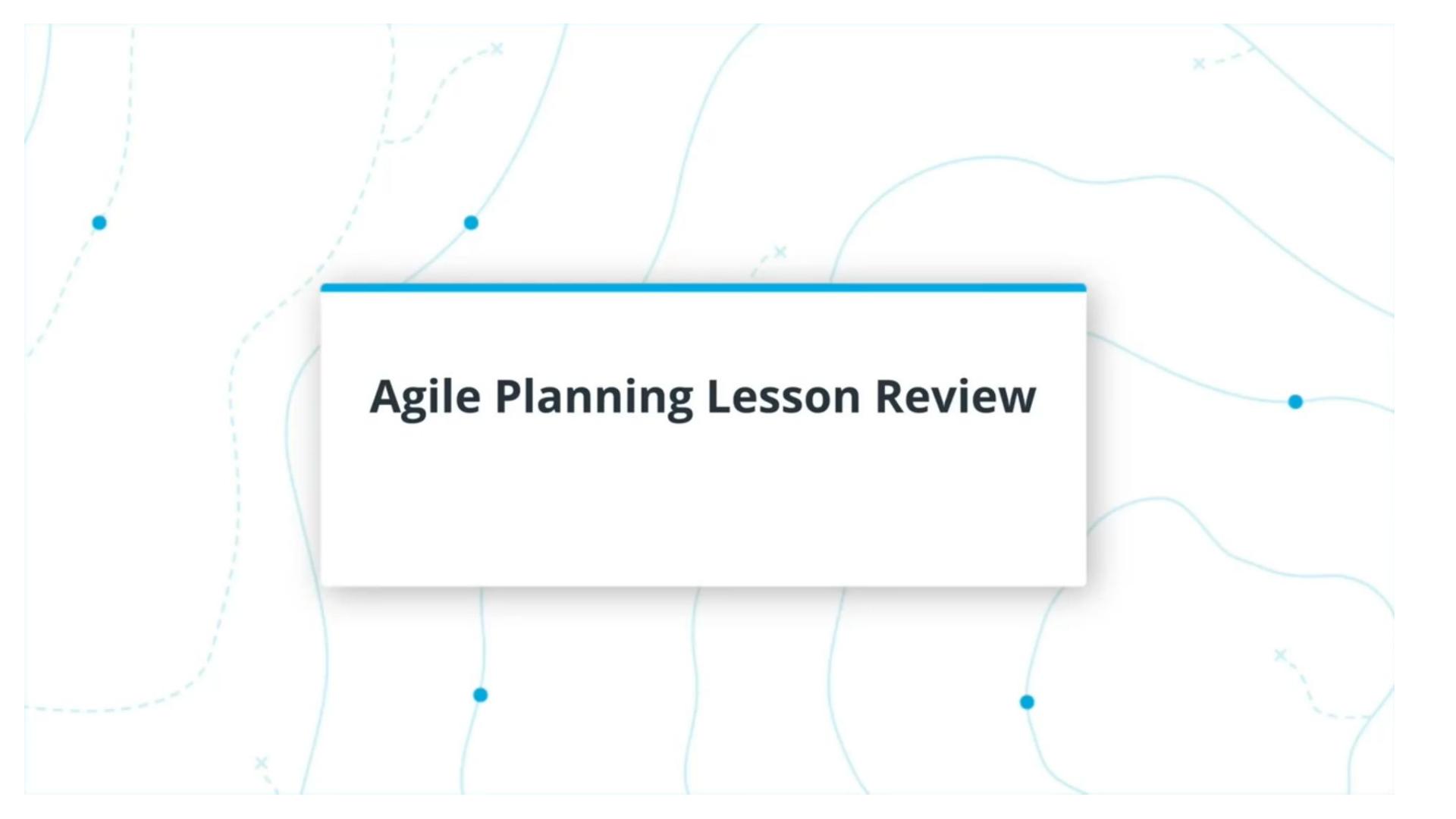
## Reality of Creating Vision Statements

Vision Statements are often a **team** effort



- Require input from the team, stakeholders, leadership and marketing
- Usually involves iteration and editing
- Should change only when necessary

***Vision statements are rarely written by just one person***



# Agile Planning Lesson Review

## Lesson Recap

---

- Intro to Planning
- Vision and Roadmaps
- Features, Epics, User Stories, and Acceptance Criteria



## Lesson Overview



- Intro to Backlogs
- Managing and Organizing the Backlog

## Learning Objectives

---

By the end of the lesson you will be able to...

- Define the purpose of the product backlog
- Explain how each role interacts with the product backlog based on the Agile framework used
- Manage a product backlog
- Apply progressive elaboration to groom a backlog

## Learning Objectives

---

By the end of the lesson you will be able to...

- Differentiate between prioritization techniques
- Implement prioritization techniques
- Prioritize a product backlog



## Why Does Prioritization Matter?

## **It's Impossible to Do It All**

---

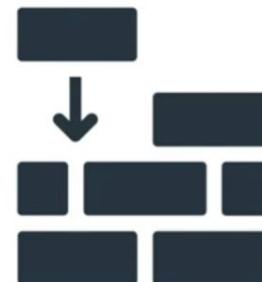


Customer requests for features, enhancements, and functionality are numerous and must be streamlined so that the team can deliver value

## **Build Quality In**

---

In order to build high-quality software, the team must focus. Without priority it is easy for the team to be distracted and quality will suffer.

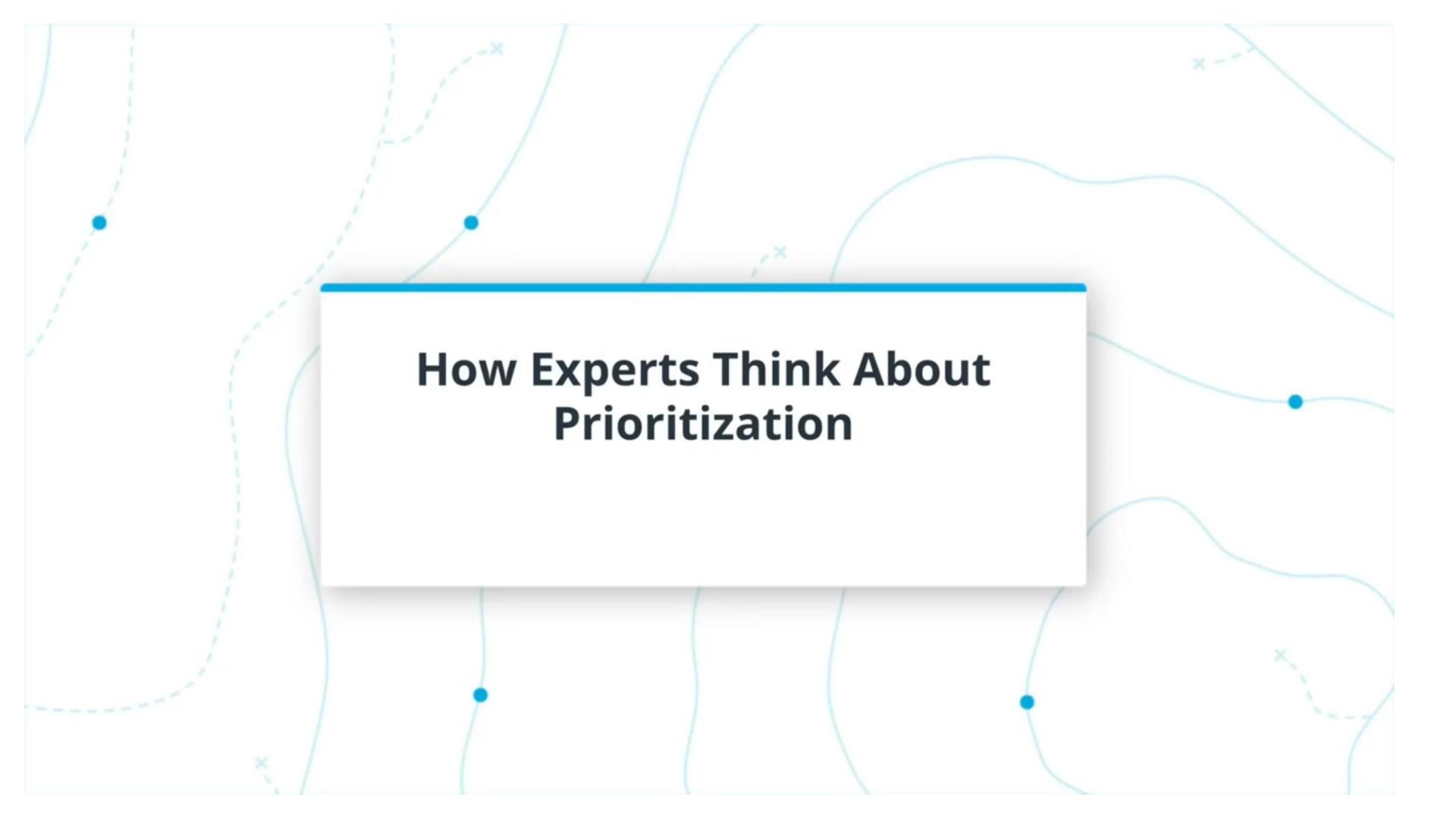


## Lack of Priority Results In...

---

- Too much Scope
- Poor quality
- Unsatisfied customers
- Unsuccessful products





## How Experts Think About Prioritization

## Two Key Questions

---

What is the business value?



How technically complex is it?



## Business Value

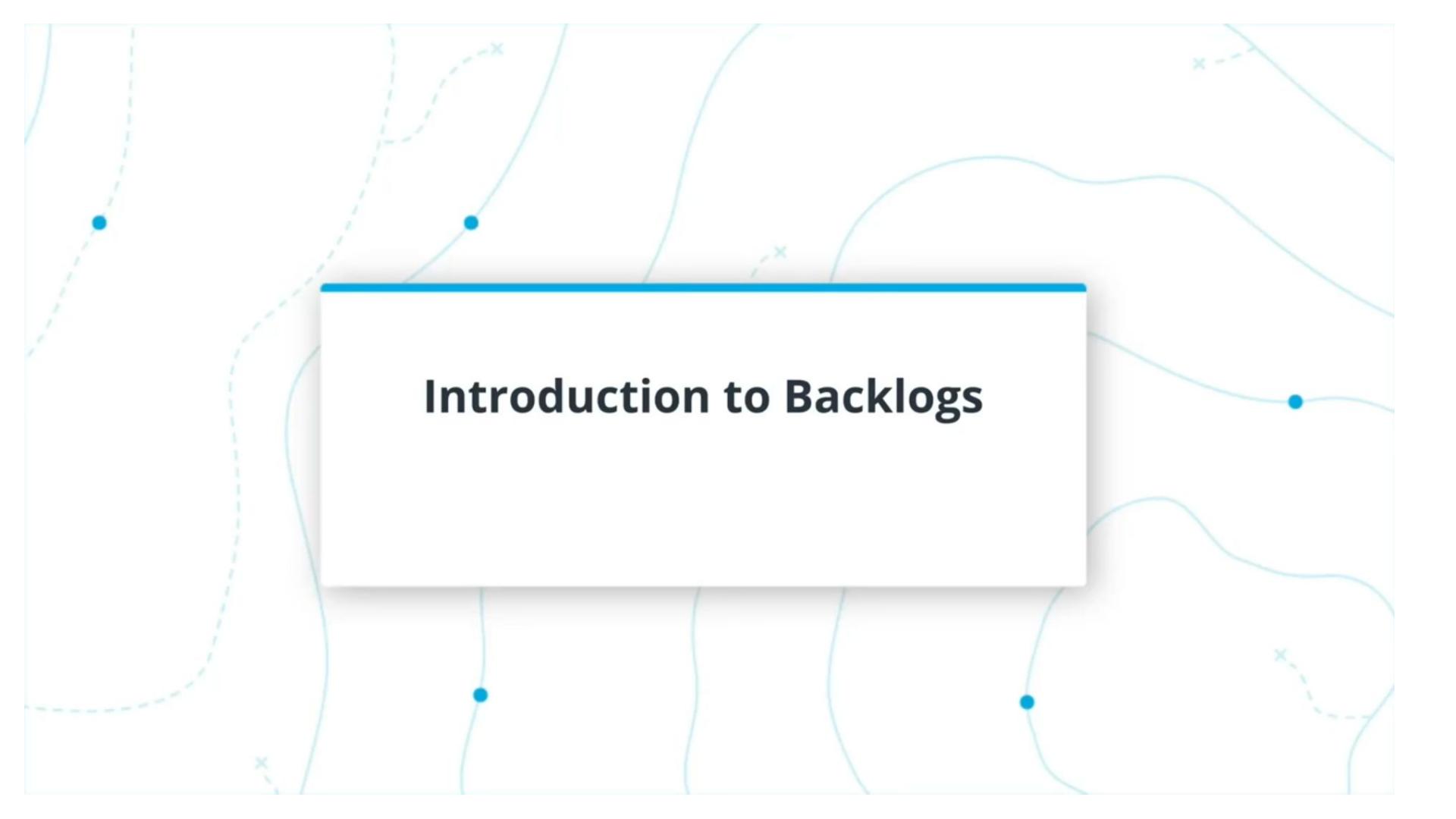
---

- What is most important for our customer?
- Are we building what will make our customer successful?
- Are we building something that our customer will benefit from?
- Are we delivering value?
- What is not critical for success?
- What is the customer asking for that they don't actually need?

## Technical Complexity

---

- What is the simplest solution to solve the problem?
- Tackle the most riskiest components first so that we can determine the right path forward.
- Do the things that could make us fail, first.



## Introduction to Backlogs

## What is the Product Backlog?

---

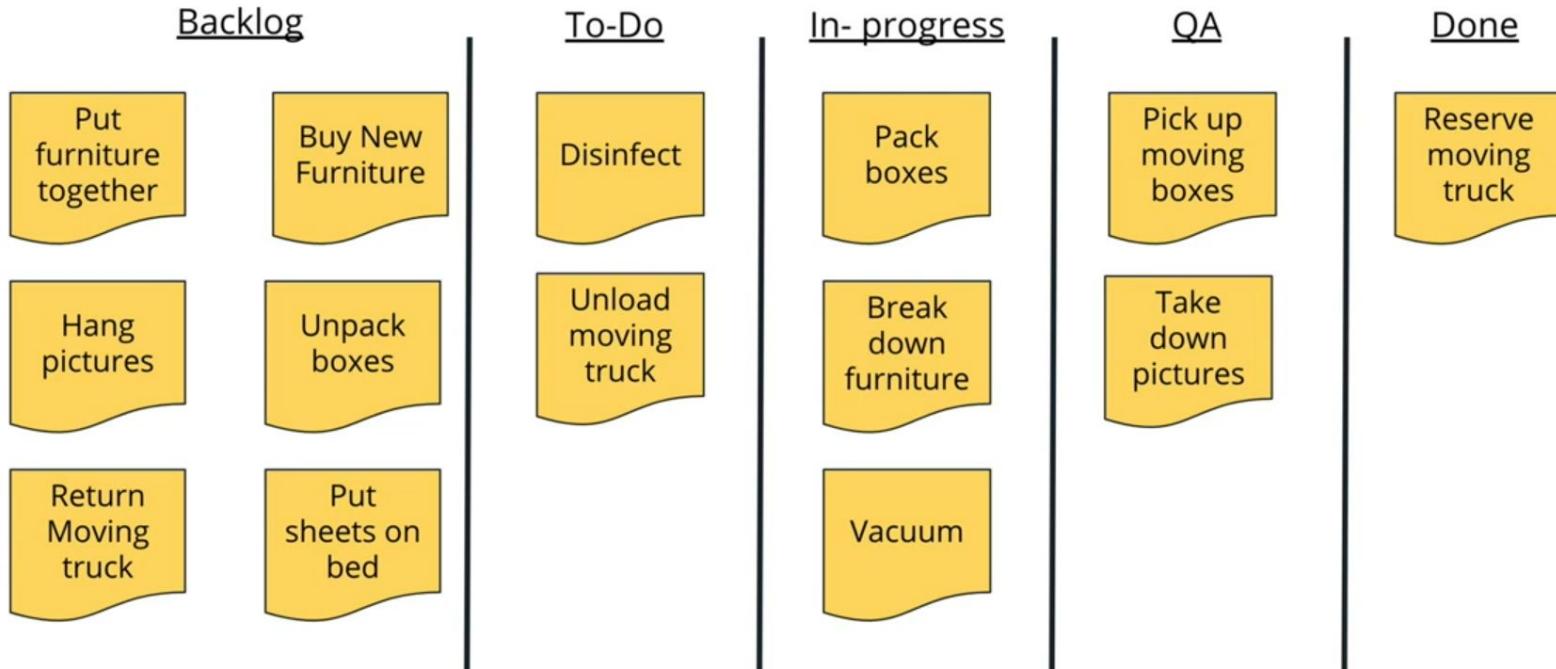
- List of all the remaining work for a product
- Planning tool for the team developing the product
- Contains user stories, features, and epics for a product
- Centralized and Prioritized “TO-DO List” for the product

## Backlog Example

### To-Do List

- ✓ Reserve moving truck
- ✓ Pick up moving boxes
- ✓ ~~Takedown pictures~~
- Pack boxes
- Break down furniture
- Vacuum
- Disinfect
- Unload moving truck
- Put furniture together
- Unpack boxes
- Put sheets on the bed
- Return Moving truck
- Hang pictures

## Example of a Backlog (Spring Cleaning)



## Backlog Interactions



### Product Owner

Manages and Prioritizes the backlog



### Development Team

Pulls prioritized work from the backlog



### Scrum Master

Helps the product owner manage the backlog

## Product Owner and the Backlog



- Owns the product backlog
- Single point of contact that prioritizes the backlog
- Adds detailed user stories that are ready to be implemented
- Adds stories, features, and epics that are to be implemented in the future
- Adds feature requests from customers to the backlog
- Cleans the product backlog by removing stories or features that won't be implemented

## The Development Team and the Backlog



- Queue of work to be completed
- Pulls in work; Single source of work
- Adds technical related stories/tasks to backlog when needed
- Estimates stories so that Product Owner can prioritize

## Scrum Master and the Backlog

---



- Assists product owner with management and prioritization
- Ensures team works on the right stories
- Protects the team from external distractions

## **Product Backlog Highlights**

---

- Centralized and Prioritized “TO-DO List” for the product
- Planning tool managed by the Product Owner
- Team’s single source of work
- Continuously changing, evolving, and never-ending



## Introduction to Backlogs Exercise Solution

## Scenario



**Mary**

Product Owner's Boss



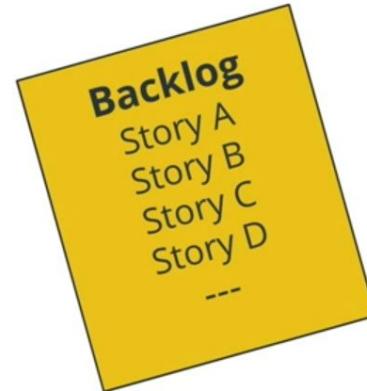
**John**

Product Owner



**Susan**

Scrum Master



## Scenario



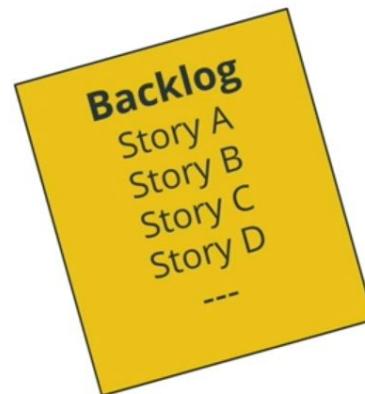
Mary  
Product Owner's Boss

New Features need to be developed!

Add them to the backlog , move them to the top and get started immediately!



John  
Product Owner



Susan  
Scrum Master

## Scenario



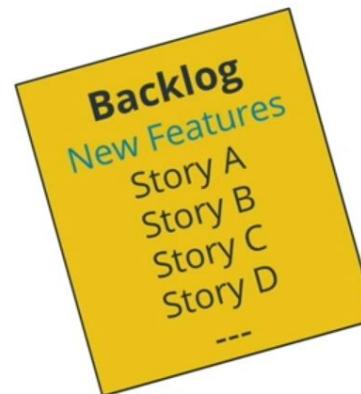
Mary  
Product Owner's Boss

New Features need to be developed!

Add them to the backlog , move them to the top and get started immediately!



John  
Product Owner



Susan  
Scrum Master

## Example Solution

How could Susan have used a more Agile approach to handling Mary's request?

- Susan should not have had the team start working on the features immediately
- Product Owner should be the only one prioritizing work
- Susan could add the stories to the backlog
- Susan could point John to the stories and tell him about the conversation she had with Mary.



## Managing and Organizing the Backlog

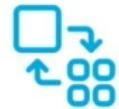
## The Backlog

---



- Continuously evolving and changing
- Product Owner must ensure that it is organized and well-managed.

## Good Product Backlog: DEEP



D Detailed Appropriately



E Estimated

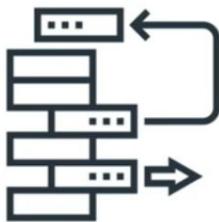


E Emergent



P Prioritized

## Is the Story “Ready”?



Product  
Owner

User Stories are  
**prioritized** and  
**well-defined**



Planning Meeting

## Adjust the Backlog For Risk

---

- Stories are added to the backlog that deliver the highest value
- Determine what stories present the most risk
- User stories with the highest value that present the highest risk are prioritized first in the backlog

## The Top of the Backlog

---

- Higher priority user stories
- Well defined user stories
- Higher risk user stories
- “Ready” to be implemented
- INVEST criteria applied to stories



## The Bottom of the Backlog

---

- Lower priority user stories
- Lower risk user stories
- User stories that are not well-defined
- INVEST criteria not fully applied to stories

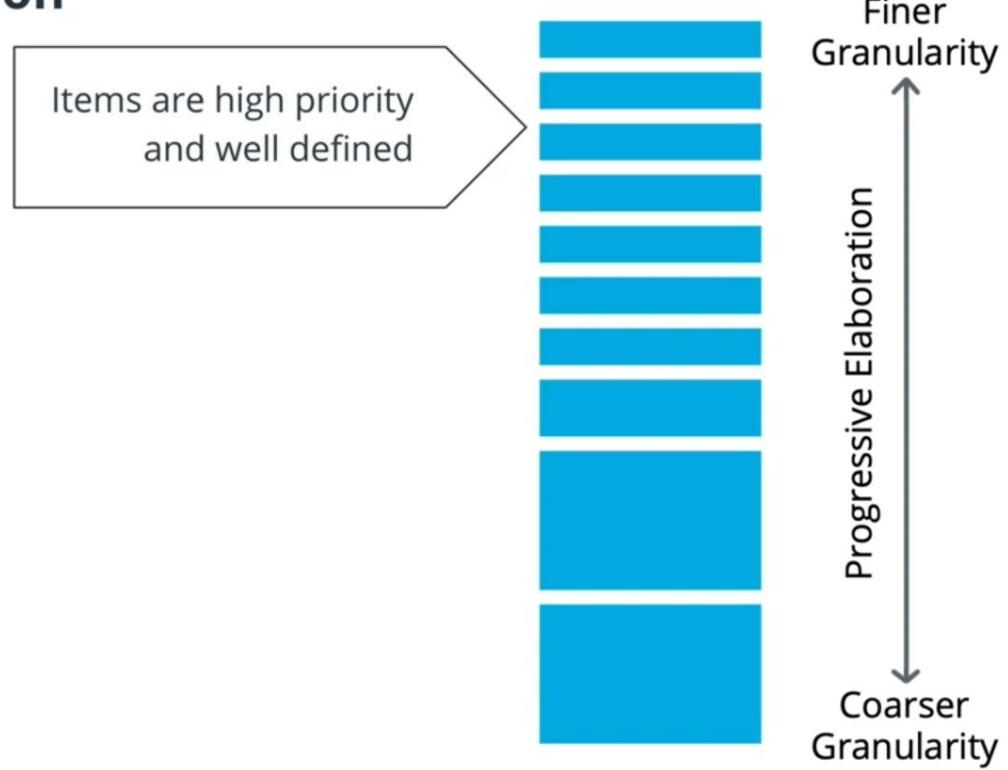


## **Progressive Elaboration**

---

The process of defining high priority user stories and arranging them at the top of the backlog and move lesser defined-lower priority stories and features to the bottom of the backlog.

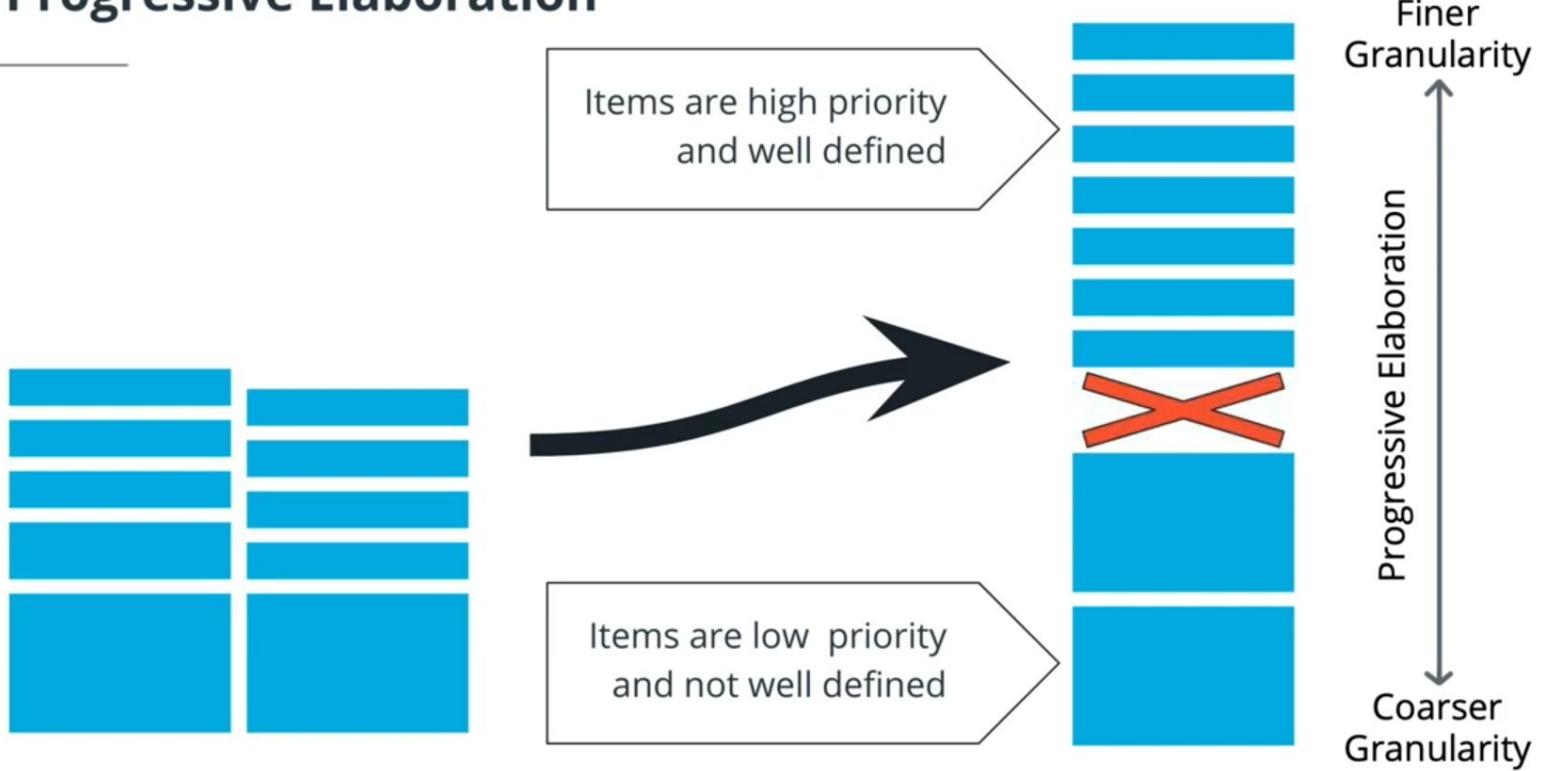
## Progressive Elaboration



## Progressive Elaboration



## Progressive Elaboration



## **Split User Stories when**

---

- Too complex or too big
- Dependencies are present
- Not able to estimate
- INVEST Criteria not met



## Scrum and the Sprint Backlog



- The **team** manages the backlog - not the Product Owner
- The amount of work does not change



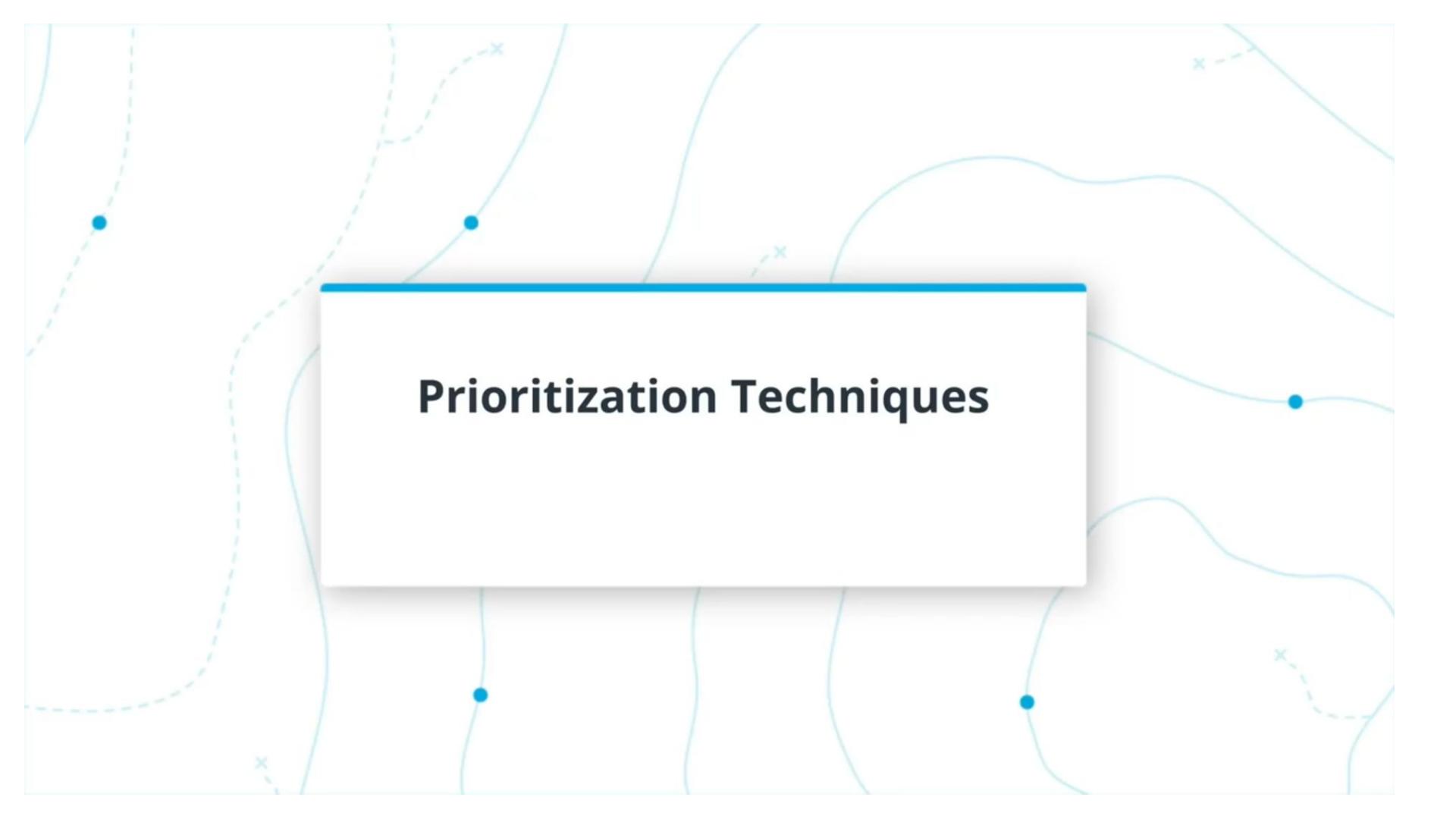
## Managing and Organizing the Backlog Exercise Solution

## Managing and Organizing the Backlog

Scenario:

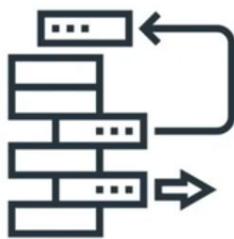
- You are the product owner for a financial management application.
- Using progressive elaboration and the INVEST criteria, assign the user stories you have been given to the “Top of the backlog” or the “Bottom of the backlog” based on the information provided about each user story.

US#	Information About User Stories (US)	Backlog Ordering
1	Depends on story 7	Bottom of the backlog
2	Team has never done this type of work before	Top of the backlog
3	Story is detailed and ready to implement	Top of the backlog
4	Story can not be tested	Bottom of the backlog
5	Story is small and testable	Top of the backlog
6	Story is too big to implement	Bottom of the backlog
7	Story has no dependencies and the team has a good understanding of the story. Story is sized appropriately	Top of the backlog
8	The team has a lot of uncertainty with completing this user story	Top of the backlog
9	This is a critical user story that is well understood and appropriately sized but has a dependency on User story 3	Bottom of the backlog
10	This is a critical user story that is well understood and appropriately sized and has no dependencies	Top of the backlog



## Prioritization Techniques

## The Responsibility of Prioritizing



Product Owner

- Responsibility of the **Product Owner**
- Can be a difficult job

## The Responsibility of Prioritizing

---

- Not an exact science
- Reduce scope instead of changing timelines or resources
- Value may get cut or pushed into the next release
- Just because you think it's valuable doesn't mean it is

## A Few Considerations

---

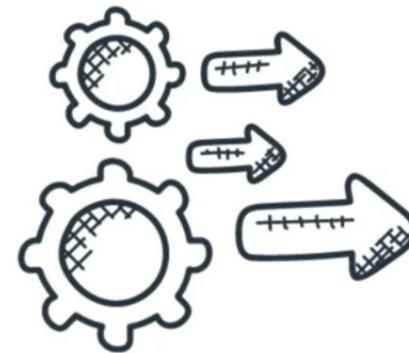
- Cost
- Time
- Technical complexity
- Customer Satisfaction



## A Few Techniques

---

- Basic Prioritization
- MoSCoW
- Dot Voting
- Play Money



## Prioritization Technique: Basic Prioritization

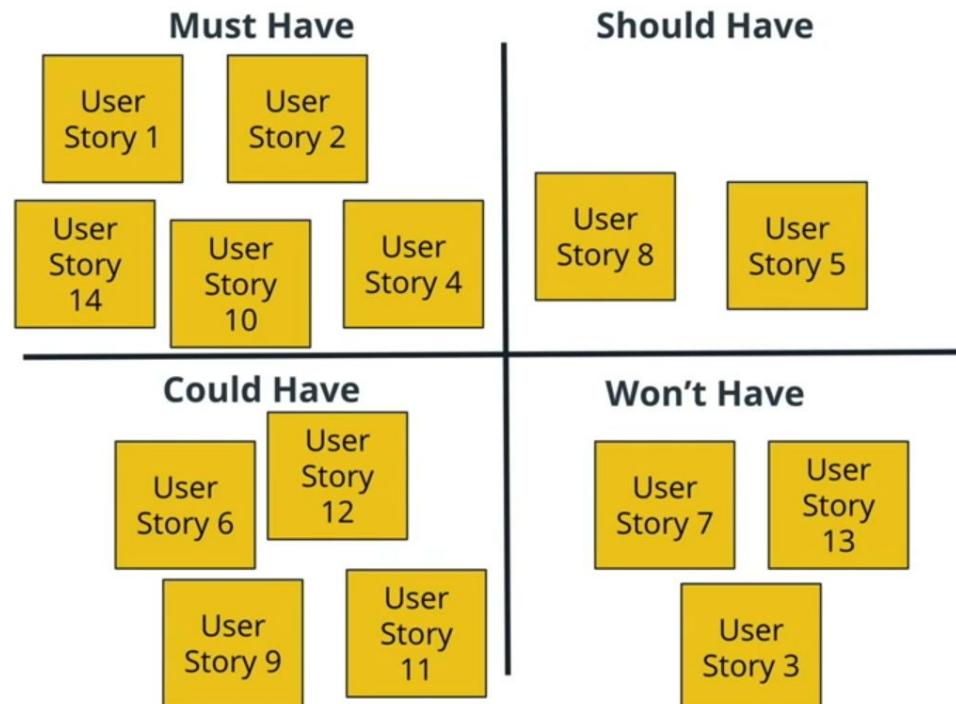
---

- Prioritize user stories based on value alone.
- User stories are assigned Priority levels such as P1, P2, or P3
- High priority user stories are assigned P1 and are implemented first
- P2 user stories are implemented next and P3 stories are last

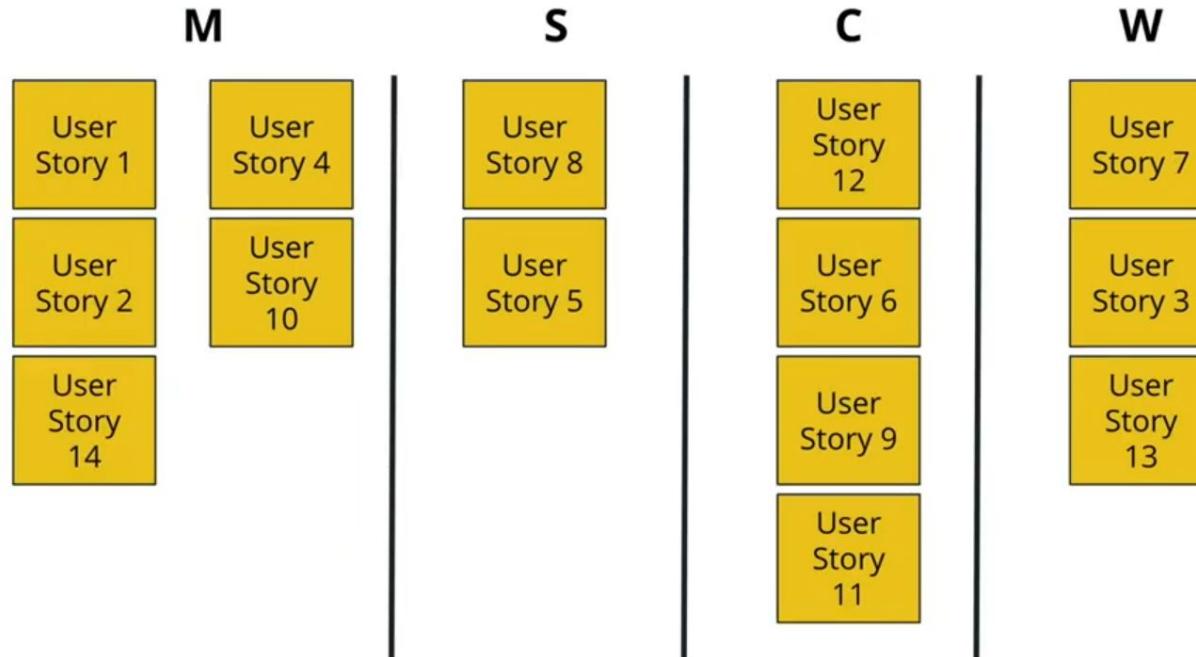
## Prioritization Technique: MoSCoW Method

-  M Must have features
-  S Should have features
-  C Could have features
-  W Won't have features

## Prioritization Technique: MoSCoW



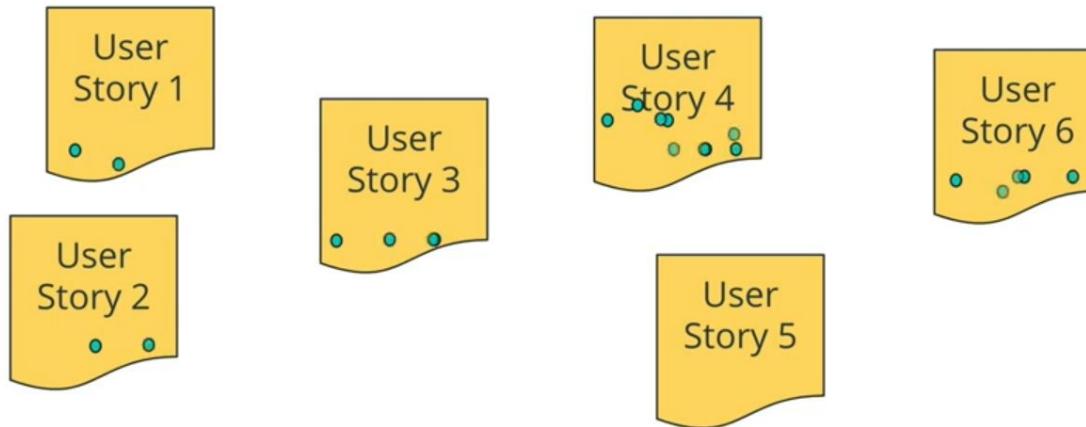
## Prioritization Technique: MoSCoW



## Prioritization Technique: Dot Voting

- User stories or features are posted on a board
- Dots represent “votes” and are given to each person
- Each person receives the same number of dots
- The number of dots that are given should be less than the number of user stories being voted on.
- Each person assigns the dots (votes) to the user stories
- User stories are prioritized according to the number of dots received

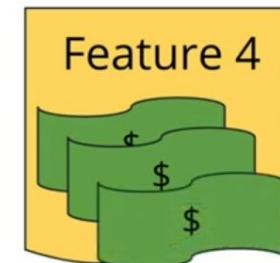
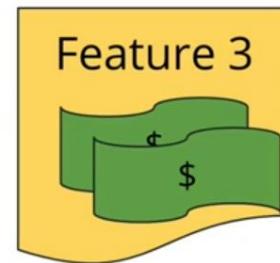
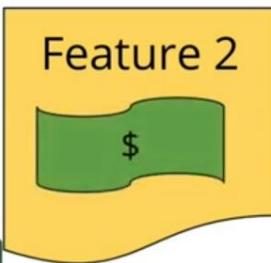
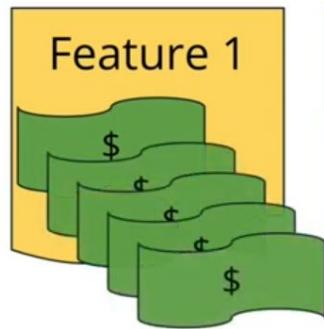
## Prioritization Technique: Dot Voting



## Prioritization Technique: Play Money

- Give each customer/stakeholder/team member/ play money that equals the product or project budget.
- Each person distributes (spends) their money on the features for the product
- Features are prioritized according to the amount of the budget they receive

## Prioritization Technique: Play Money



## Prioritization Overview

---

- Prioritization is not a trivial responsibility
- Product Owners must consider and weigh different factors
- The prioritization techniques are great way to help Product Owners choose the right scope to build



## Prioritization Techniques Exercise Solution

## Prioritization Techniques

Scenario:

You are buying a new car for your family of 7. Prioritize the list of features for your new car using the MoSCoW prioritization method.

Copy the list into the text box and place the corresponding letter (M,S,C,W) from the MoSCoW method beside the item.



## Must Haves

Must Have Car Features	Priority
Tire Installed	M
Seatbelts Included	M
Able to Seat 7 People	M
Steering Wheel Included	M
Brakes	M

## Should Haves

Should Have Car Features	Priority
Radio	S
Navigation	S

## Could Haves

Could Have Car Features	Priority
Black Exterior	C
Sunroof	C
Chrome Rims	C
V6 Engine	C
Panoramic Roof	C

## Won't Haves

Won't Have Car Features	Priority
Coffee Maker	W
Refrigerator	W
V8 Engine	W
Radio with Bluetooth	W
Trailer Hitch	W



## **Risk and Prioritization Influence**

## How Risk Informs Prioritization



## Prioritizing to Fail Fast

- Risks have the potential to make the product or project fail



## Prioritizing to Fail Fast

- Risks have the potential to make the product or project fail
- Agile teams focus on things that present the most risk, first
- The team confronts the things first that could make them fail
- Failure is overcome faster
- The team attains the knowledge needed for the product to succeed



## Examples of Risks

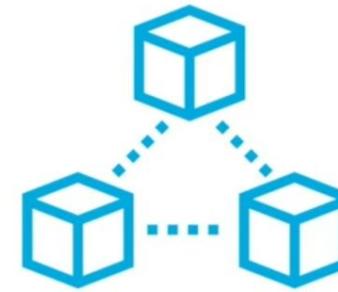
### Super Complex

Moving data from a data center to the cloud needs to be refined and cleaned to move to another location

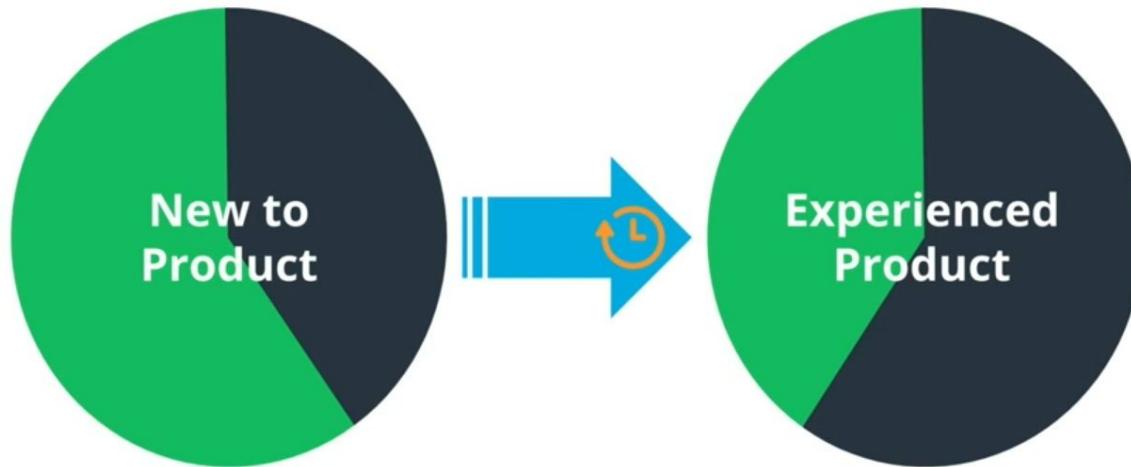


### No Prior Experience

Team was assigned to create a network of blockchains with no prior experience.



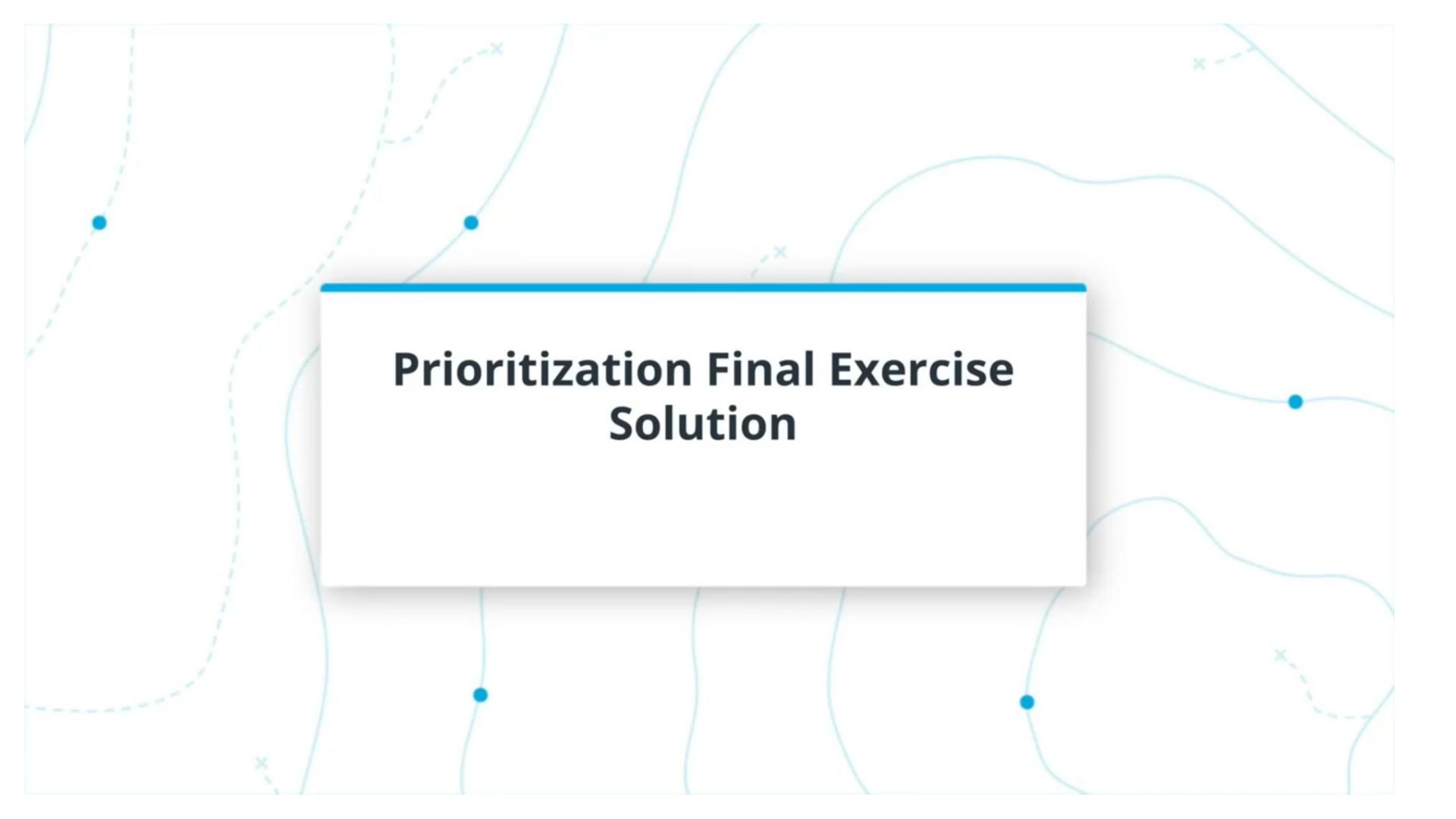
## Prioritization Influence Evolution



**Product  
Owner**



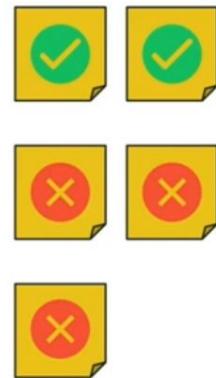
**Development  
Team**



## Prioritization Final Exercise Solution

## Prioritization Final Exercise

Scenario: Team is unable to finish the work



### CHALLENGES

- Waiting for other systems and external teams
- Missing critical components of user stories
- New stories lack detail

## Prioritization Final Exercise



Product Owner

What agile practices or techniques should the product owner use to help the team deliver what they committed to?



Scrum Master

What can the scrum master do to help the team and the product owner?

## Example Answer 1

---

*The Product Owner should use the INVEST criteria when creating user stories and prioritizing the backlog to ensure that the user stories are as independent on other systems and teams as possible. Deprioritize dependent stories until the dependencies have been completed.*

## Example Answer 2

---

*The Product Owner should write clearer acceptance criteria and ensure that the development team understands what the intended outcome of the user stories are.*

### **Example Answer 3**

---

*The Product Owner use progressive elaboration to manage and organize the backlog so that less detailed user stories are at the bottom of the backlog and more detailed user stories are at the top of the backlog.*

## **Example Answer 4**

---

*The Scrum Master should coach the product owner to manage dependencies by prioritizing independent user stories at the top of the backlog.*

## **Example Answer 5**

---

*The Scrum Master should ensure that the team has access to the product owner to ask clarifying questions about the prioritized user stories and acceptance criteria.*

## **Example Answer 6**

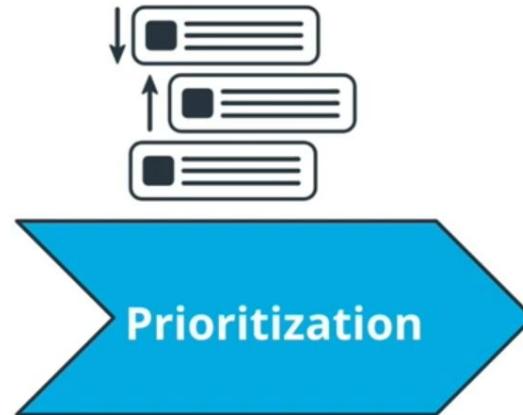
---

*The Scrum Master should coach the product owner to constantly refine and prioritize the backlog so the most detailed user stories are the top of the backlog.*

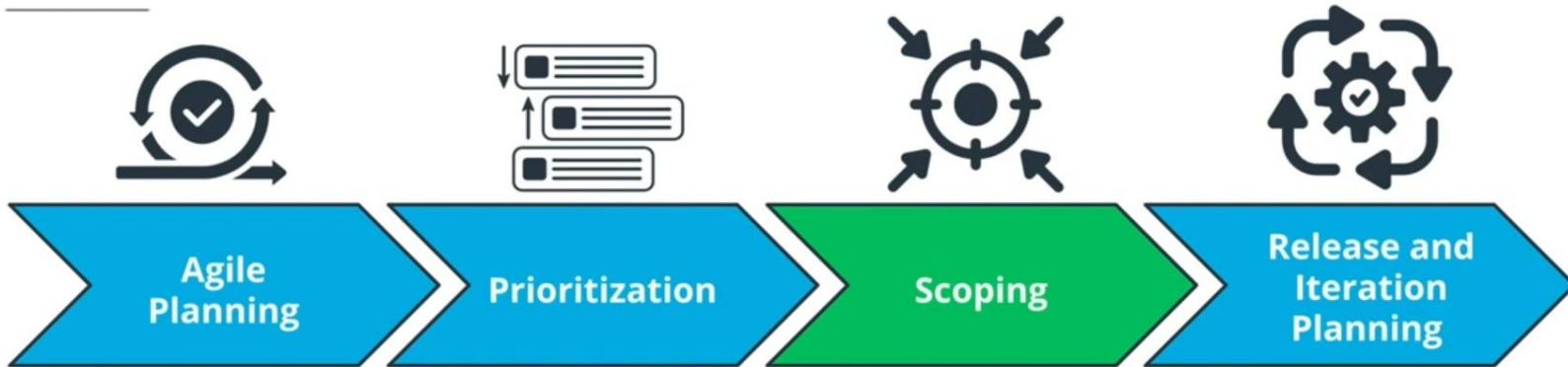
## Lesson Recap

---

- Intro to Backlogs
- Managing and Organizing the Backlog
- Prioritization Techniques



## Lesson Overview



- Intro to Scoping and Estimation
- Estimation Techniques
- Defining "Done"

## Learning Objectives

---

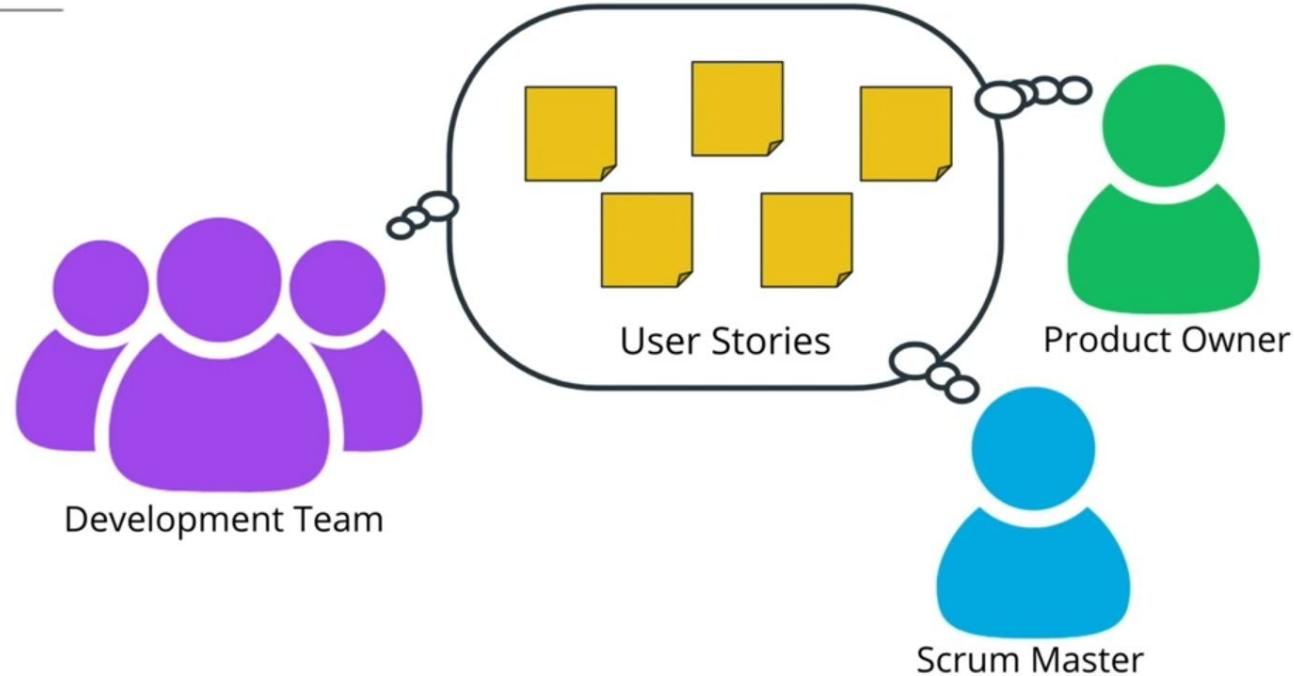
By the end of the lesson you will be able to...

- Manage the scope of a specific user story.
- Revise the definition of done for a specific user story.
- Estimate user stories relative to size, scope, and level of effort.
- Differentiate between various estimation techniques.
- Apply various estimation techniques.
- Explain ideal time



## Why Does Scoping Matter?

## Why Does Scoping Matter?



## Scoping is Important...

---

- Product Owner can prioritize better
- Provides understanding for teams
- Team can estimate
- Stories can be “split” where needed

## Approach to Scoping and Estimating



Fast



Easy



Expose Unknowns

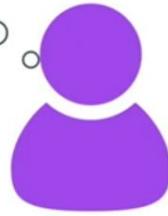


Relative

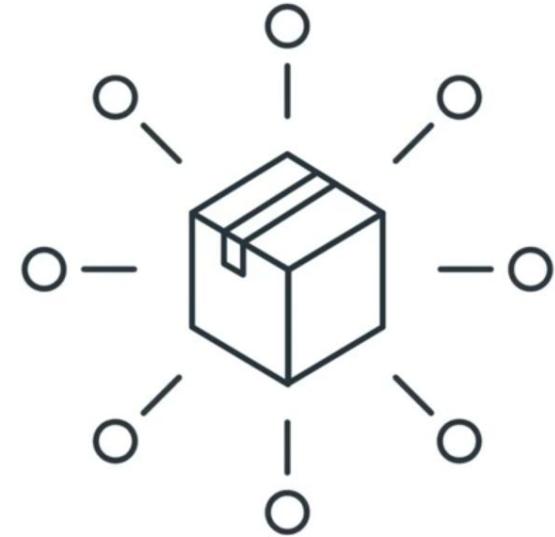
## Story



Scrum  
Master



Stakeholder



## Story



Scrum  
Master



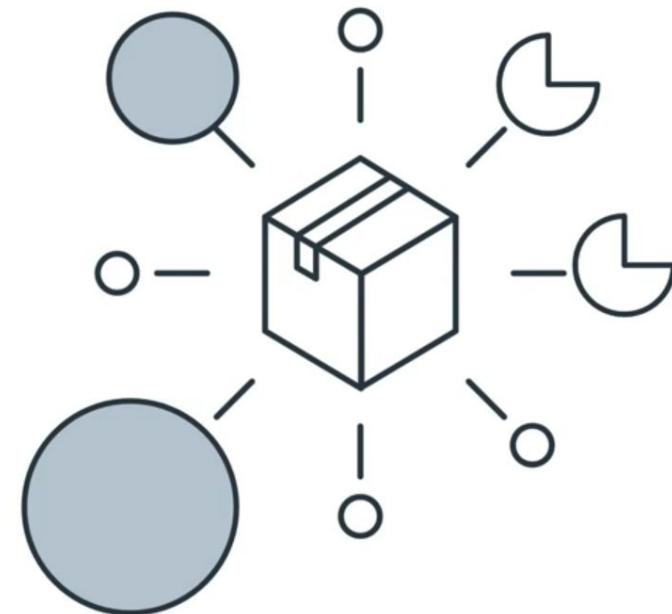
Product  
Owner



Stakeholder



Team



## Story



Scrum  
Master



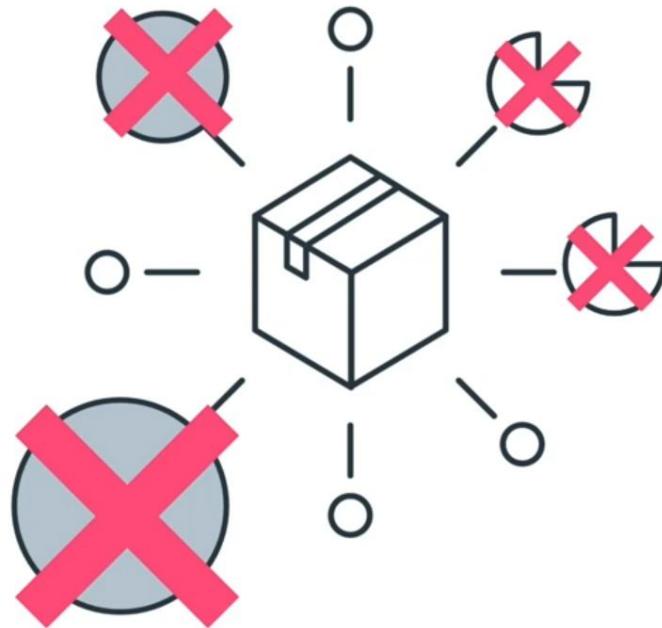
Product  
Owner



Stakeholder



Team



## Story



Scrum  
Master



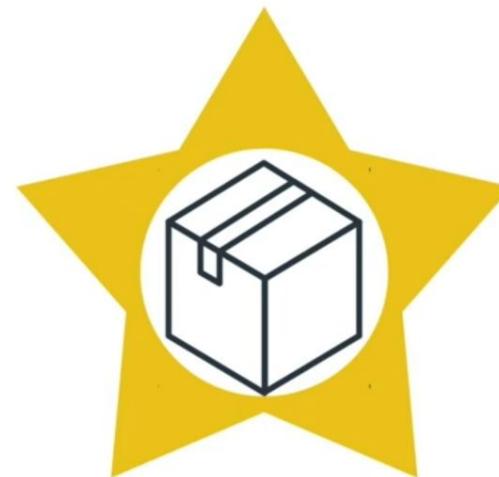
Stakeholder

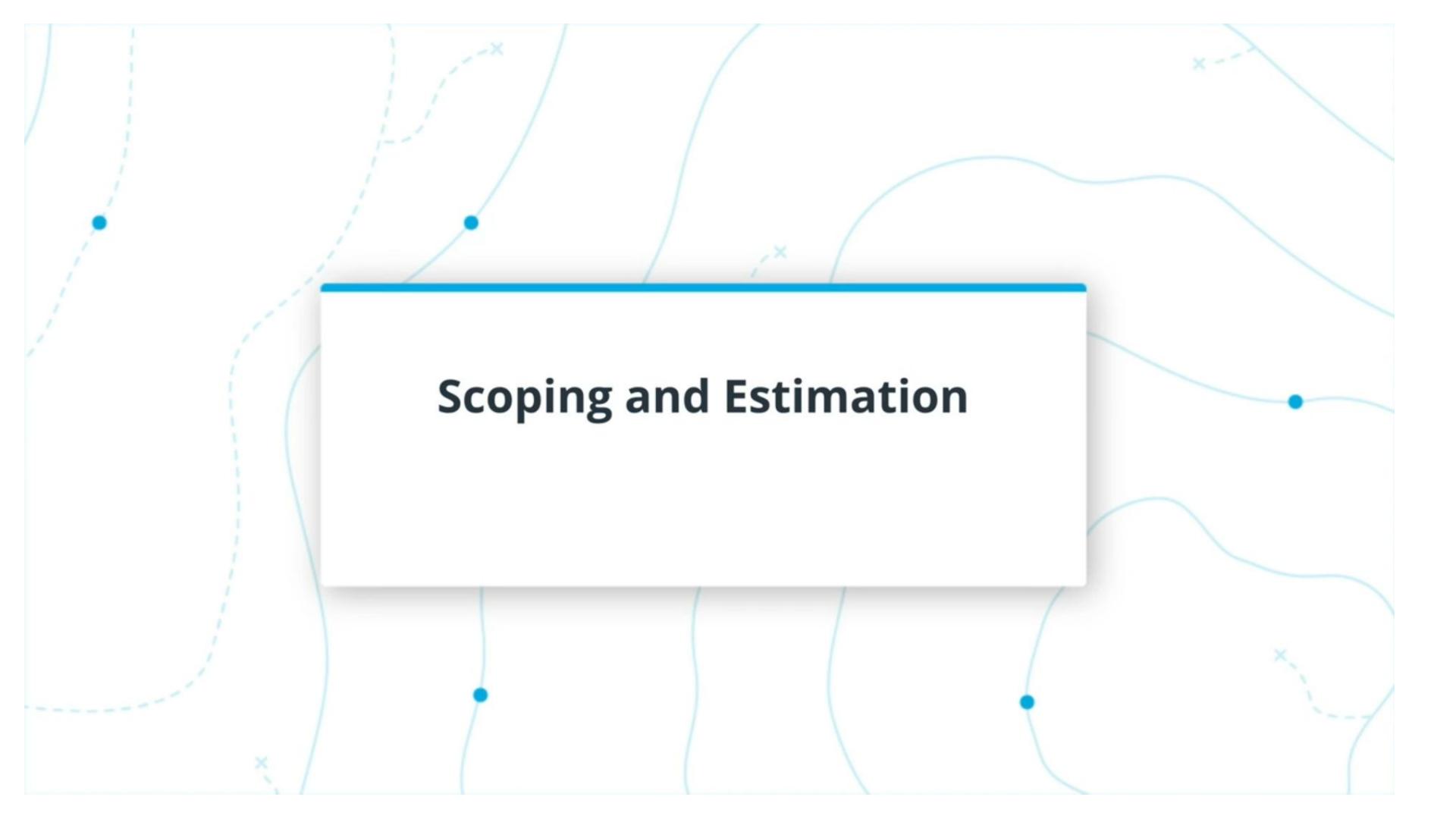


Product  
Owner



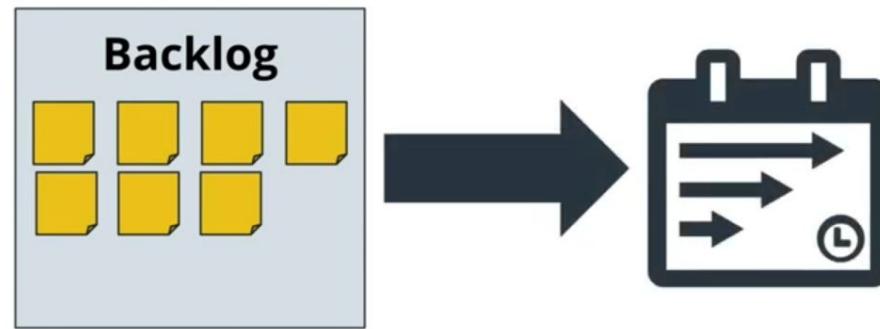
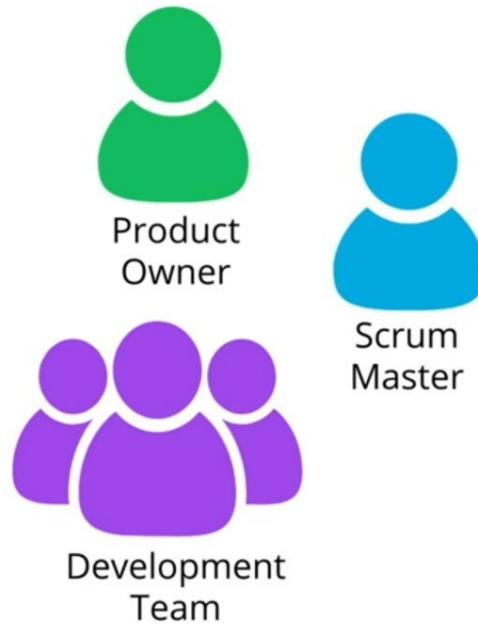
Team





## Scoping and Estimation

## Better Planning



## **How Scoping Works**

---

1. The Product Owner adds and prioritizes user stories in the backlog.
2. The Product Owner and team discuss the work together during a planning meeting.
3. The team provides a relative estimate for each user story.

## Relative Estimates in Agile Planning

---

- The process of applying estimates by considering size, complexity, level of effort, and unknowns
- Software can not be estimated in absolutes or exact measurements
- Most often teams use Story Points to represent relative estimates

## Story Points

---

- Tool used to provide relative estimates that represent scope
- Drive consensus and conversation to ensure understanding
- Numbers are relative and not equal to days, hours, mins, etc
- Have different meanings from team to team
- Used to determine the velocity of a team

## Components of User Story Estimates



Complexity



Size



Level of Effort



Unknowns

## Complexity

---

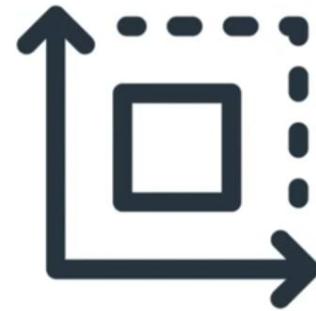


- How hard is it?
- How many steps do we have to take to complete?

## Size

---

- How big is it?
- How much of it is there?
- Ex. 100 or 1000 Rows of data



## Level Of Effort (LOE)

---



- How long will it take us?
- How many of us can work on it at the same time?
- How long did it take us to do something similar?

## Unknowns

---

- Have we done this before?
- How certain are we that we can do it?
- What do we know?
- What don't we know?

???

## User Story Estimates Take Into Account



Complexity



Size



Level of Effort



Unknowns

## Relative Estimates



Bricks



House



Fire Station



Hotel



City

## Relative Estimates



Bricks

1



House

3



Fire Station

8



Hotel

13



City

21

## More on Estimates

---

- Estimation is done during planning ceremonies/meetings
- Estimates are done in *Ideal Time*
- Big estimates are an indicator of too much complexity and too many unknowns (>8)
- Smaller estimates indicate that the team is more certain (<8)
- **Only the team doing the work can estimate the work**

## Reducing Scope

---

- User stories should be estimatable (INVEST)
- If estimates are too big user stories should be split
- When user stories are split, they do not have to equal the sum of the original estimate

## Teams Should Estimate User Stories In Ideal Time



*The amount of time it would take to complete the user story without any distractions.*

## Ideal Time

---

- Do not factor in time out of the office
- Do not factor in time spent responding to emails or IMs
- Do not factor in meetings, events, happy hours, etc.

## How They Relate

---

- **Scope** is measured using **Relative Estimates**
- **Relative Estimates** are most often represented by **Story Points**
- Estimates should be based on **Ideal Time**





## Scoping and Estimation Exercise Solution

## Scoping and Estimation Exercise Solution

Assign each animal a number based on relative

- Size
- Scope
- Complexity to each other

with the numbers provided.

Animal	Number Assignment
Elephant	100
Giraffe	8
Lion	13
Yorkshire Terrier	$\frac{1}{2}$
Bull Mastiff	2
Fox	1
Hippopotamus	40
Tiger	20
Black Bear	3
Mouse	0



# Estimation Techniques

## Estimation

---

- Estimates should be relative
- The team estimates, not the Product Owner
- Drives conversation and consensus
- High Estimates indicate too much scope
- Newly formed teams will estimate more conservatively until trust is built

## Story Points

---

- Numerical representation of relative estimates
- When Story Points are too high, split the user story
- Applied using the Fibonacci sequence

## Fibonacci Sequence

---

**0, 1/2, 1, 2, 3, 5, 8, 13, 20, 40, 100**

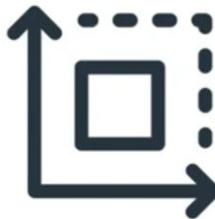
- Not sequential but has gaps
- Used as Story Points
- Forces teams to estimate relatively

*Sequence of values is © Mountain Goat Software, LLC*

## What to Consider When Estimating



Complexity



Size



Level of Effort



Unknowns

## Estimation Techniques

---

- Planning Poker®
- T-Shirt Sizing
- Affinity Estimating



## Estimation Technique: Planning Poker®

- Rapid technique that is good for products at all phases
- Team members are given index cards or planning poker cards
- Cards have the Fibonacci Sequence
- After the user story is discussed and understood by everyone the team points by raising their card at the same time
- Misalignment means there is confusion about the user story so more discussion might be necessary
- Team repeats if the estimates are not relatively aligned

## Estimation Technique: Planning Poker®

As a user I can add my credit card number so that I can make an online payment

0

1

2

3

5

8

13

## Estimation Technique: Planning Poker®

As a user I can add my credit  
card number so that I can  
make an online payment



## Estimation Technique: T-Shirt Sizing

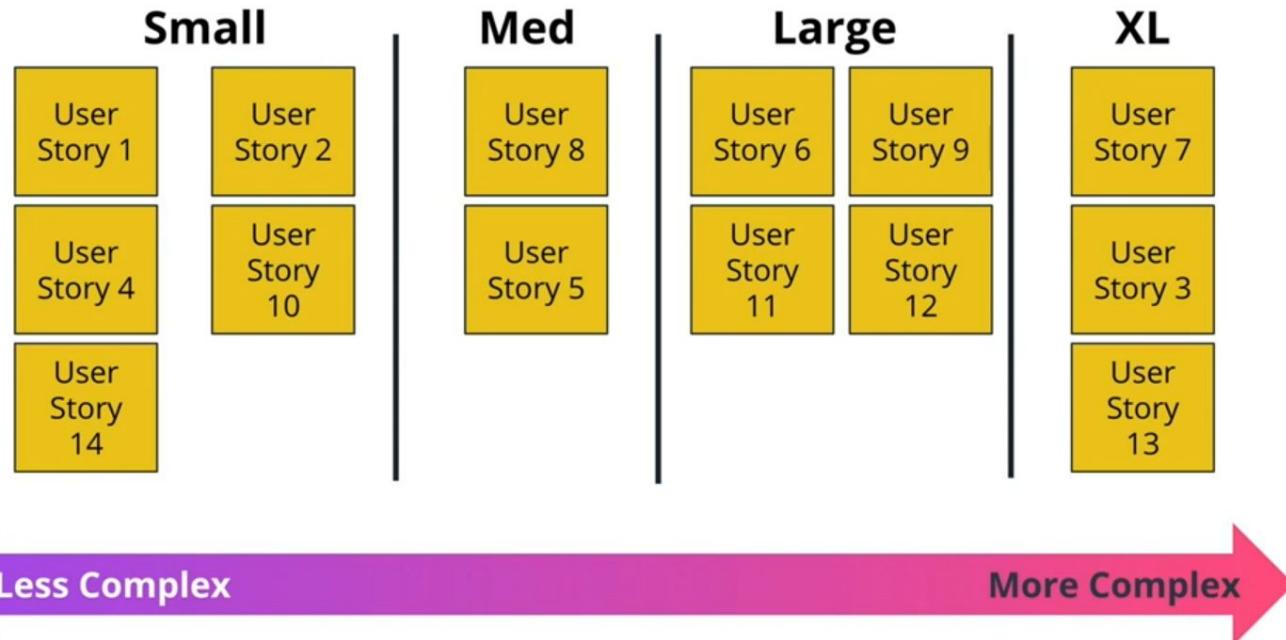
- Rapid estimation technique
- Good for projects that are just beginning or that have a lot of unknowns
- Team uses t-shirt sizes (S,M,L,XL) to estimate
- Good for estimating epics

## Estimation Technique: Affinity Estimation

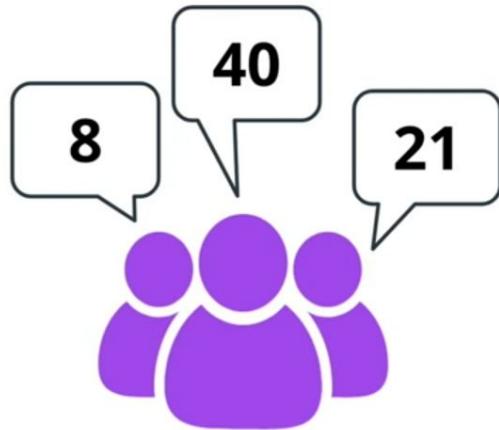
---

- Rapid estimation technique
- Good for new products with a lot of stories to estimate
- To estimate team uses t-shirt sizes (S,M,L,XL), coffee cup sizes, or Fibonacci sequence
- Team places similar user stories into the same bucket

## Estimation Technique: Affinity Estimation

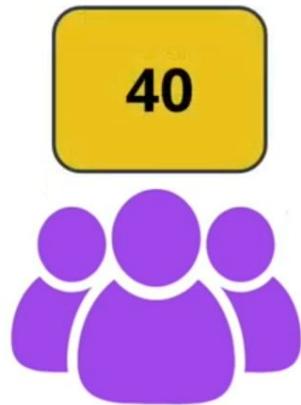


## Disagreements on Estimates



- Estimates should drive consensus
- Sometimes teammates disagree with estimates
- Move on and revisit estimate later during planning

## Disagreements on Estimates



- Estimates should drive consensus
- Sometimes teammates disagree with estimates
- Move on and revisit estimate later during planning
- When agreement can't be reached choose the most conservative (biggest) estimate



## Estimation Techniques Exercise Solution

## Estimation Techniques - Scenario

- You are the Scrum Master
- Development team is building a mobile budget management application
- Team is using Planning Poker® to estimate user stories that will be in the next sprint
- Team disagrees on the Story Points for one of the user stories
- Two members of the team think that the story should be 5 points
- 3 members of the team think that the story should be 3 points.

## Example Answer

---

*If the team can't agree I would take the conservative approach and make the story 5 points. The Story Points are an estimate not an exact. Any person on the team could end up working on the story therefore I want to take the conservative approach and account for everyone's perspective on the size, scope, and complexity.*



What Does "Done" Mean?

## Are we done yet?

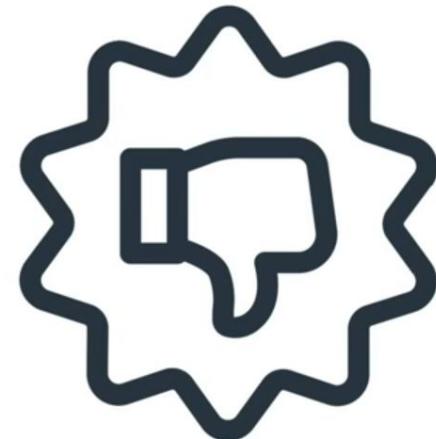
---



Definition of Done



Unclear Direction



Quality Suffers

## Team's Definition of Done

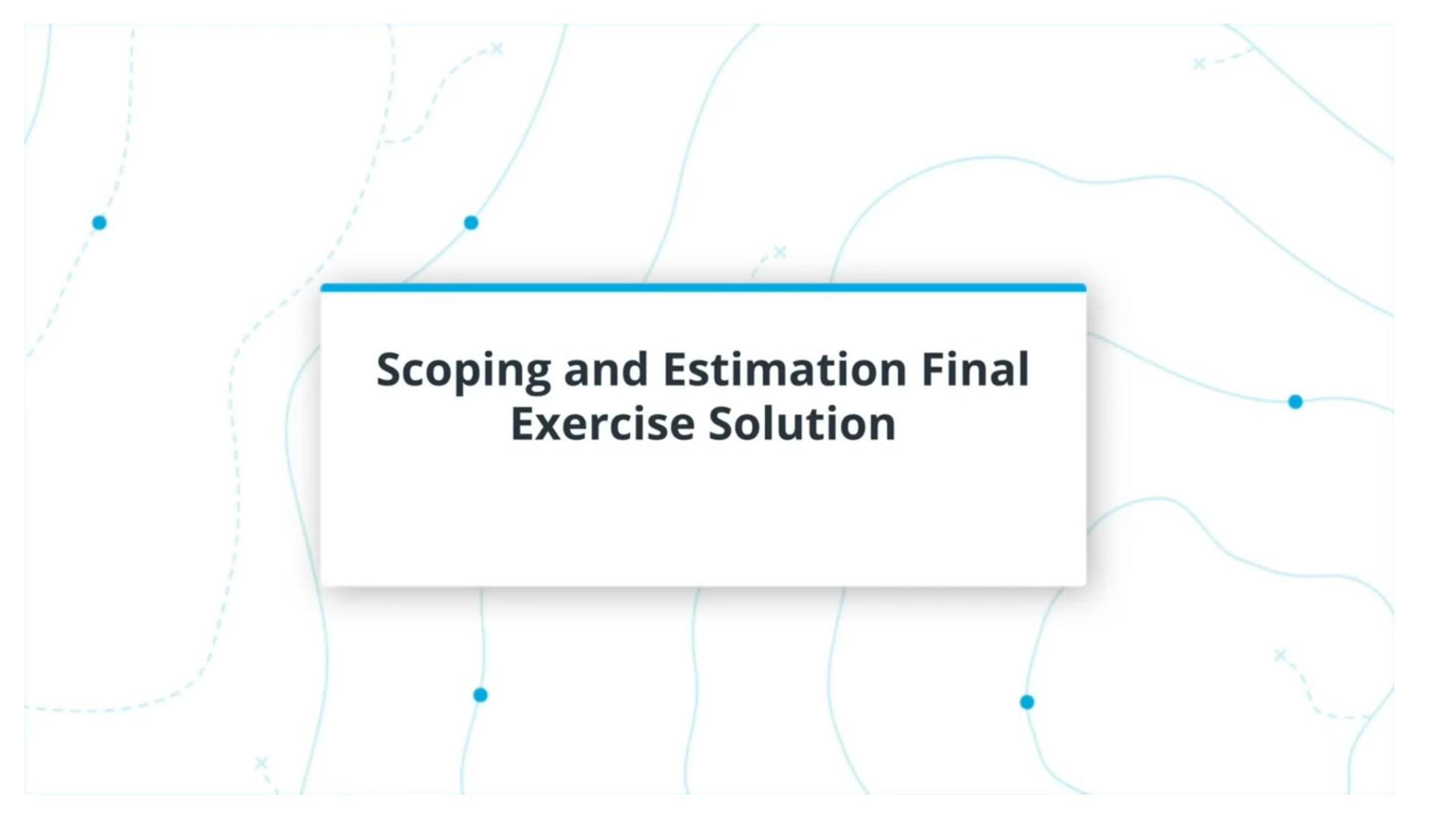


- Agile teams establish a definition of done (DOD) for developing software
- The DOD is the team's agreement for everything that needs to happen for code to be "done"
- If the code is not developed to the standard of "done" for the team then they team must keep working on it
- Teams take their definition of done into account when estimating user stories and breaking the user stories out into smaller tasks
- The DOD can vary from company to company and from team to team

## Acceptance Criteria Definition of Done



- Acceptance criteria provide the team with the parameters for implementing user stories
- The acceptance criteria must be met in order for the story to be accepted as “DONE” by the Product Owner
- This requires conversation and transparency between the team and Product Owner
- We demo the working software to the Product Owner to confirm that the acceptance criteria has been met
- Important to distinguish between the acceptance criteria being met and the Product Owner adding scope



## Scoping and Estimation Final Exercise Solution

## Estimation Techniques - Scenario

- Newly formed development team just finished a sprint planning meeting
- Working on a new sales management product
- Upcoming sprint they just planned will be their first sprint
- Product Owner is frustrated about the outcome of the sprint planning meeting
- Team's estimates were really high and inaccurate for several stories
- Wants team to commit to more work in the sprint because he has really tight timelines to deliver an MVP.

## Example Answer 1

---

*The team's estimates are high because the team is new. The team is estimating conservatively because they have not worked together before, therefore they haven't built the trust and understanding to estimate less conservatively. As the team develops trust and understands each other's strengths and weaknesses they will be able to estimate better.*

## Example Answer 2

---

*The team doesn't understand the product yet, therefore there are a lot of unknowns. As they begin working on the product and understand what they are building there will be less unknowns and they will have more accurate estimates.*

### **Example Answer 3**

---

*The user stories that Kevin the Product Owner provided are too big.*

*In the future he can split or descope user stories to make them smaller so that the team can commit to getting more user stories completed.*

## Lesson Recap

---

- Intro to Scoping and Estimation
- Estimation Techniques
- Defining "Done"



**Scoping**

## Lesson Overview



- Intro to Release and Iteration Planning
- Agile Techniques
- Scrum Specifics

## Learning Objectives

By the end of the lesson you will be able to...

- Explain how release and iteration planning are related
- Identify the outcomes of release and iteration planning
- Plan a Value-Driven MVP
- Define the purpose of a spike
- Define timeboxes and manage the potential outcomes of timeboxes
- Create Scrum plans for release and sprints
- Apply Scrum best practices to set the proper cadence



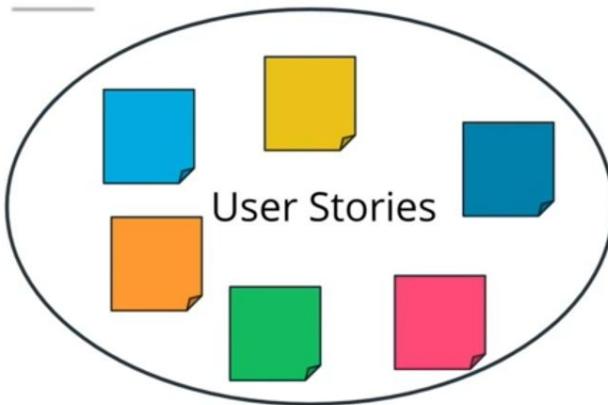
## Why Does Release and Iteration Planning Matter?

## Release and Iteration Planning



- Provides the team with direction
- Helps guide the team at high, yet tactical level

## Release and Iteration Planning



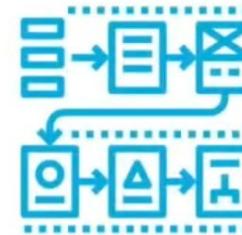
## Release and Iteration Planning



Identify potential  
impediments

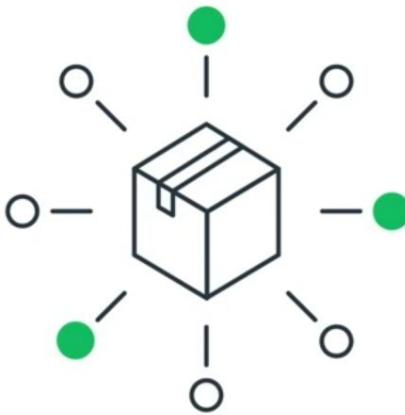


Communicate with  
stakeholders and  
customers



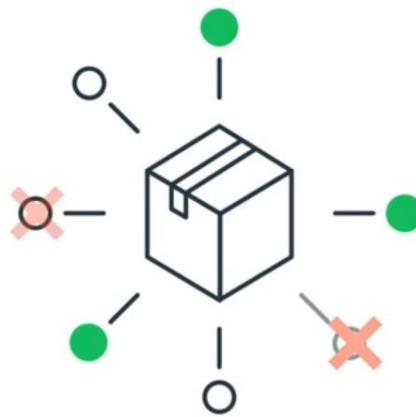
Provides traceability

## Minimum Viable Product (MVP)



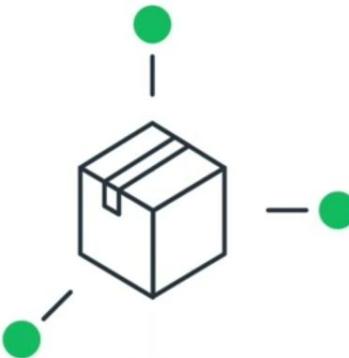
The minimum set of features  
needed to be successful

## Minimum Viable Product (MVP)



The minimum set of features  
needed to be successful

## Minimum Viable Product (MVP)



The minimum set of features  
needed to be successful

## What Sets of Value Should We Deliver?

What themes are we focused on?



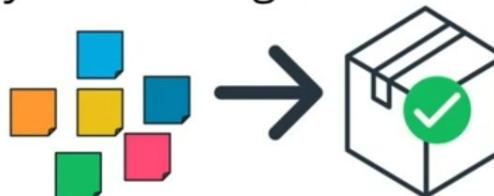
Reporting?  
Importing?  
Upgrades?  
Dashboard  
Redesign?

Will the pieces result in something not fully functioning?

Is this intentional?

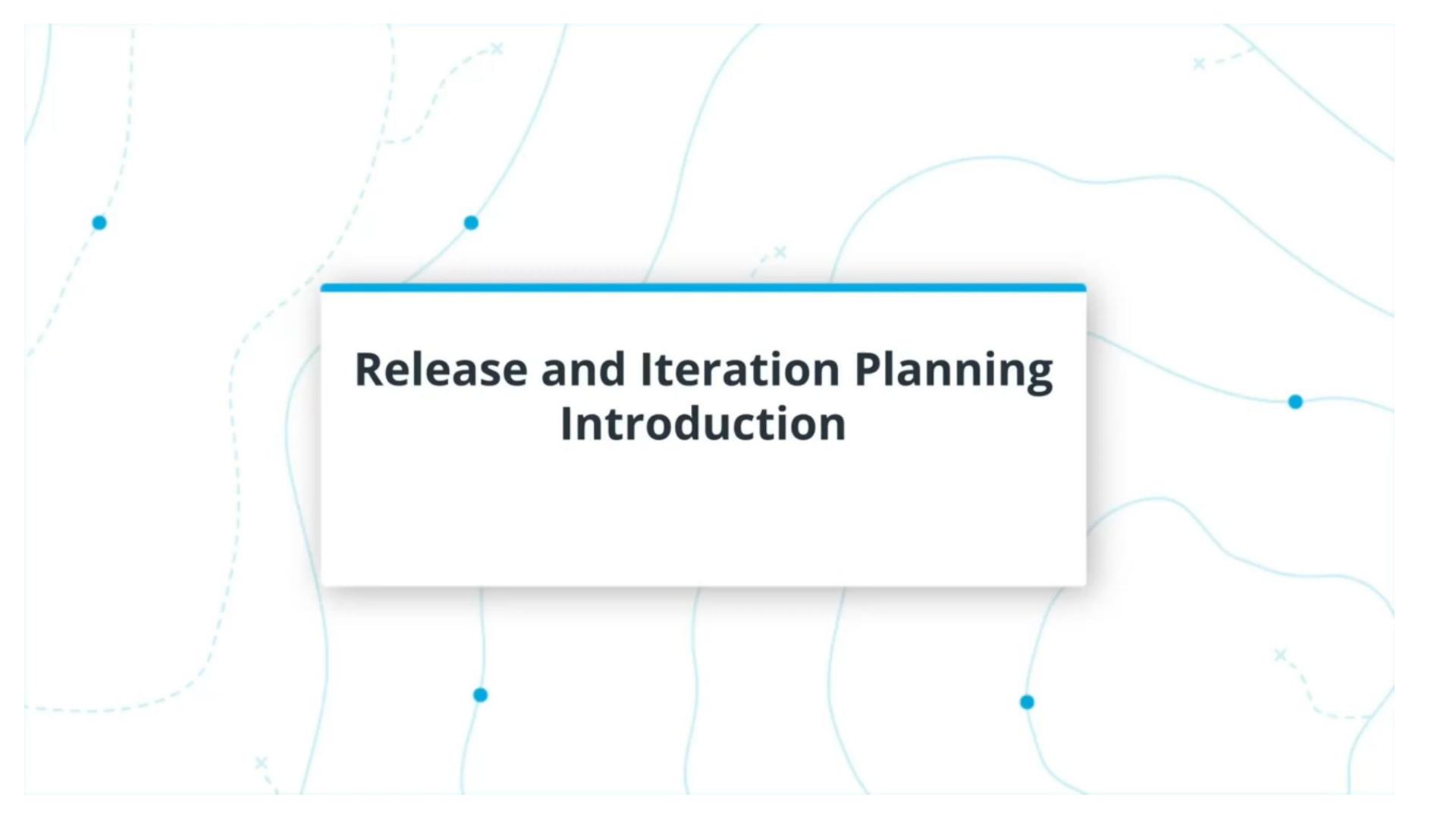


Do the user stories align to deliver a fully functioning feature?



Does this plan equate to value?



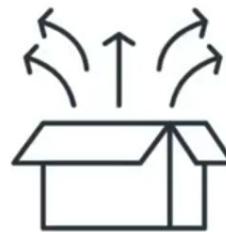


## **Release and Iteration Planning Introduction**

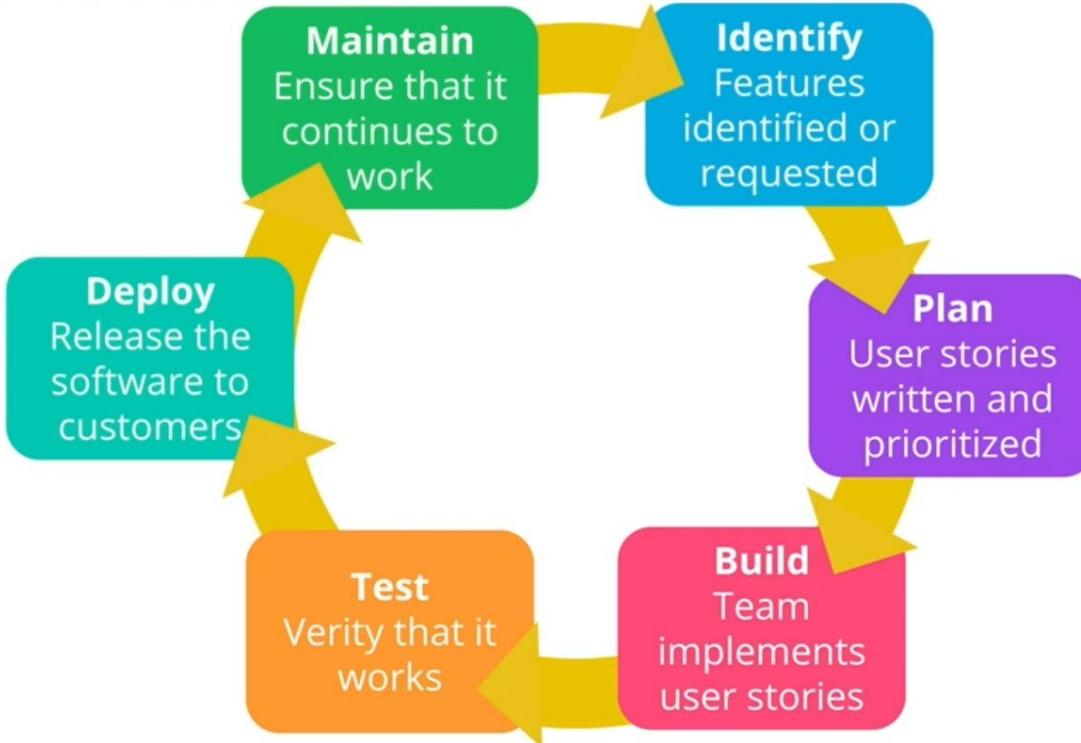
## What is a Release?

---

*The process of giving the latest version of the product to the customer or end user.*



## Getting to the Release...



## What Is a Release Plan?

- Team's plan for completing user stories and functionality that will be included in a release
- Made up of several iterations
- Shows which iteration user stories will be complete
- Aligns user story implementation in logical order
- Created in a release planning meeting

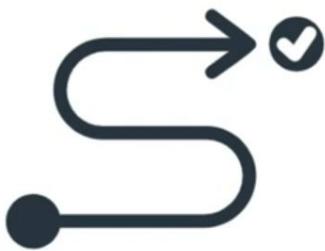


## What Is an Iteration?

---

- A timed cadence used to plan, commit, and deliver work
- Each iteration results in working software
- Scrum: Sprint that is 1- 4 weeks
- Most often team's use 2 week iterations/sprints
- Once a team sets the interval it should not change for the life of the project or product

## What Is an Iteration Plan?



- Detailed & logical implementation plan for user stories
- Team's commitment for completing user stories within iteration/sprint
- Only the user stories that the team can complete are included
- Depends upon team's velocity

## Who Creates Release and Iteration Plans?



- Prioritizes user stories based on value.
- Estimates the user stories in the backlog
- Determines how much of the prioritized work they can complete
- Makes a commitment to deliver the work they can complete



## What Is Needed To Create Release And Iteration Plans?

- A prioritized backlog
- Estimated user stories
- Team's velocity
- Team's availability



## Using the Plan

---

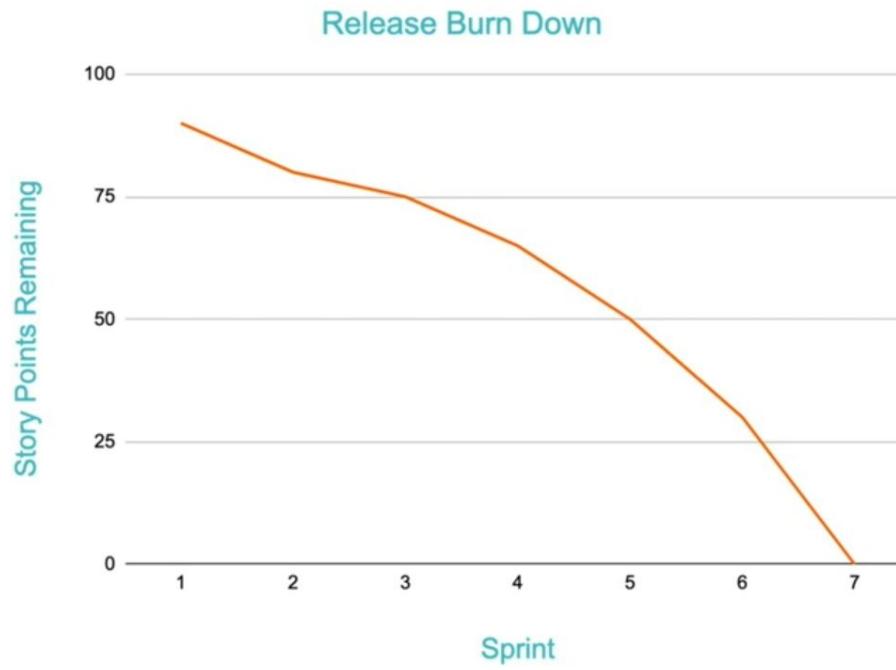
The team works toward delivering the functionality that has been outlined in the iteration and release plan so that working software can be delivered to the customer.

## Burning Down the Release

---

- Measures how much work is remaining in the release
- Lets us know if we are on track
- Helps us understand the impact of adding user stories to the release
- Displayed on Graph:
  - Y-Axis represents Story Points
  - X-Axis represents Sprints or weeks

# Release Burndown Chart



## **Before Shipping Software...**

---

- Test to verify what is needed is there
- Test to validate that it is working as intended
- Demo to customer
- Testing is done continually
- Report defects (bugs)

## **After Software is Shipped...**

---

- Test to verify what is needed is there
- Test to validate that it is working as intended
- Testing is done continually
- Report escaped defects (bugs)

## Testing

Manual



Automated



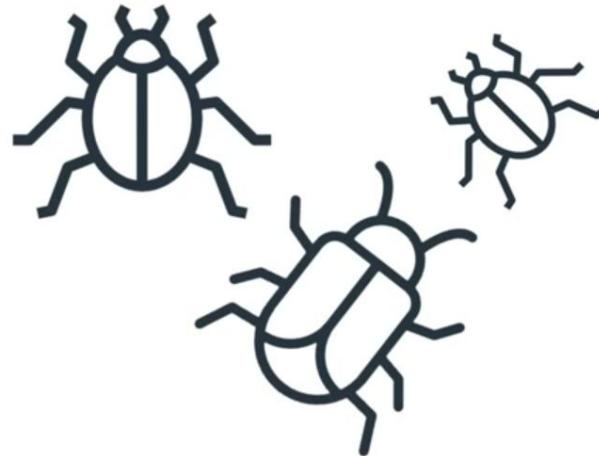
Combination



## Managing Defects

---

- Value and impact
- Add to backlog
- All software has defects



## Managing Defects



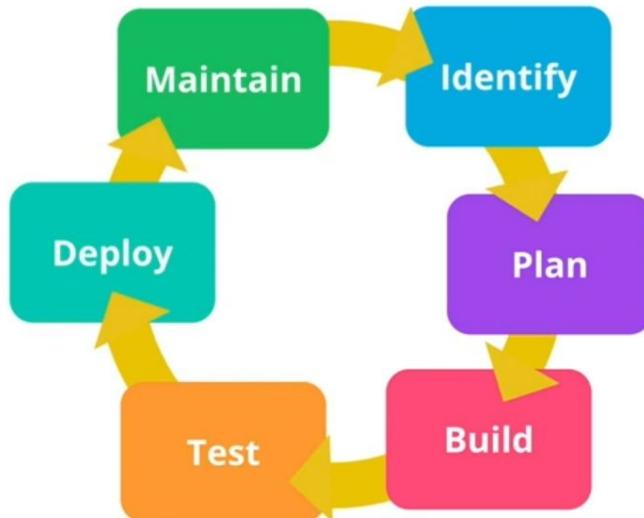
- Software can be released with defects if it still delivers value
- Be transparent with customers about defects and plans to fix them

## Escaped Defects

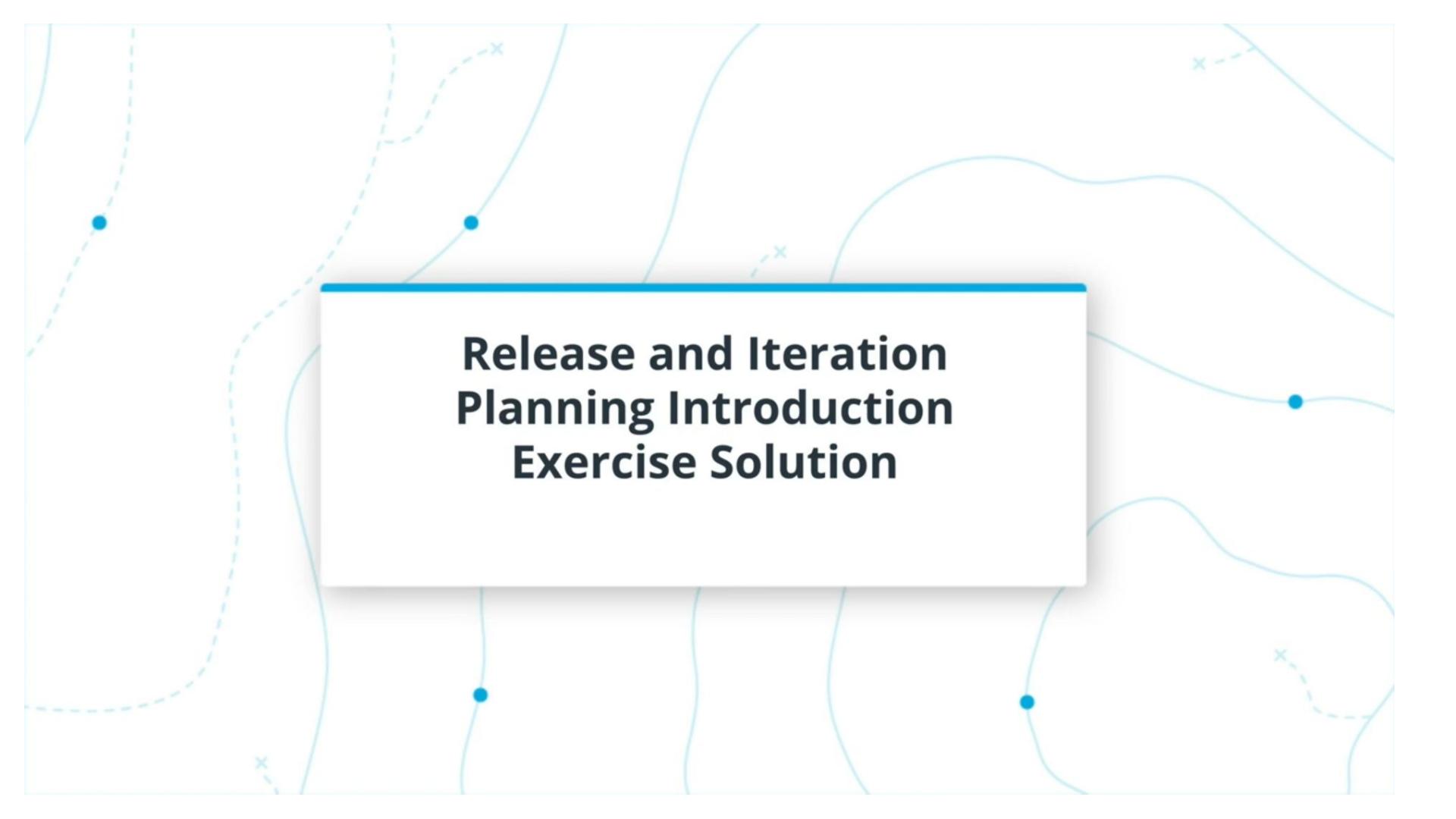
---

Defects that are found by the customer after the software has been released. Increases in escaped defects can be a sign of testing and development problems.

## Summary



- Continuously release the latest versions of the software continuously
- Release and iteration plans are tactical plans that guide development
- Testing is an essential part of building
- Defects should be added to the backlog to be prioritized



# **Release and Iteration Planning Introduction Exercise Solution**

## Release and Iteration Planning

Scenario: You are the Product Owner for a fitness product

- Finalize the release plan for the next couple of releases three weeks ago
- Release plans have been communicated to your stakeholders.
- Since that time 2 members of the development team are going to leave the company
- Another developer will be on paternity leave for 4 weeks

## Example Answer 1

---

*The Product Owner should let the stakeholders and investors know that the development team is losing 2 team members which will impact the scope of the features that were planned for upcoming releases.*

## Example Answer 2

---

*The Product Owner should work with the stakeholders to re-prioritize the work to deliver the highest value and determine the scope of the “must-have” features for the next releases.*

### **Example Answer 3**

---

*The Product Owner should work with the remaining team members to rescope and replan the work for each release based on the changes to the team and their impact.*

## **Example Answer 4**

---

*The team member going on paternity leave should not have an impact on the release plan because their time away from the team should have been considered during the release planning that took place three weeks ago. The team should be holding each other accountable for communicating expected out of office plans as vacation, paternity/maternity leave, training, military duty, etc because this impacts the teams capacity and the amount of work they can deliver.*



## Agile Techniques for Release and Iteration Planning

## Execute on the Plans

- MVP
- Velocity
- Continuous Integration (CI)
- Continuous Delivery (CD)
- Managing Risks



## Minimum Viable Product (MVP)



The **minimal** set of features that the team needs to deliver for the product and/or customer to be **successful**

## Minimum Viable Product (MVP)



## Velocity

The amount of scope that the team is able to complete each iteration



- Helps the team commit appropriately
- Measured by the average number of Story Points completed each iteration

## Velocity For New Teams

---

- New teams need to complete 3-5 sprints to establish a velocity
- New teams estimate velocity for first 3-5 sprints



## Velocity Equation

$$\frac{\text{Total # of Story Points}}{\# \text{ of Sprints}} = \text{Velocity}$$

$$\frac{90 \text{ Story Points}}{5 \text{ Sprints}} = 18 \text{ points/sprint}$$

## Agile Approach To Managing Risks

---

- Risk are threats to the product's success
- Tackle the highest risk, highest value user stories first
- Fail fast mindset
- Prove that we CAN or CAN'T do it

## Timeboxes

---

- A set period of time to complete work
- Parameter of time helps team avoid spending too much time creating a solution
- Timeboxes result in working software or proof that a concept will or won't work

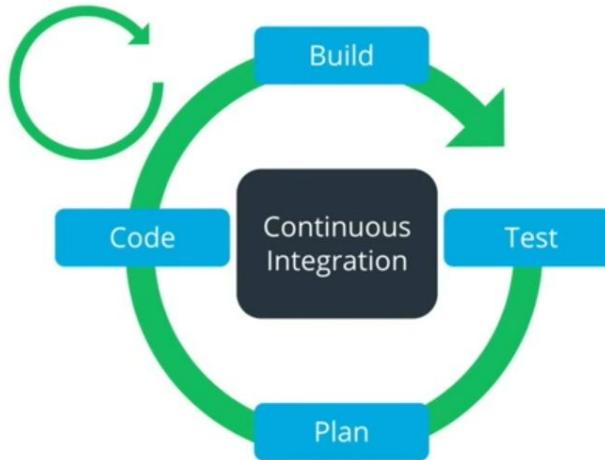
## Spikes

---

- A timeboxed deep dive that a development team uses to discover or prove if an approach will work
- Used to determine technical and functional approaches
- Risk Management technique

# Continuous Integration

The art of frequently integrating a developer's newly built code into the team/product code repository



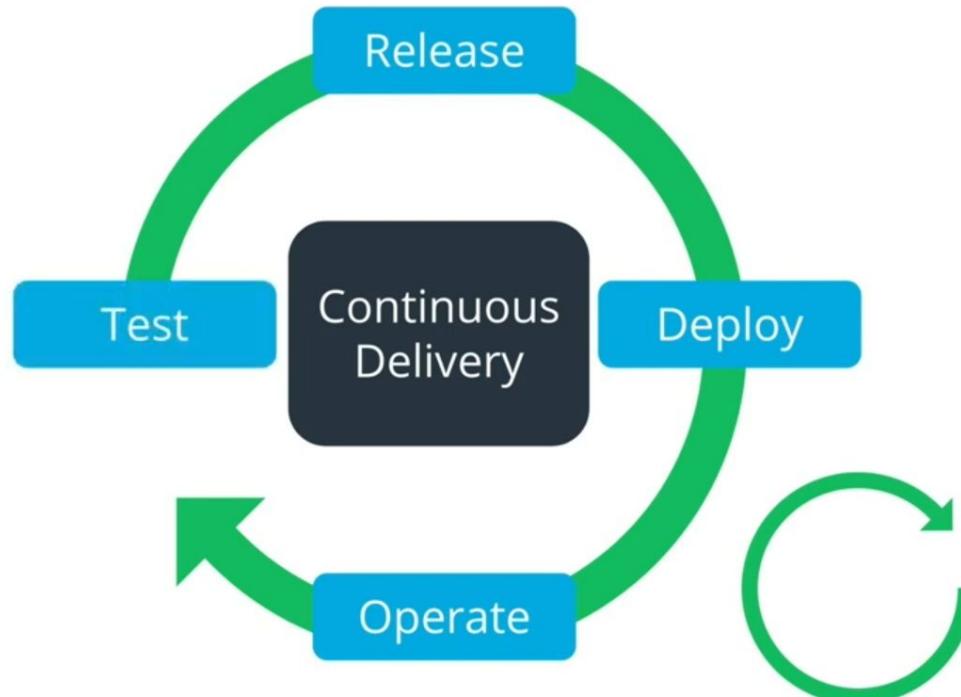
- Done in a development environment first
- Integrations help discover problems with code faster
- Fix problems faster
- Less problems once released

## Continuous Delivery

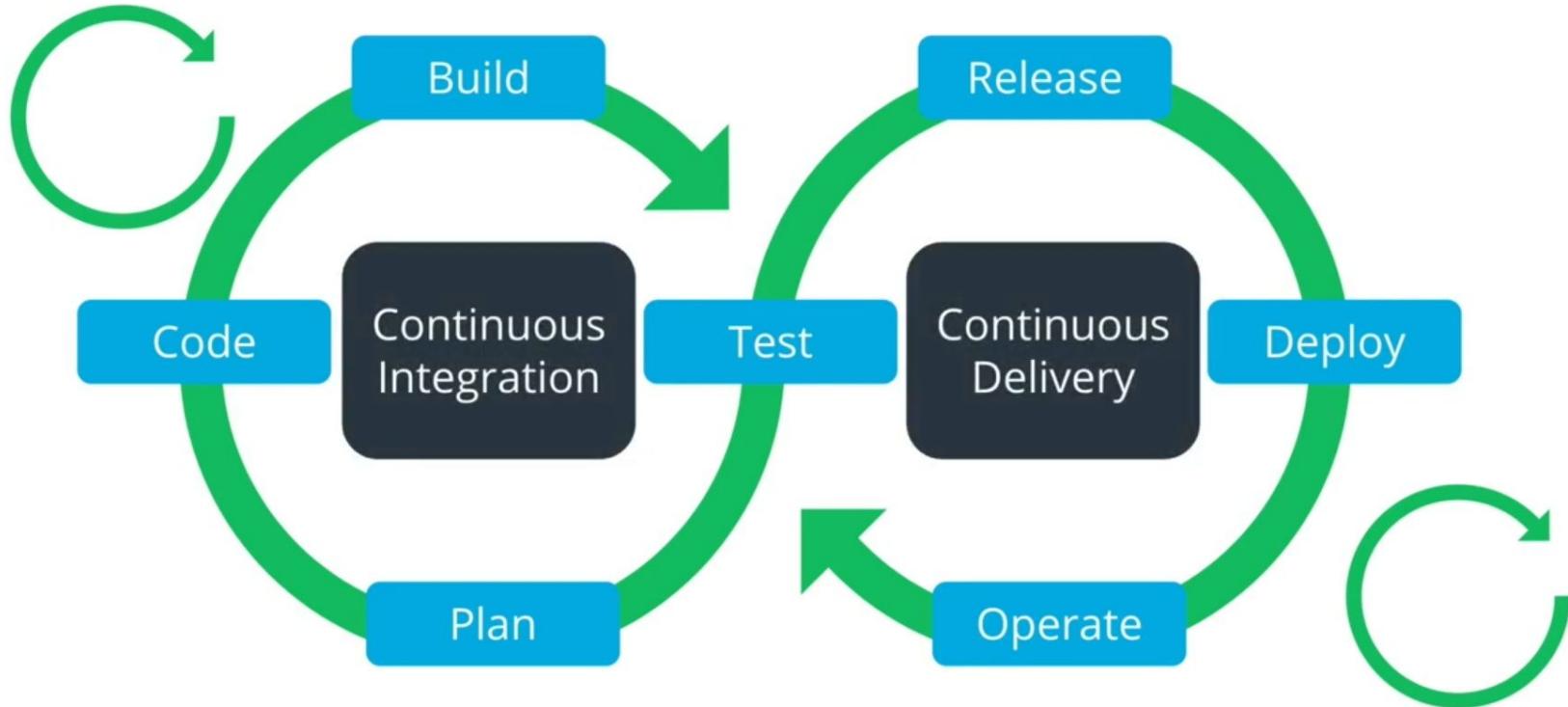
---

- Perform frequent demo working software to customers
- Frequently release the working software to customers
- Give value earlier and more often
- Receive and implement feedback

## Continuous Delivery



## Continuous Integration and Delivery



## Agile Techniques for R+I Planning Overview

- The MVP concept is the basis for releasing software
- Velocity helps ensure that teams are operating at a sustainable pace
- Timeboxes should be used as guardrails for creating solutions
- The process of Continuous Integration and Delivery (CI/CD) helps teams discover problems faster



## Agile Techniques for Release and Iteration Planning Exercise Solution

## Agile Techniques for R+I Planning Exercise Solution

A. What is each team's average velocity?

Sprints	Team A	Team B
1	12	31
2	20	23
3	16	19
4	18	25
5	17	24
Average = $\frac{\text{Sprint1} + \dots + \text{Sprint5}}{\# \text{ of sprints}}$		
Average Velocity	17	24

## Agile Techniques for R+I Planning Exercise Solution

A. What is each team's average velocity?

Team A = 17

Team B = 24

B. Which team is having more success?

- Velocity can't determine the success of a team.
- Velocity can't be used to compare teams.
- Many factors account for the velocity of a team including but not limited to:
  - experience of the team
  - estimating techniques
  - complexity of the work
  - availability of the team



## Scrum Specific Planning

## Scrum Planning Is Driven By

---

- Release Plan
- Scrum Events
- Feedback loops
- Timeboxes

## Release Plan

Product Backlog	Story Points
Create Playlist	1
Add Playlist	3
Search by artist	0.5
Make payment	5
Change payment	8
Automatic timeout	3
Search by title of song	1
Filter by genre	13
Share playlist	20
Follow an artist	8
Like an artist	5
Download album	3
Download Song	8

Release

## Scrum Events: Sprint Planning

---

- Product Owner presents Prioritized backlog
- Team reviews the priorities and gains clarity
- Team commits to deliver work they can complete
- Work goes into the sprint backlog
- Timing: 2-4 hours; Every 2 weeks before the sprint starts

# Sprint Plan

Product Backlog	Story Points	
Create Playlist	1	
Add Playlist	3	
Search by artist	0.5	
Make payment	5	Sprint 1
Change payment	8	
Automatic timeout	3	
Search by title of song	1	
Filter by genre	13	
Share playlist	20	Release
Follow an artist	8	
Like an artist	5	
Download album	3	
Download Song	8	

# Sprint Plan

Sprint Backlog

Sprint Backlog	Story Points
Create Playlist	1
Add Playlist	3
Search by artist	0.5

Product Backlog

Product Backlog	Story Points
Make payment	5
Change payment	8
Automatic timeout	3
Search by title of song	1
Filter by genre	13
Share playlist	20
Follow an artist	8
Like an artist	5
Download album	3
Download Song	8

★Release

## Sprint Planning Responsibilities



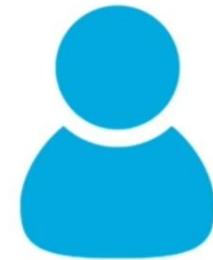
### Product Owner

Defines and  
Prioritizes the work



### Development Team

Estimates the work  
and commits to  
delivering



### Scrum Master

Facilitates the  
planning ceremony

## **Product Owner and Sprint Planning**

---

- Leads the sprint planning meeting
- Adds detailed user stories to the backlog
- Prioritizes the work in the product backlog



## The Development Team and Sprint Planning



- Estimates the work
- Determines the best way to do the work
- Makes commitment to deliver work that they can complete
- Determines the sprint backlog

## **Scrum Master and Sprint Planning**

---



- Facilitates the sprint planning meeting
- Assists Product Owner with management and prioritization
- Ensures team does not overcommit to work for the sprint

## Work Commences

---

After the sprint backlog is set the team starts the 2 week sprint



## Scrum Events: Daily Stand Up

Team members make daily commitment to execute plan

- What did you do?
- What do you plan to do?
- What impediments are you facing?



**15 mins  
Every day**

## Scrum Events: Sprint Review

Team members demo working software to customers

- Team shares completed work
- Customers give feedback



**1 Hour**

**Every 2 weeks at the end of the sprint**

## Scrum Events: Retrospective

Team looks for ways to improve working processes

- What went well the past 2 weeks?
- What didn't go well the past 2 weeks?
- What action should we take to improve?



**1-2 Hours**

**Every 2 weeks after the sprint has ended**

## Feedback Loops

---

The cadence of Scrum events drive constant flow of feedback to help teams and the product improve as time goes on.



## Work Not Done

---

Work that is not completed within the sprint's timebox should be placed in the product backlog so that the Product Owner can prioritize.



## Scrum Specific Planning Exercise Solution

## Scrum Specific Planning

Scenario: Development team for a clothing retailer

- Team has been using Scrum for over a year
- Length of sprints depends on the amount of work



## Example Answer 1

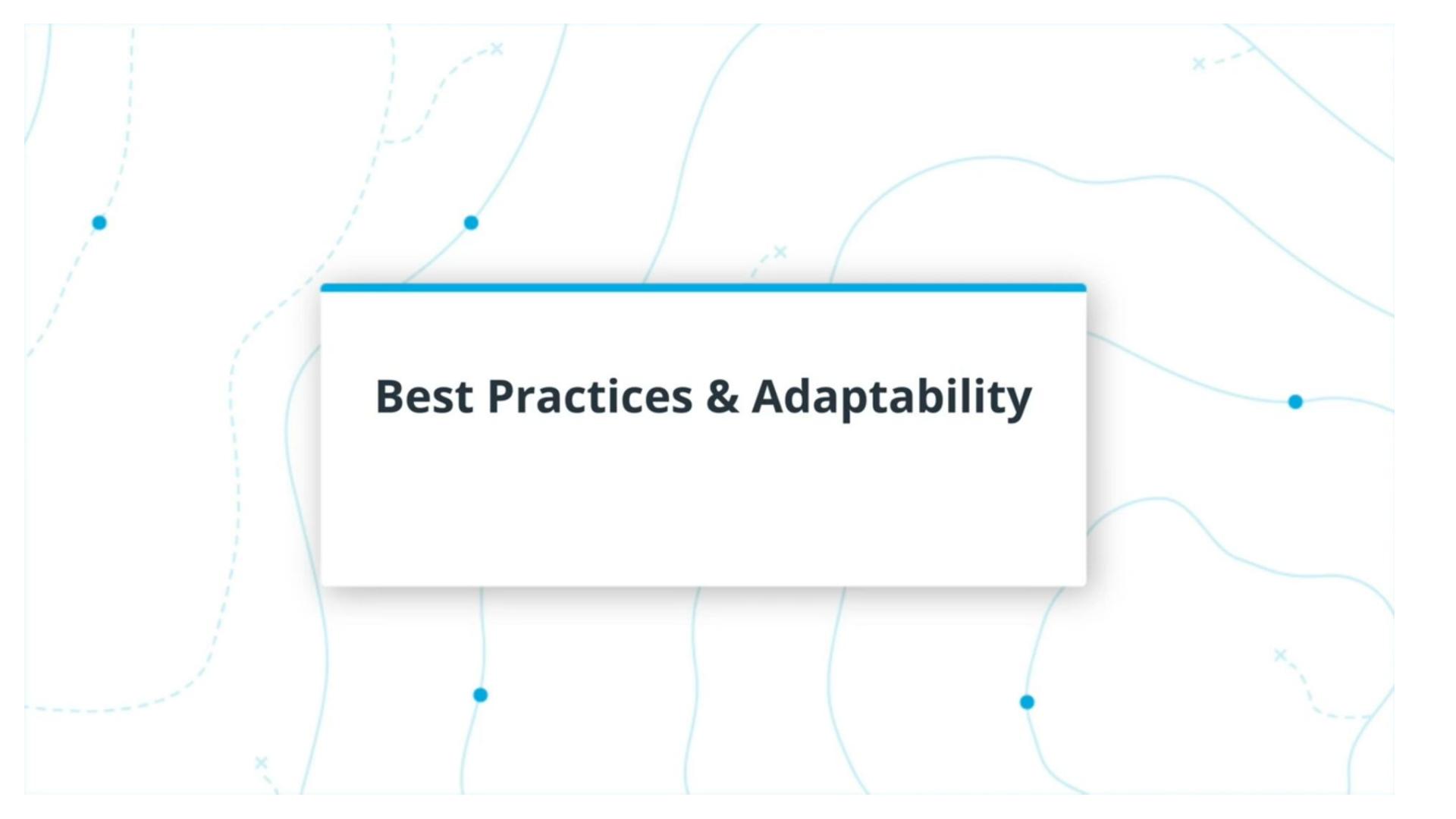
---

*The team needs to choose the length of time their sprints should last and not change them. Changing sprint durations throughout the year will prevent the team from establishing a velocity. Without a velocity the team will not be able to plan releases or be predictable about work.*

## Example Answer 2

---

*Changing sprint lengths will also prevent the team from establishing predictability. Good Agile teams can make highly accurate predictions about when work will be complete. Without an established sprint duration the team can never reach highly accurate predictions.*



## Best Practices & Adaptability

## Best Practices to Avoid Bad Habits

---



- Each developer must integrate their code frequently
- It is a best practice to integrate code daily
- Improves quality because problems are found faster
- Set continuous integration standards as a part of the DOD
- Example: Every engineer must merge code daily

## Adapt to the Environment

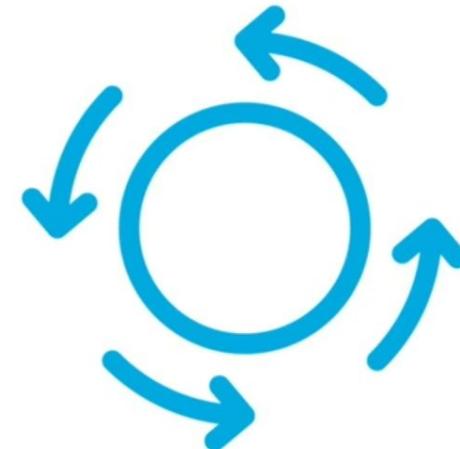
---

- Don't be too prescriptive
- Adjust when necessary
- Be Creative and Have Fun!



## Cyclical Nature of Planning

- Agile Planning is Cyclical
- Prioritizing is Cyclical
- Continuously Deliver Value





## **Release and Iteration Planning Final Exercise Solution**

## Release and Iteration Planning Final

Scenario:

- Team changing from waterfall to an Agile delivery management process
- Team is preparing for an upcoming release that contains completed work from 7 sprints.
- To prepare for the release the team starts integrating the code from each sprint and they notice hundreds of defects.
- More defects than expected unsure they can fix the defects in time for the release

## Example Answer 1

---

*The team can leverage continuous integration. As they build working software they should be integrating it regularly to identify defects as early and as often as possible.*

## Example Answer 2

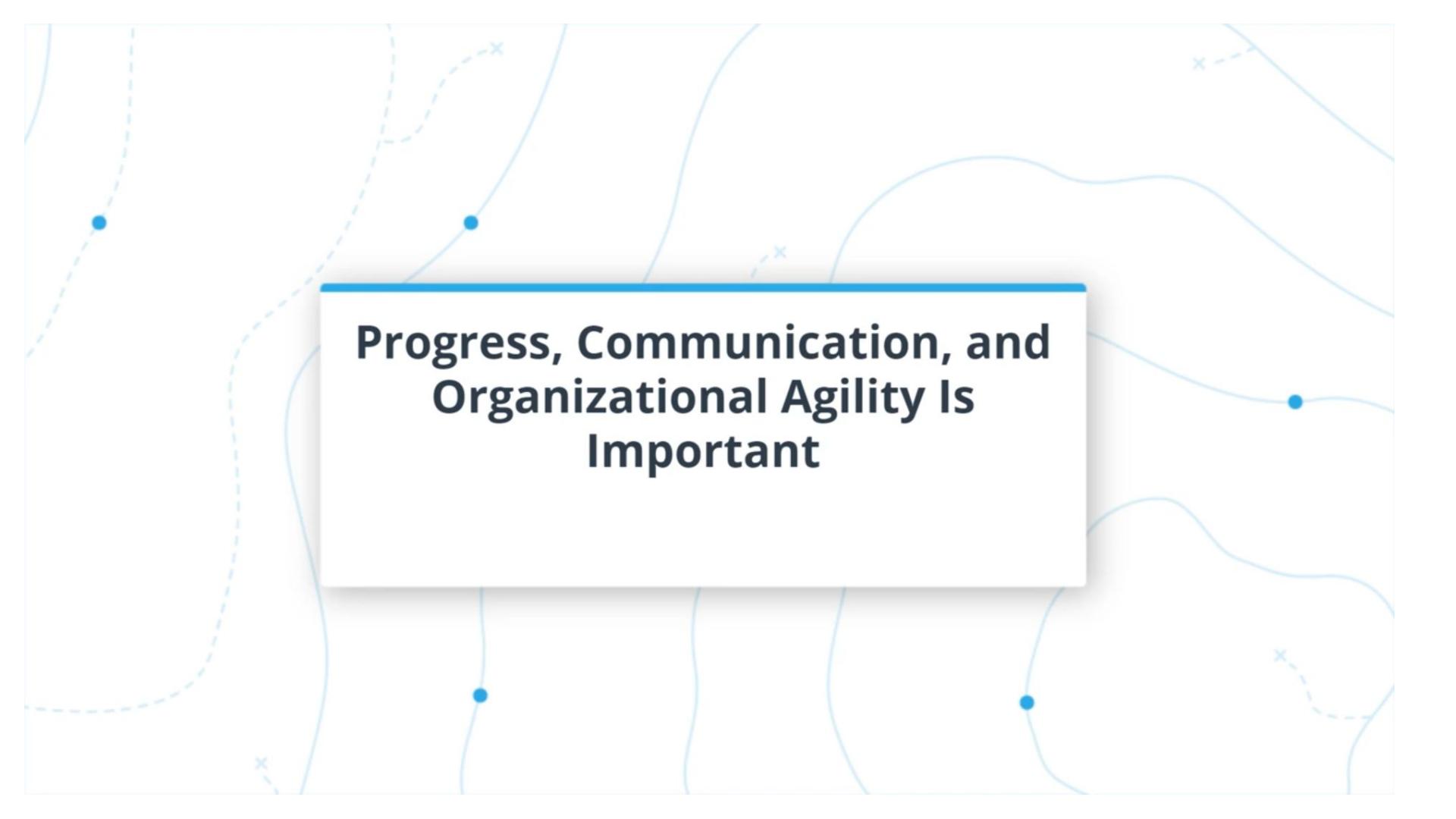
---

*Defects represent value just like a user story. The release does not have to be delayed if the defects won't prevent the users from getting the value they need delivered to them. The team should add the defects to the product backlog and the Product Owner should prioritize the most critical and "valuable" defects the same way they would new features.*

# Course Overview



- Intro to Planning
- Vision and Roadmaps
- Features, Epics, User Stories, and Acceptance Criteria
- Intro to Backlogs
- Managing and Organizing the Backlog
- Prioritization Techniques
- Intro to Scoping and Estimation
- Estimation Techniques
- Defining "Done"
- Intro to Release and Iteration Planning
- Agile Techniques
- Scrum Specifics

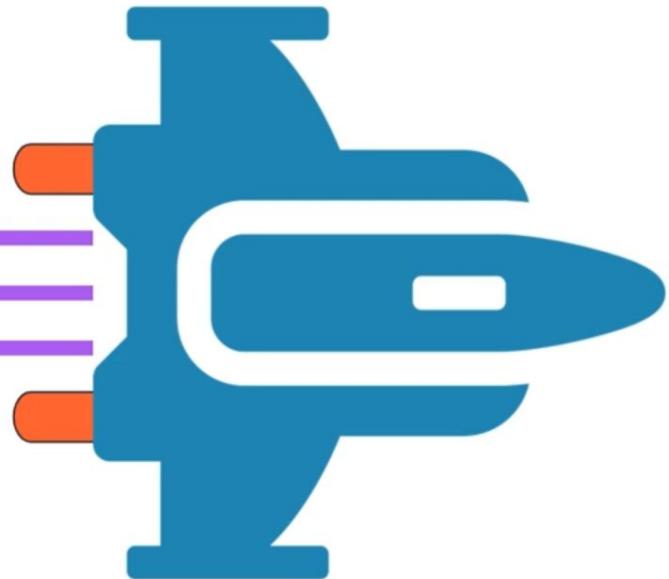


**Progress, Communication, and  
Organizational Agility Is  
Important**

## Air Force Example

---

AGILE



## Air Force Example



Top Down  
Support



Technical  
Knowledge



Acceleration of  
Critical Programs



# **Progress, Communication, and Organizational Agility Stakeholders**

## Agile Communication & Stakeholders



**Product  
Owner aka  
Voice of the  
Customer**



**Scrum  
Master**



**Team**

## Agile Communication & Stakeholders



Small or  
Large Orgs



Technical

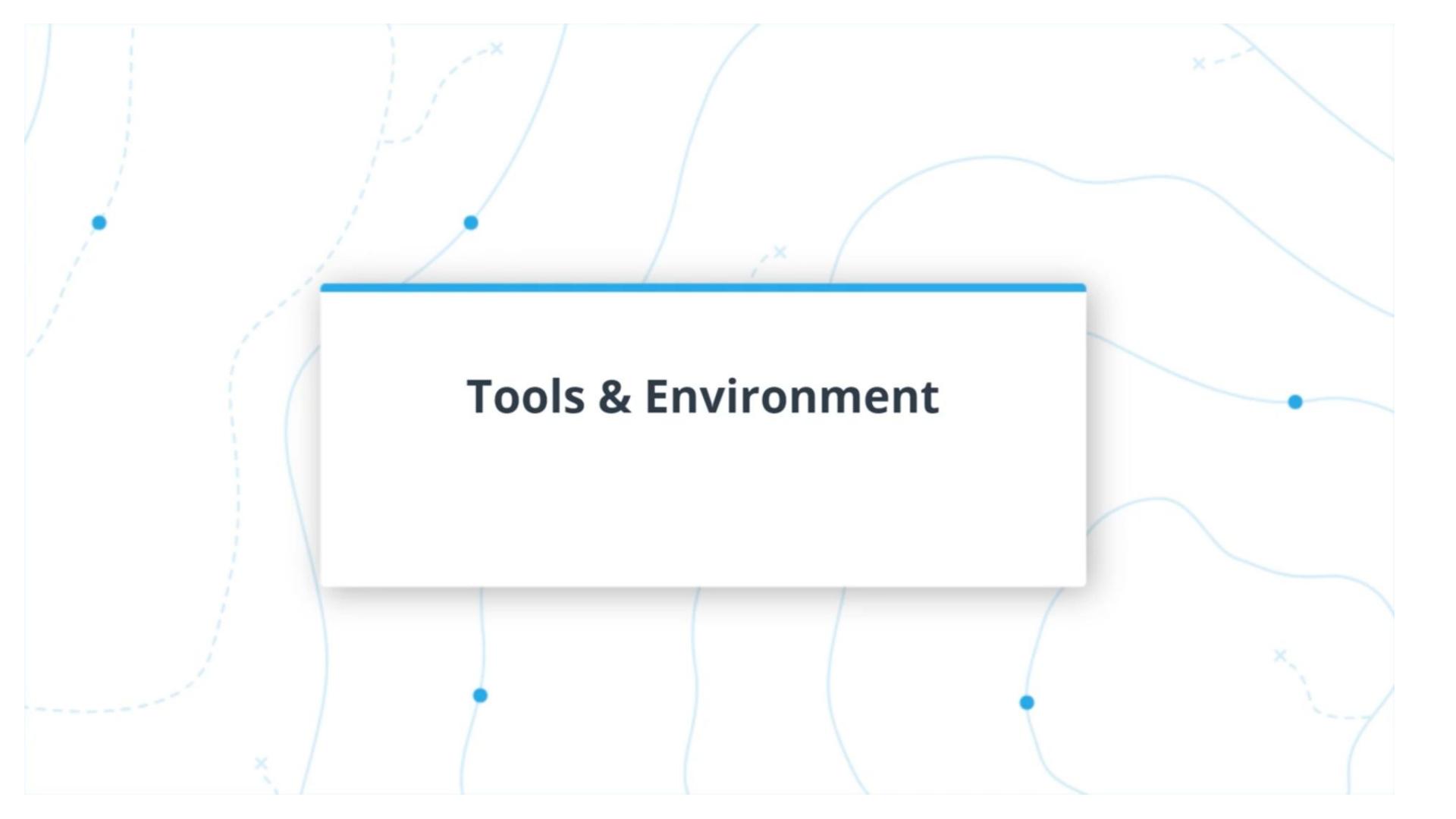


Non-Technical

## Agile Mindset



**Agile Mindset**



## Tools & Environment

## Tools and Environment



Laptops or Desktop

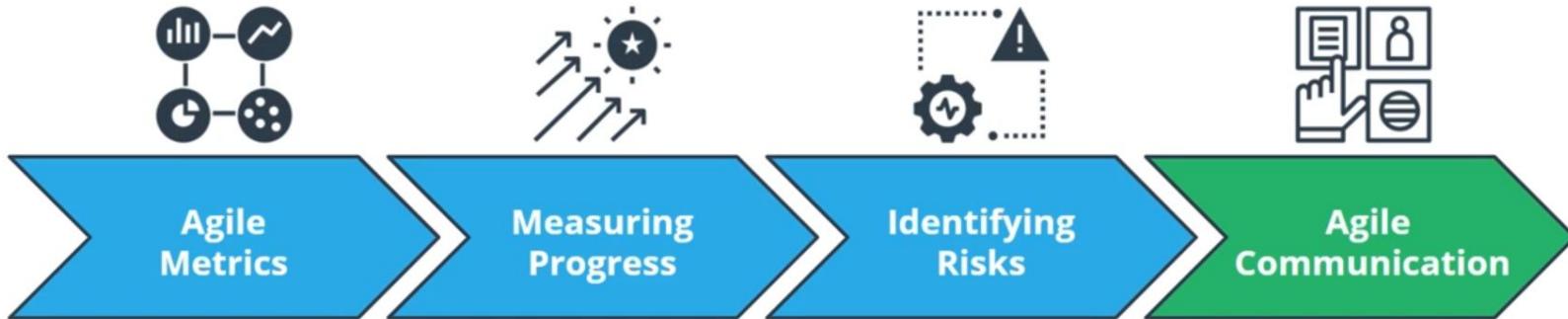


Spreadsheets

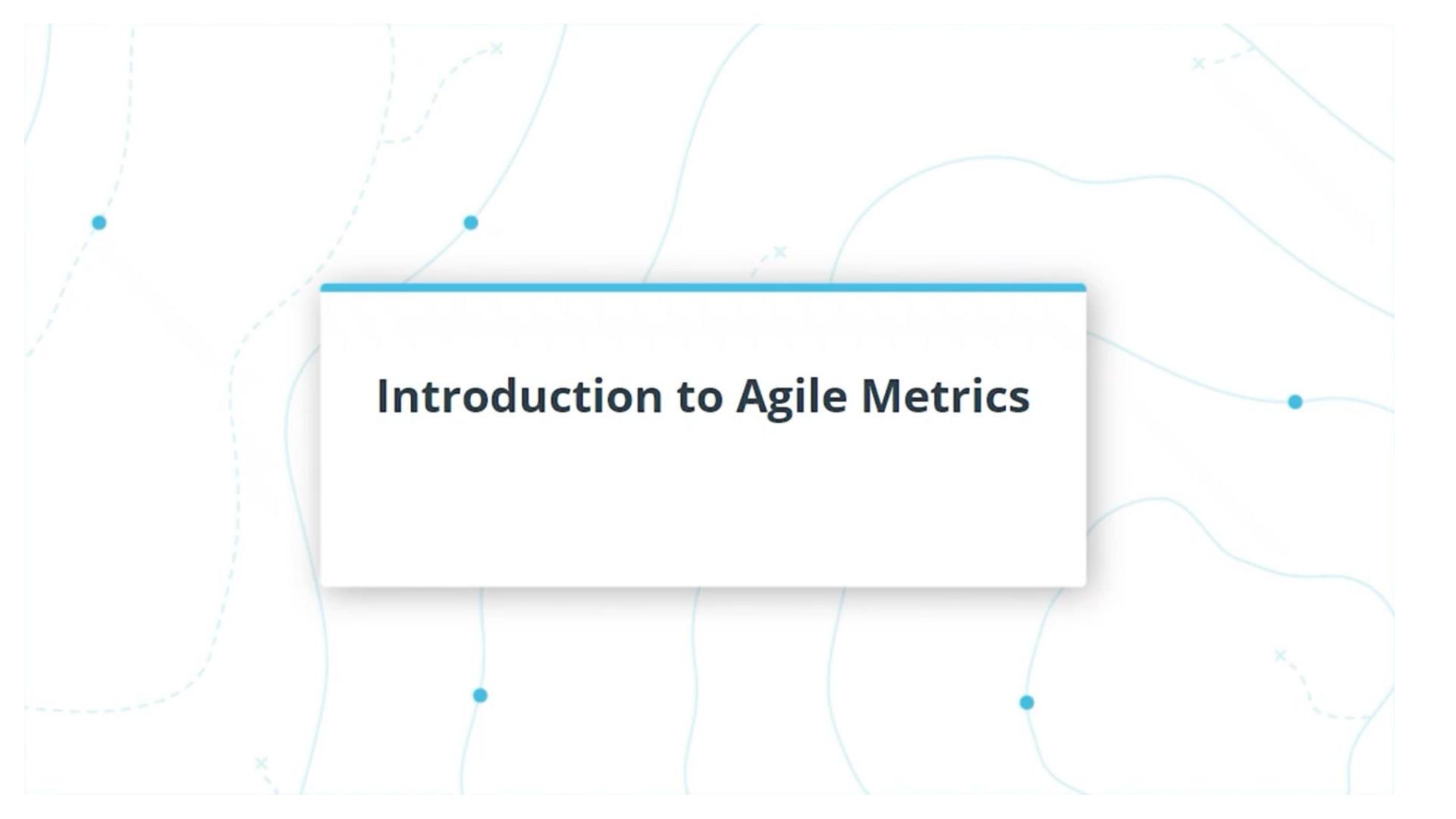


Video Recording  
Device (Optional)

# Course Overview



- Intro to Metrics
  - Scrum Metrics
  - Kanban Metrics
  - Escaped Defects
- Continuous Improvement
  - Retrospectives
  - Information Radiators
- Adapting to Change
  - Technical Debt
  - Failure Patterns/ Mitigation Strategies
- Communication Strategies
  - Internal Communication
  - External Communication

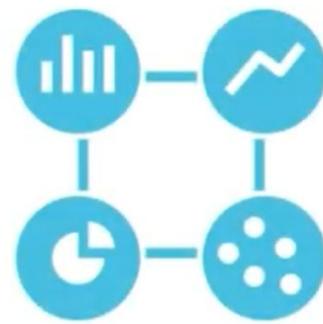


# Introduction to Agile Metrics

## Agile is Two Sided



Soft Skills



Metrics

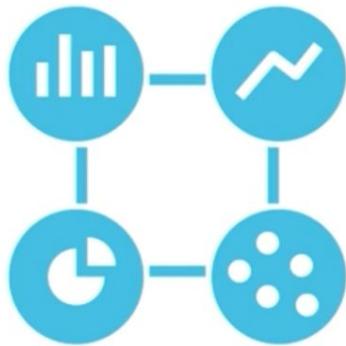
## Soft Skills

---



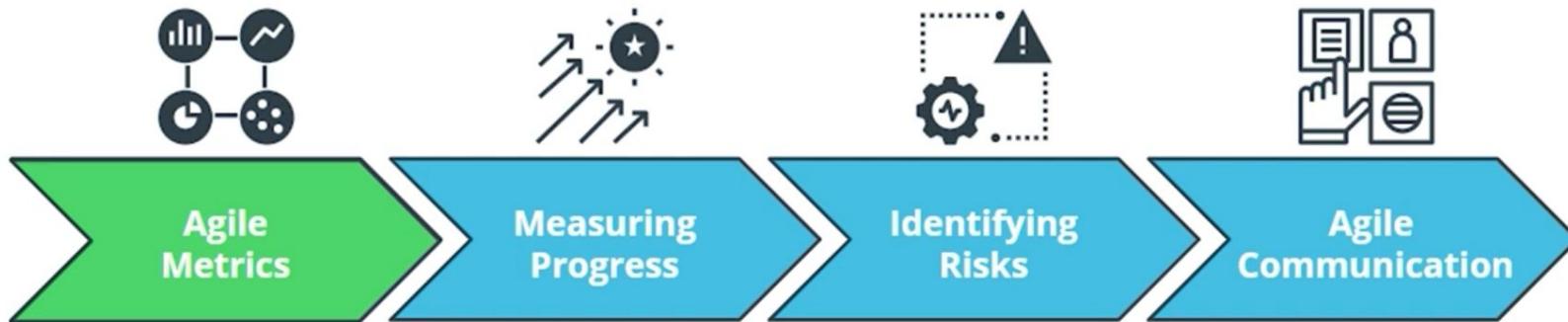
- Working Together
- Collaborating
- Swarming

## Metrics



- Cannot be disputed
- Information about current project status
- Accurate prediction of completion

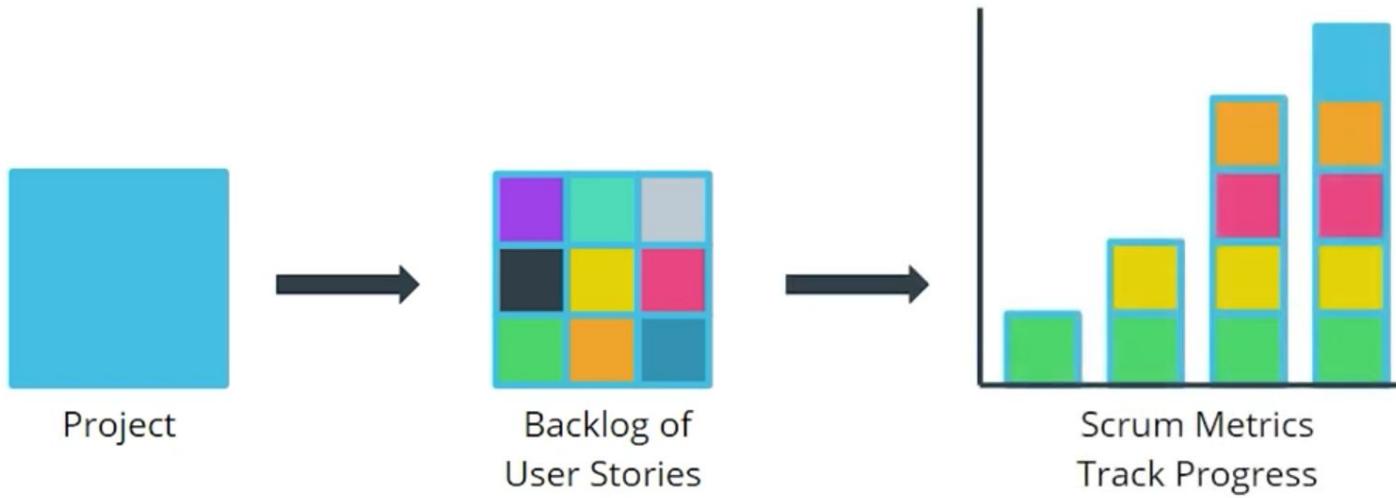
# Lesson Overview



- Intro to Metrics
- Scrum Metrics
- Kanban Metrics
- Escaped Defects

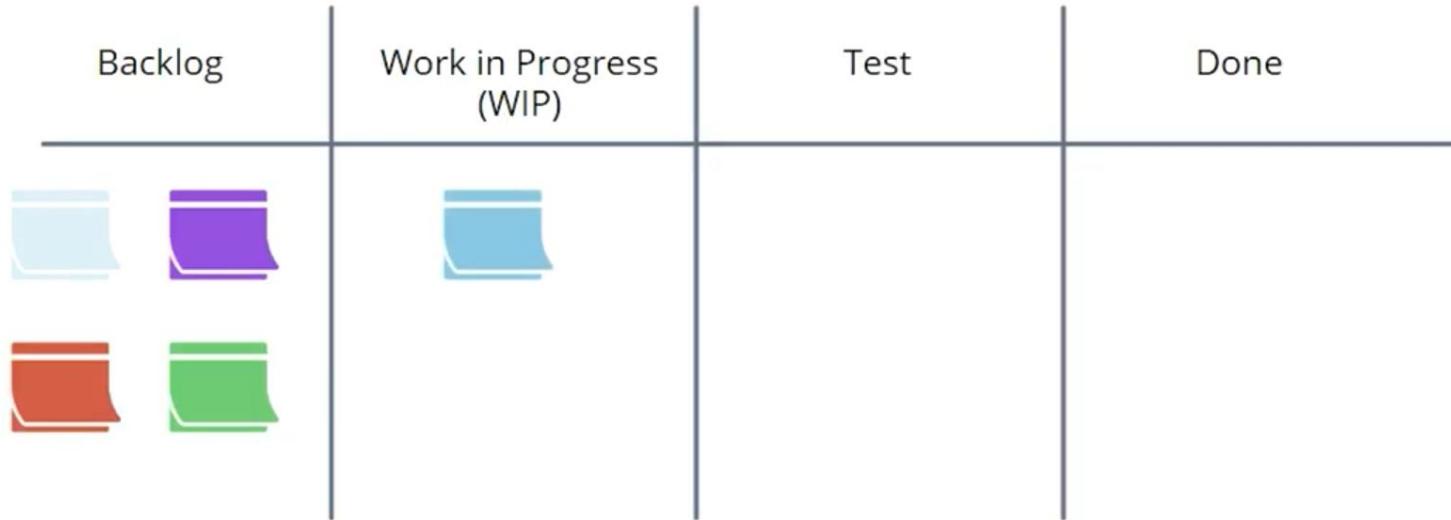
## Scrum Metrics

In Scrum it's about a backlog of work to bring value to the customer



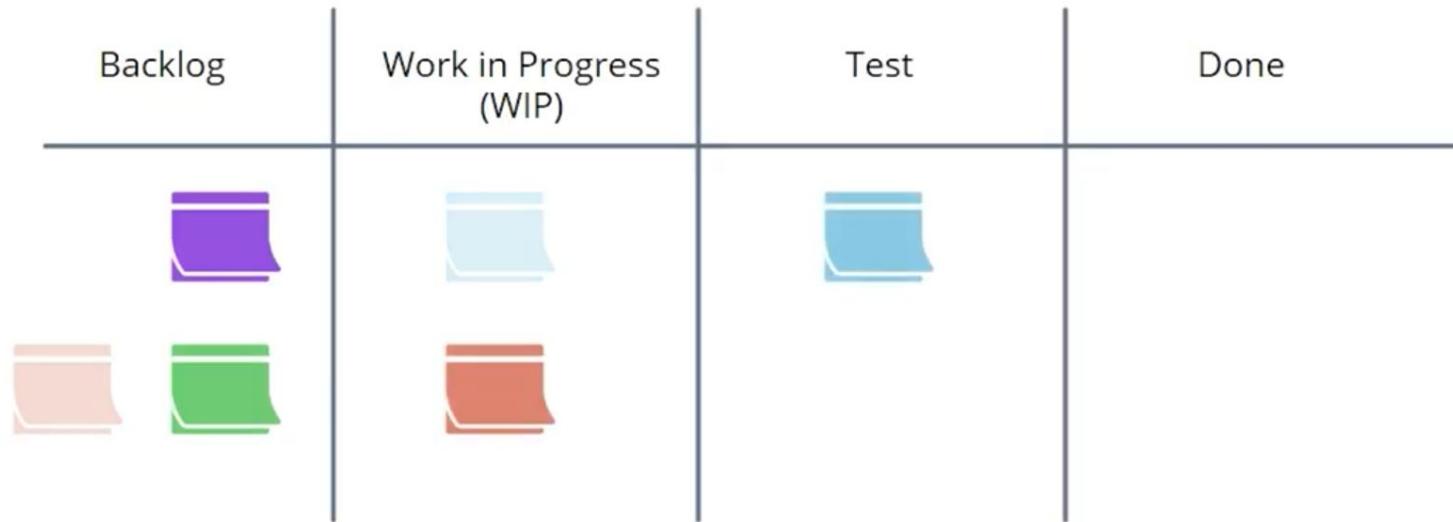
## Kanban Metrics

In Kanban it's about the 'flow' of work



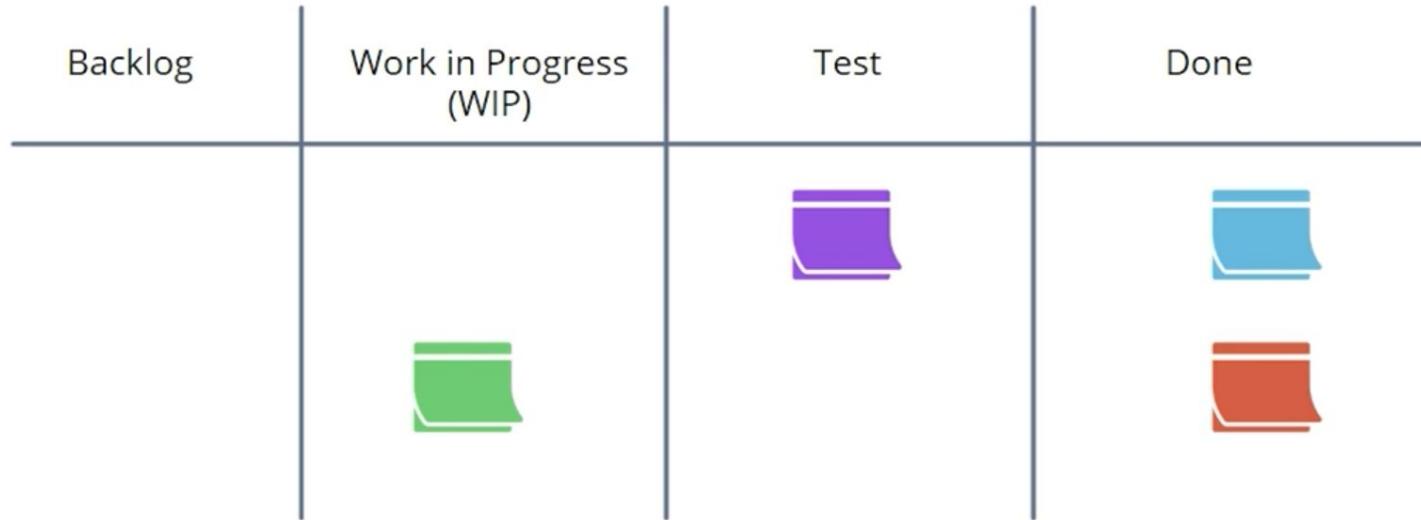
## Kanban Metrics

In Kanban it's about the 'flow' of work



## Kanban Metrics

In Kanban it's about the 'flow' of work



## Learning Objectives

---

By the End of the Lesson You Will Be Able To...

- Explain the importance of using metrics in Agile
- Differentiate between **outputs** and **outcomes**
- Calculate a **Velocity**
- Determine the **Lead Time** and **Cycle Time**
- Monitor the status of **Work in Progress (WIP)**
- Estimate when work should be completed
- Identify **escaped defects** and how to handle them appropriately



## Why Do Agile Metrics Matter?

## Information in Agile



## Focus on Transparency



We want people to dig into  
metrics that are  
**IMPORTANT**

## Agile Metrics Tell a Story

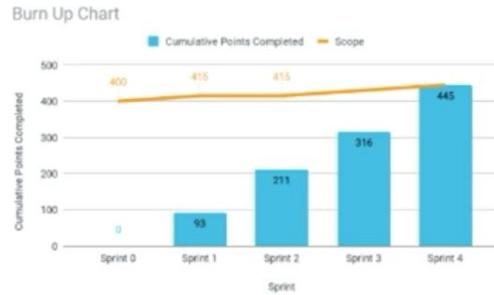
---

- How much work has been done?
- How much work is remaining?
- Are we working at peak efficiency?

Guess what- there's a metric for that!

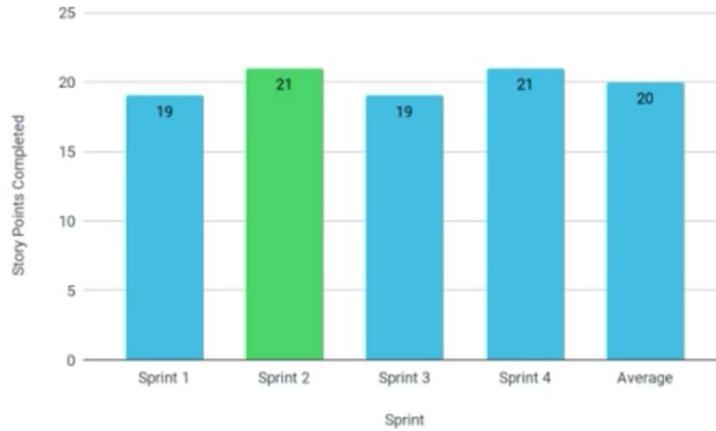


## Agile Metrics Are a Visual Representation of Progress



- Tells the story as it unfolds
- Volumes of information at a glance

## Agile Metrics Drive Team Decisions

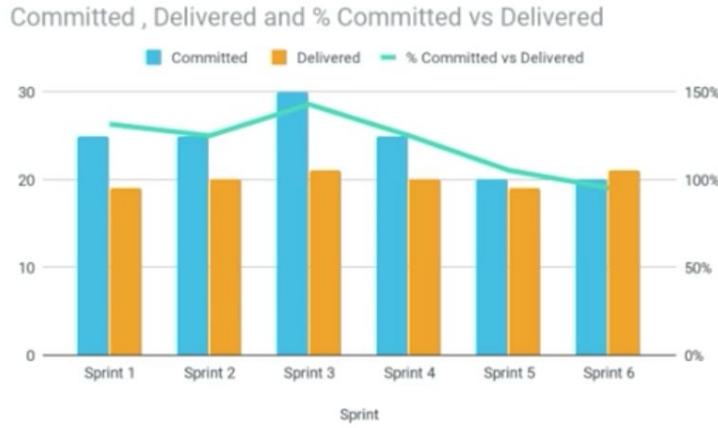


Speed up?  
Slow down?  
More training needed?

- Available immediately
- Updates on-demand and live

***Allows teams to  
find and correct  
problems earlier***

# Agile Metrics Drive Management Decisions



Overworked?  
Under delivering?  
Gold-plating the work?  
Performing at peak?

- View team progress at a glance
- Current metrics -- not data from months ago

***Helps management decide  
when the team needs a  
course correction***

## Key Points To Remember About Agile Metrics

---

- On demand
- Transparent
- Make decisions based on best info at the moment, adjust, and hit the customer targets sooner



## Developing Your Intuition About Agile Metrics

## Using Metrics

Agile metrics calculated and displayed by tools like Jira or Version One



But where do numbers come from?



## Using Metrics

Agile metrics calculated and displayed by tools like Jira or Version One



Understanding Metrics  
Is critical



## Metrics vs. Charts

---

In general,  
**Metrics are data**

In this course,  
**Metrics are Charts** that display the data

## Metrics and the Agile Manifesto

---

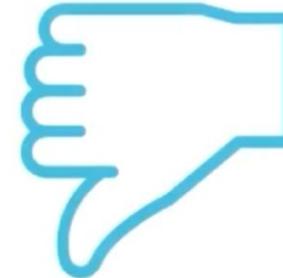
*Individuals and  
Interactions  
over process and tools*

-- *The Agile Manifesto*

- Metrics are a helpful *tool*
- The *meaning behind the data* is what is important

## The Good And Bad About Agile Metrics

- Tell a story
- Visual representation of project status
- Created automatically
- Only a snapshot
- May be misleading





# Introduction to Metrics

## Part 1

## Metrics for Traditional Projects

---

- Planned Value
- Actual costs
- Cost Variance
- Return on Investment
- Earned Value
- SLOC (Software Lines of Code)

## Traditional Metrics

---

### **Focus on *Outputs***

- Measures the team's ***effort***
- Credit for work-in-progress
- Anything produced has value -- even if it doesn't work

## Agile Metrics

---

### **Focus on *Outcomes***

- Measures the ***value*** the work is providing to the customer
- Focuses on completion -- 98% is not done!

## Example: SLOC (Software Lines Of Code)



More Output



## Example: SLOC (Software Lines Of Code)



More Output



Better Outcome

## Traditional Project Management Uses Stoplights

- Green - good
- Yellow - worrisome
- Red - bad



Jump to 00:02:22

## Traditional Project Management Uses Stoplights

- Green - good
- Yellow - worrisome
- Red - bad



## Watermelon Example



Looks green. Must be done!



Nope! Not done at all!

## Agile Metrics Are About the Journey to Create Value

---

- Burn Down Charts
- Burn Up Charts
- Velocity Charts
- Committed vs. Delivered charts
- % of Code Coverage
- Lead time & Cycle time

## What Makes Agile metrics Special?

---

- Agile Metrics track value delivered
- Prioritized by the PO
- Highest value items are delivered first!

## Scrum Metrics



- Work done in an Iteration or Sprint
- Paints a picture of progress
- Key metrics include **Burn down**, **Burn up**, **Velocity** and **Committed vs Delivered**

# Burn Down Chart

Used to track how much work is remaining



- Story points are on the y-axis

## Burn Down Chart

Used to track how much work is remaining



- Story points are on the y-axis
- Sprints are on the x-axis

# Burn Down Chart

Used to track how much work is remaining



- Story points are on the y-axis
- Sprints are on the x-axis
- Story points are reduced as work is completed

# Burn Down Chart

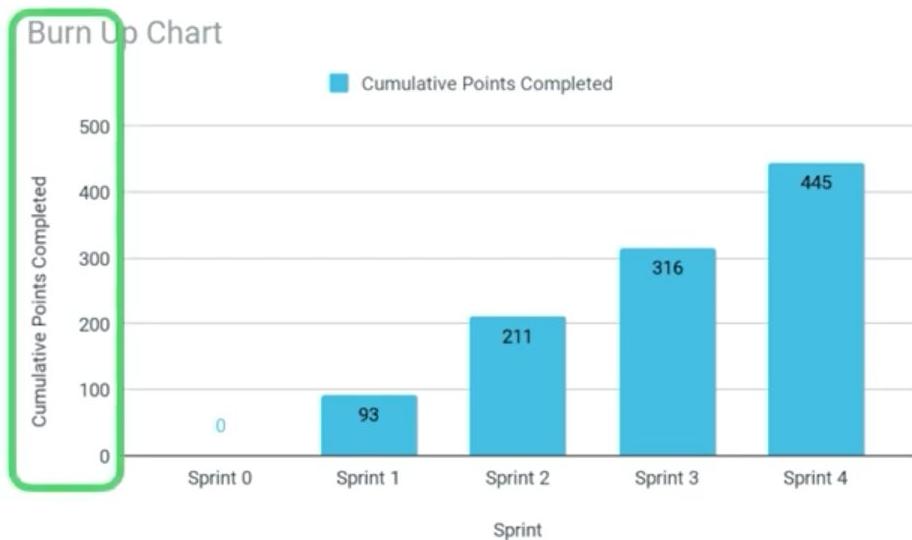
Work That Has Already Occurred is tracked



- Story points are on the y-axis
- Sprints are on the x-axis
- Story points are reduced as work is completed

# Burn Up Chart

## Completed Work



- Story points are on the y-axis
- Sprints are on the x-axis
- Story points are increased as work is completed

# Burn Up Chart

## Completed Work



- Story points are on the y-axis
- Sprints are on the x-axis
- Story points are increased as work is completed

# Burn Up Chart

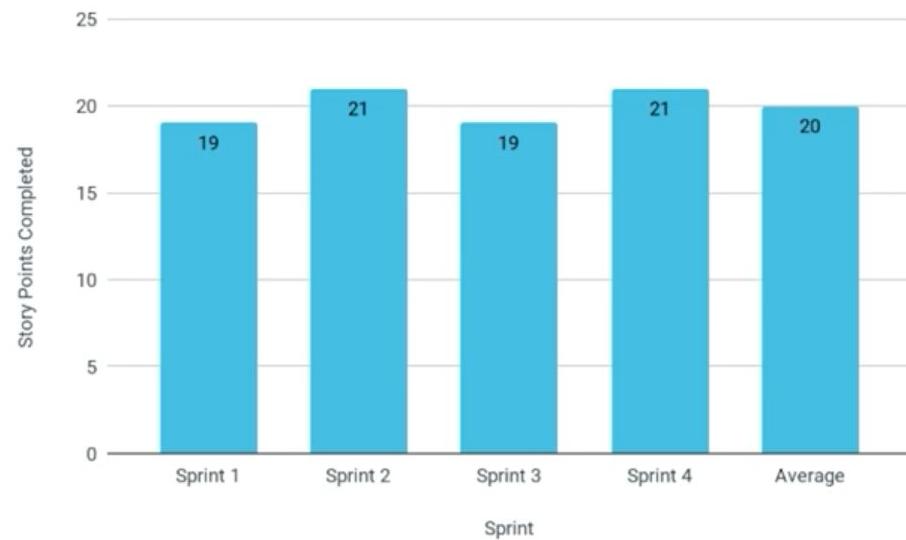
## Completed Work



- Story points are on the y-axis
- Sprints are on the x-axis
- Story points are increased as work is completed
- Scope line displays the forecast of total work required

# Velocity Chart

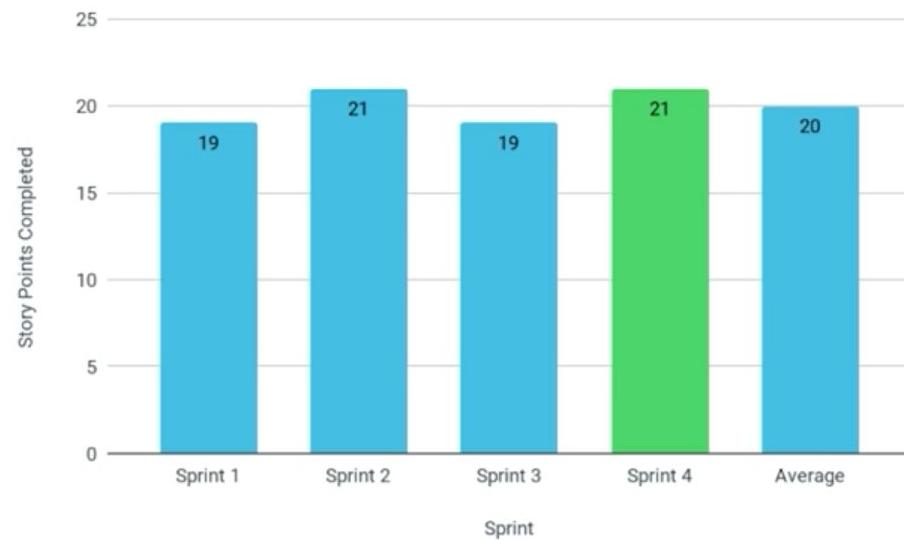
## Work Delivered Over Time



- Story points are on the y-axis
- Sprints are on the x-axis
- Completed story points for each sprint are shown

# Velocity Chart

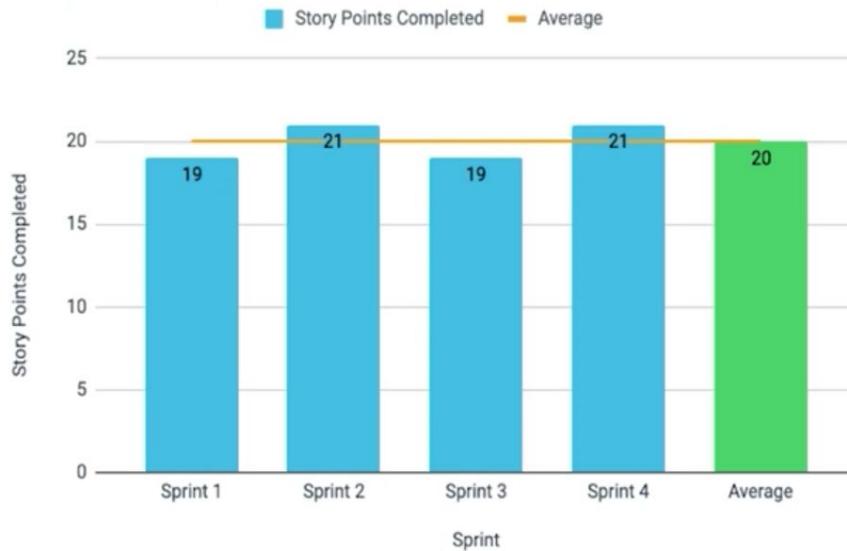
## Work Delivered Over Time



- Sprint 1 completed 19 story points of work
- Sprint 2 completed 21 story points of work
- Sprint 3 completed 19 story points of work
- Sprint 4 completed 21 story points of work

# Velocity Chart

## Work Delivered Over Time

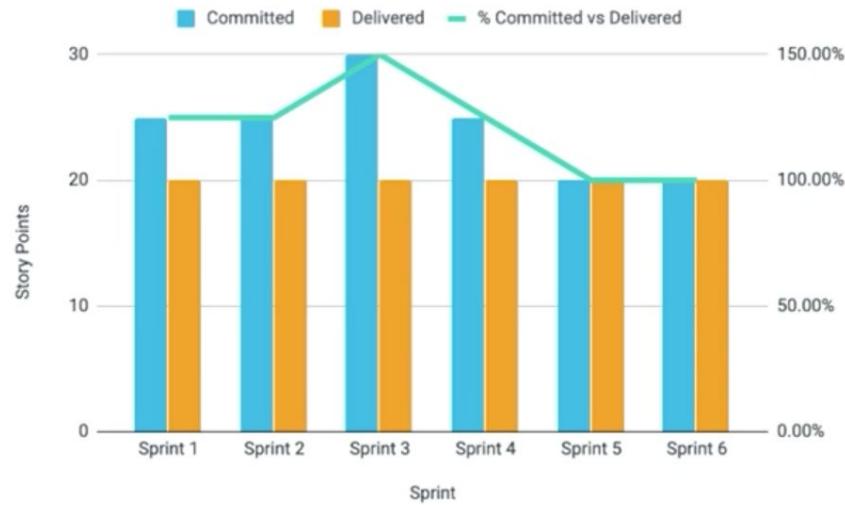


Total Number of Story Points  
Average =  $\frac{\text{Total Number of Story Points}}{\text{Sprints Completed}}$

$$\text{Average} = \frac{80}{4} = 20$$

## Committed vs Delivered

How Much Work Was Committed/How Much Work Was Delivered

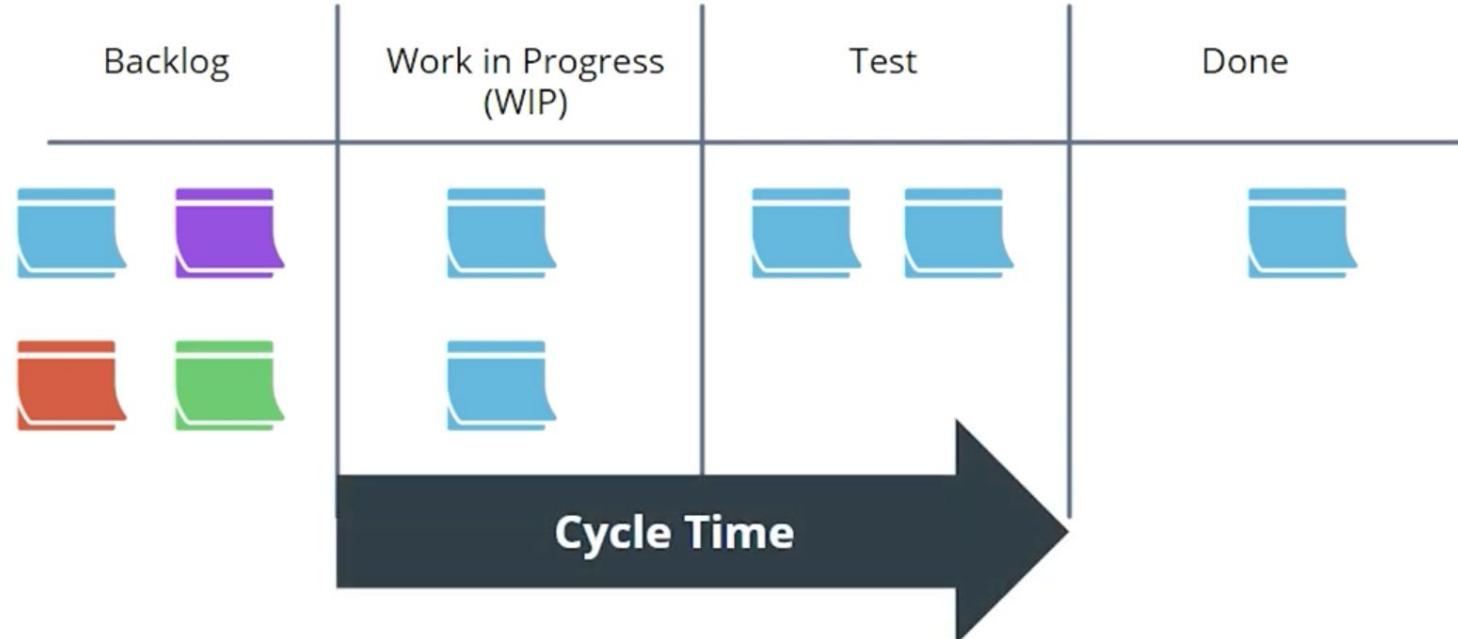


## Kanban Charts

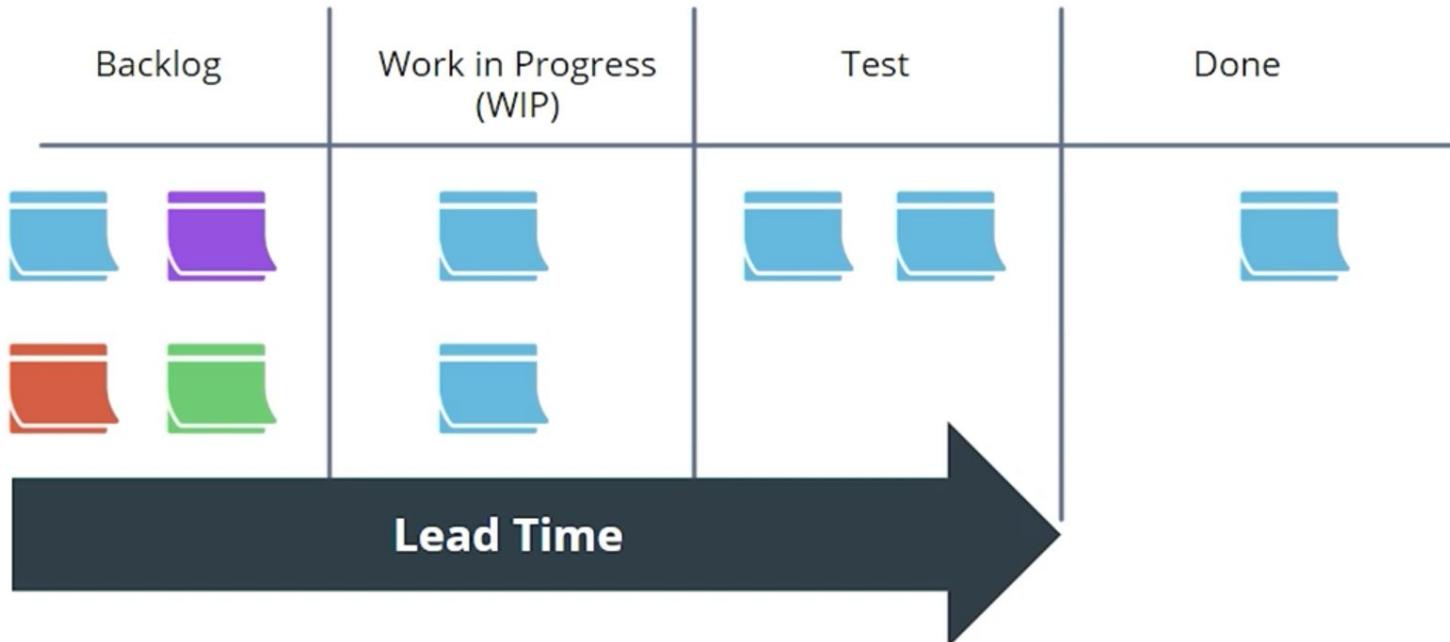


- Managing the flow
- Removing bottlenecks
- Key metrics are **Cycle Time** and **Lead Time**

## Kanban Metrics



## Kanban Metrics



## Kanban Metrics

Backlog (Avg Days)	WIP (Avg Days)	Test (Avg Days)
7	2	1

**Cycle Time** = 3 days       $2 \text{ WIP days} + 1 \text{ Test day}$

**Lead Time** = 10 days       $7 \text{ Backlog days} + 2 \text{ WIP days} + 1 \text{ Test day}$

## **Goal of Kanban is Flow!**

---

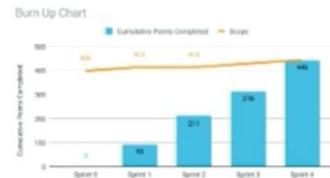
- Reduce WIP → more work is FINISHED
- Longer Lead Time = longer time to get solution to customers

# Agile Charts

## Scrum Metrics



Burn Down

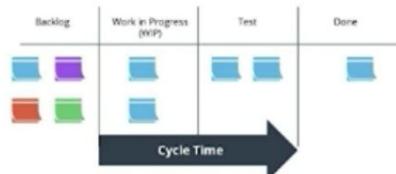


Burn Up



Velocity

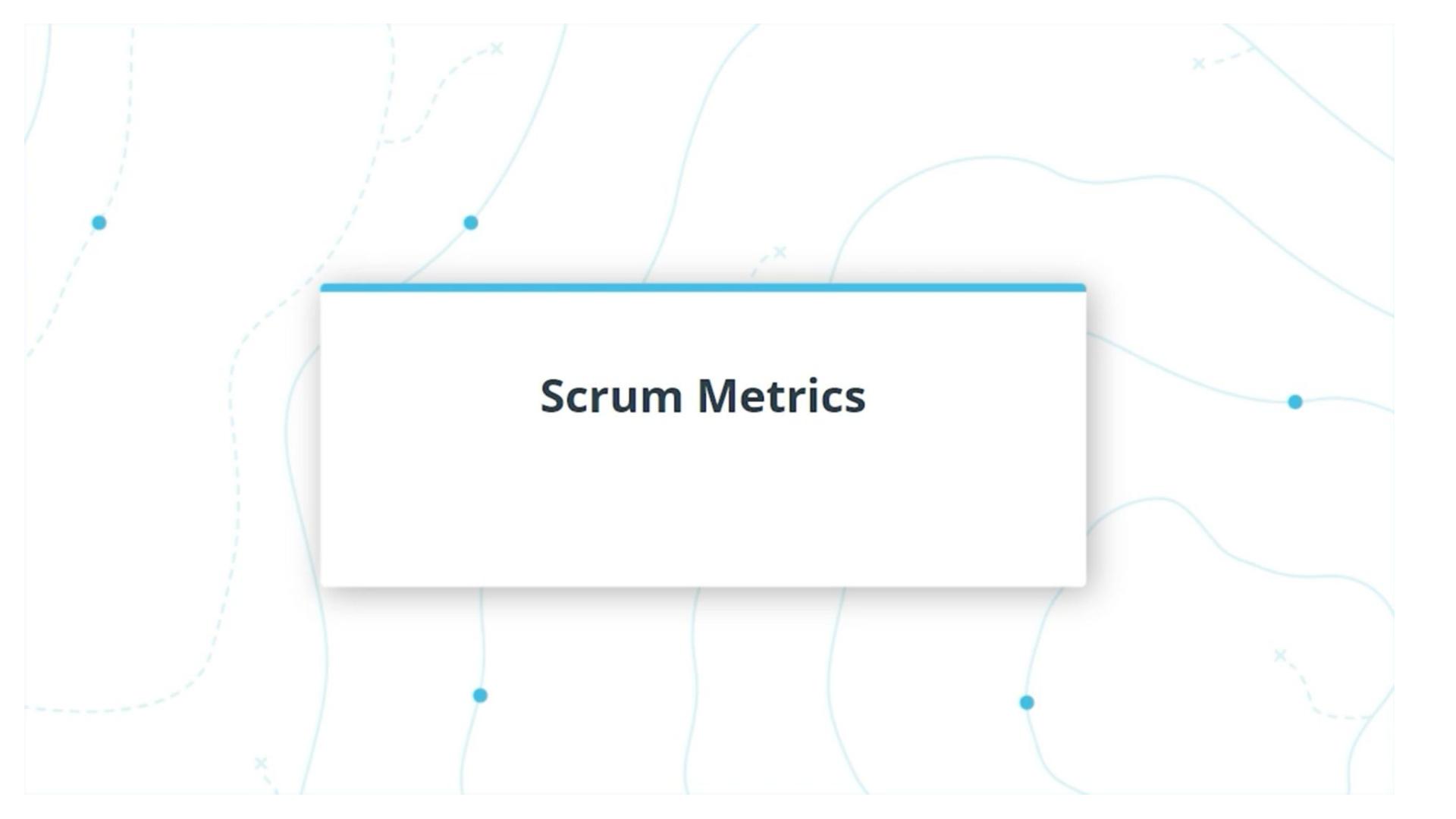
## Kanban Metrics



Cycle Time



Lead Time



# Scrum Metrics

## Creating Charts to Display Scrum Metrics



- Learn the most common Agile charts
- Where do metrics come from?
- How are the charts useful?

*Scrum Charts are a graphic display of progress*

fx

A	B	C
1 Sprint	Story Points Remaining	Story Points Completed
2 Sprint 0	100	0
3 Sprint 1	77	23
4 Sprint 2	51	26
5 Sprint 3	29	22
6 Sprint 4	0	29
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		
21		



Escaped Defects

Burn Down Chart

Velocity Chart

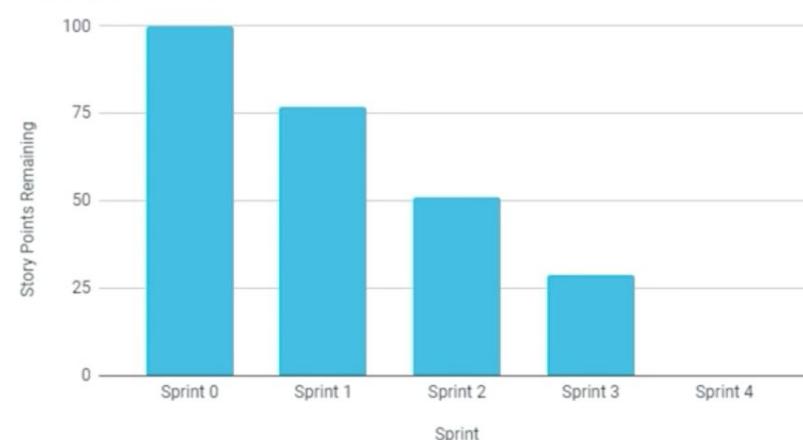
Committed vs Delivered Chart



5 2 100% \$ % .0 .00 123 Default (Ro... 10 B I A

	A	B	C
3	Sprint 1	77	23
4	Sprint 2	51	26
5	Sprint 3	29	22
6	Sprint 4	0	29
7			
8			
9			

Burn Down Chart





**Calculating Velocity**

**Work Done**

(Story Points)

$$\text{Velocity} = \frac{\text{Work Done}}{\text{Number of Sprints}}$$

## Calculating Velocity

$$\text{Velocity} = \frac{125 \text{ Story Points Completed}}{5 \text{ Sprints}} = 25$$

fx | Sprint

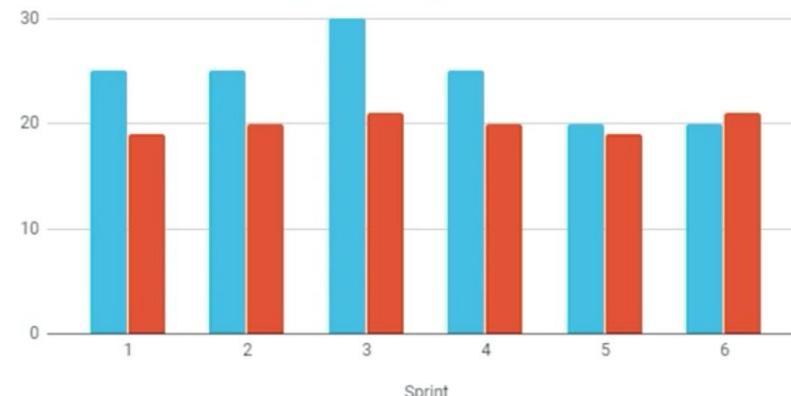
	A	B	C	D	E	F	G	H
1	Sprint	Committed	Delivered	% Committed vs Delivered				
2	1	25	19	132%				
3	2	25	20	125%				
4	3	30	21	143%				
5	4	25	20	125%				
6	5	20	19	105%				
7	6	20	21	95%				
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								
21								



fx Sprint

	A	B	C	D	E
5	4	25	20		125%
6	5	20	19		105%
7	6	20	21		95%

Committed Vs Delivered

■ Committed   ■ Delivered

## Chart editor

Setup

Customize

Chart style

Chart &amp; axis titles

## Chart title

Title text

Committed and Delivered

Title font

Theme Default...

Title font size

Auto

Title format

B I E

Title text color

Auto

&gt; Series



undo redo 100% \$ % .0 .00 123 Default (Ro... 11 A

fx

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

In Part 1, find the total number of story points completed the the sprints as well as the average number of points completed.

Then in the Part 2, use the velocity and story points to determine the number of Sprints you need to complete the all of the work. Then use this information to answer the questions in the classroom.

Part 1	
Sprint 1	27
Sprint 2	23
Sprint 3	25
Total	75
Avg	25

Part 2	
Velocity	25
Story Points	112
Sprints	4.48

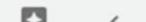
Answer	5



Agile Metrics Scrum Exercise

Agile Metrics Escaped Defects Exercise

Agile Metrics Final Exercise



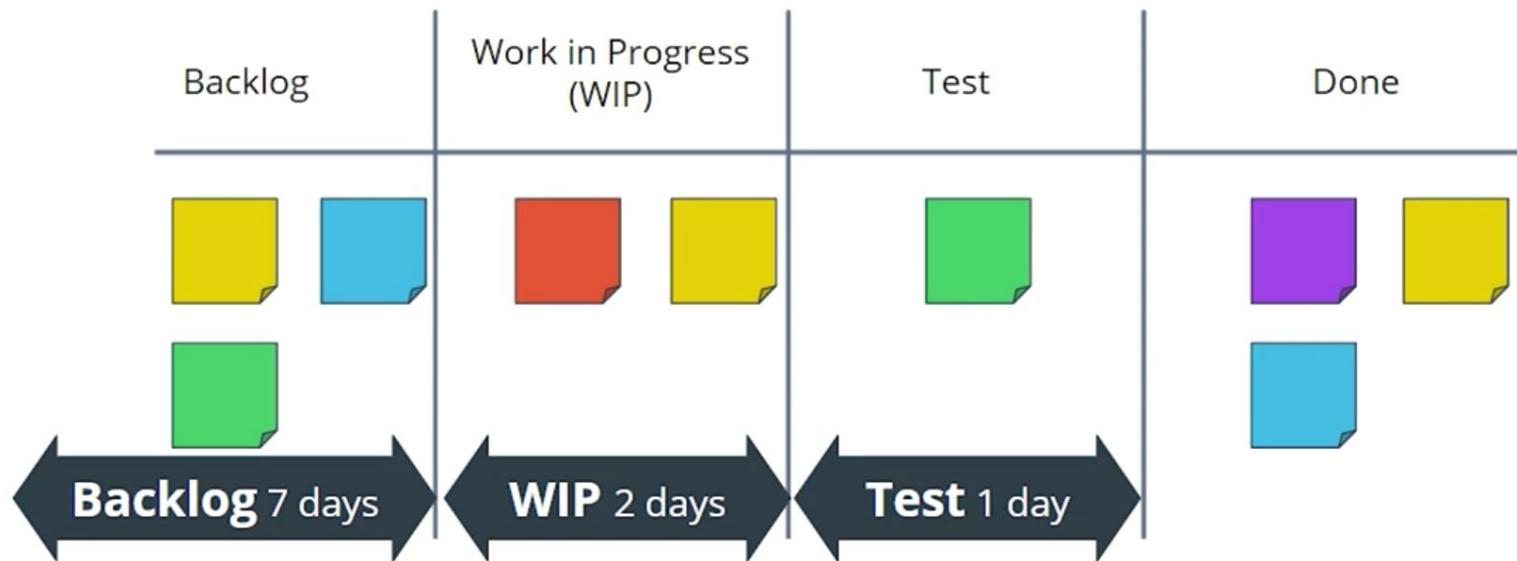
## Kanban Metrics Focus on the "Flow" of Work

---

- Time to value / time to market / time to close
- Cumulative Flow Diagram (CFD)

*Kanban Metrics focus on  
the time it takes to get work to completion*

## Calculating Lead Time



**Lead Time:** 7 days + 2 days + 1 day = 10 days

## Estimating Time To Completion

---

**Time to Completion** = Lead Time \* Number of Items in Backlog

## Example: Time To Completion



**Lead Time**  
10 days

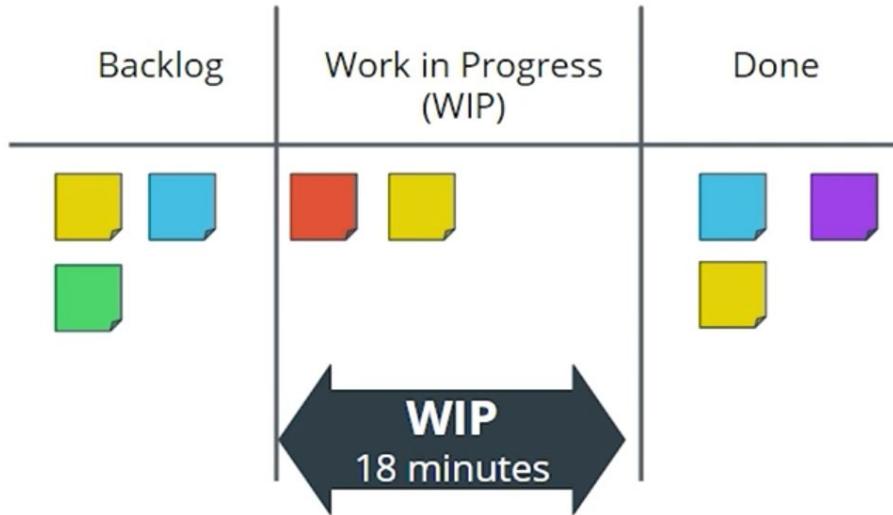


**Backlog**  
3 Items



**30 days to Complete**  
= everything in backlog

## Calculating Cycle Time for Call Centers



**Cycle Time: 18 minutes**

## Call Center Hold Times

0 Minutes



5 Minutes



10 Minutes



## Calculating Call Center Cycle Time

$$\frac{100 \text{ Trouble Tickets}}{40 \text{ Hours}} = \begin{array}{l} 2.5 \text{ Trouble tickets per hour} \\ 24 \text{ minute Cycle Time} \end{array}$$

## What are Escaped Defects?

---

*Bugs that have escaped your  
Quality Assurance and are  
released to the customer*



## Why Should I Care About Escaped Defects?

---

- More costly to fix after release
- May be dangerous to your customers



## Calculating Escaped Defects

---

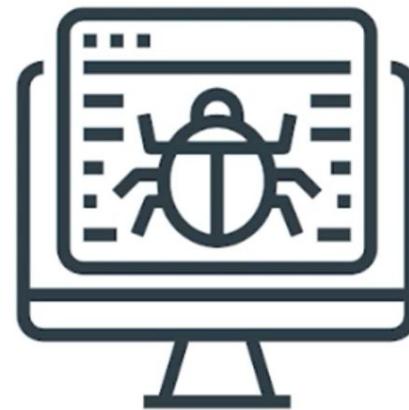
Escaped Defects =

$$\frac{\text{Bugs in } \mathbf{PRODUCTION}}{\text{Total Bugs}}$$

## What Are Bugs?

---

*Bugs are problems in the software - something we didn't expect to happen or an unintended consequence*



## Where Do We Find Bugs?



Development

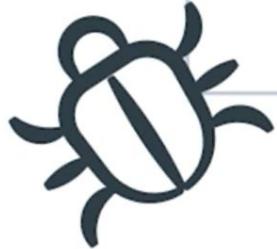


Staging



Production

## Tracking Bugs: Example



Development	Staging	Production	Total
55	10	1	66

## Tracking Bugs: Example

$$\text{Escaped Defects} = \frac{1 \text{ Bug found in Production}}{66 \text{ Total Bugs}} = 1.51\%$$

## Goal: Reduce Bugs Found In Production

---

More bugs are caught BEFORE  
the product is released to  
customers



## Escaped Defects

Sample bug locations for a larger product

	<b># of Bugs</b>	<b>% of Bugs</b>
Development	640	77.48%
Staging	140	16.95%
Production	46	5.57%
Total	826	100%

## Tracking Bugs



Tracking **Escaped Defects** is important



Tracking **all bugs** may not be cost effective

## Tracking Bugs



10 minutes



1 minute

## Changing How You Track Escaped Defects

New Project

	# of Bugs	% of Bugs
Development	250	59%
Staging	123	29%
Production	50	12%
Total	423	100%

Mature Project

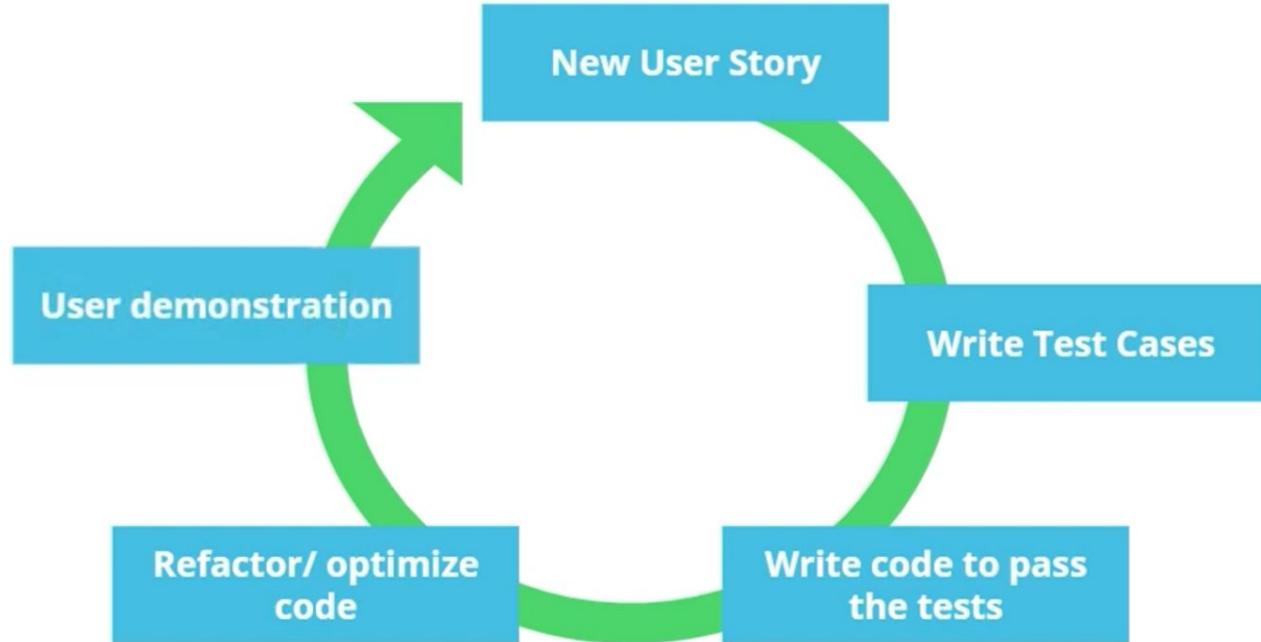
	# of Bugs	% of Bugs
Development	993	99.0%
Staging	9	0.9%
Production	1	0.1%
Total	1089	100%

## Strategies To Reduce Escaped Defects

- Track bugs by environment
- Establish a "Bug Bounty" to reward team members for finding bugs
- Increase the number of testers
- Implement Code Standards
- Enforce standards with Code Reviews
- Introduce Test Driven Development



## Test Driven Development (TDD)



## TDD & Code Coverage

---

Code Coverage = 
$$\frac{\text{Lines of code}}{\text{Automated tests for each line of code}}$$

## TDD & Code Coverage

---

$$\text{Code Coverage} = \frac{42}{42} = 100\%$$

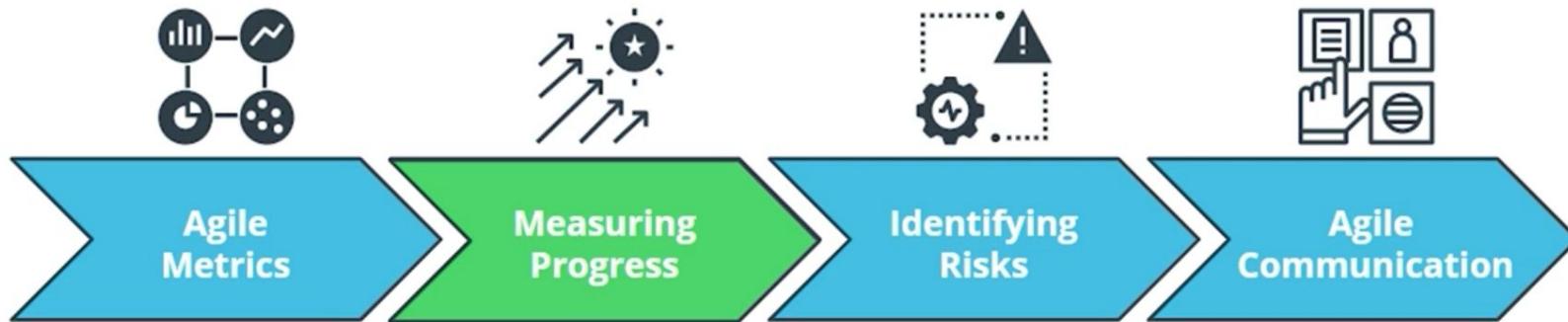
## Why Use TDD?

---

- Forces discipline
- Ensures code is tested
- Automates testing
- Makes regression testing easier

## Up Next

---



- Continuous Improvement
- Retrospectives
- Information Radiators

# Course Overview



Agile Thinking



Sustainable  
Development



Patterns and  
antipatterns



Retrospectives

## Learning Objectives

---

By the End of the Lesson You Will Be Able To...

- Identify the different parts of the continuous improvement process that lead to a sustainable development level
- Differentiate between patterns and antipatterns of the continuous improvement process
- Effectively apply Agile thinking to reach sustainable development
- Run a retrospective to effectively identify what went well, what didn't go well and what can be improved on

## Learning Objectives

---

By the End of the Lesson You Will Be Able To...

- Identify the difference between lessons captured and lessons learned
- Effectively prioritize the next steps to improve on the lessons learned and add it to the backlog
- Identify Information Radiators (IR) in the real world
- Use the appropriate chart type to build an IR to communicate a specific metric
- Create a BVIR to effectively communicate project status

## Measuring Progress and Impact

---

- Taking cues from your team
- Knowing where you are
- Knowing where you want to go
- Making changes to ensure you get there

## Measuring Progress and Impact: Tricks of the Trade

### Conduct Agile Assessments



- How empowered is the team?
- How often does the team meet users?
- How often is work passed from one team to another?

## Agile Coaching - Bad Example

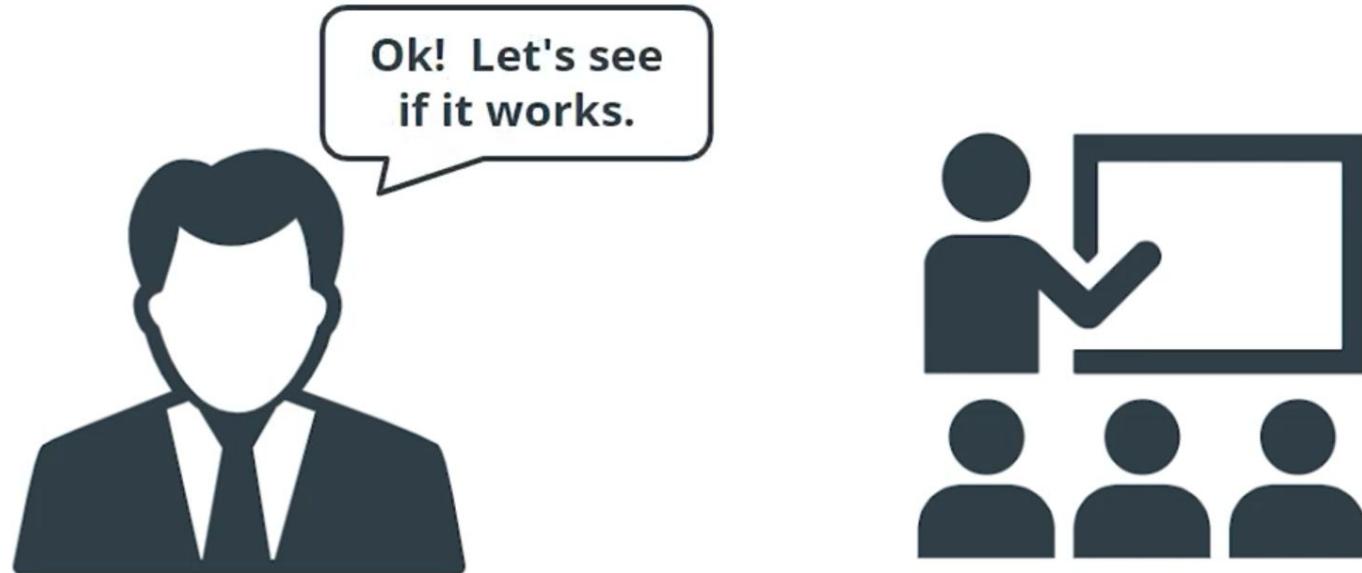


## Agile Coaching - Bad Example

- Dashboards show real-time information
- Report would take too much time and money



## Agile Coaching - Bad Example



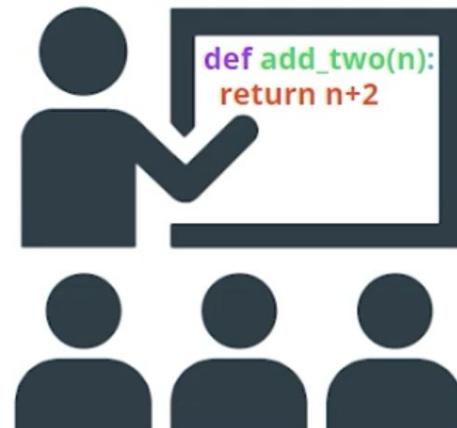
## Agile Coaching - Not So Bad Example!



CIO is Open to  
New Ideas

Ok! Let's see  
if it works.

Agile is About  
Showing the Code



```
def add_two(n):  
    return n+2
```

# Measuring Progress and Impact

## Points to Remember

- Use surveys to gather information
- Review metrics for trends
- Listen to what people say vs. what they actually do
- Get involved with the teams and get to know how they work
- Follow the Agile Manifesto

