Being Agile

Course Content

Level 1

- ✓ **Agile**
  - ✓ Agile Introduction
    - ✓ What is Agile
    - ✓ Agile Mind-set
    - ✓ Challenges in Traditional methodologies
    - ✓ Agile Myths and Facts
    - ✓ Numbers on Agile[Stats]
  - ✓ Agile Manifesto
  - ✓ Agile Principles
  - ✓ Agile Umbrella
    - ✓ XP
    - ✓ Lean
    - ✓ Kanban
    - ✓ Scrum
- ✓ **Scrum**
  - ✓ Introduction to Scrum
  - ✓ Definition of Scrum
  - ✓ Uses of Scrum
- ✓ **Scrum Framework**
  - ✓ Big Picture
  - ✓ Iterative and Incremental
  - ✓ Inspect and Adapt
- ✓ **Scrum Theory**
  - ✓ Empiricism
  - ✓ Pillars of Scrum
  - ✓ Scrum Values
- ✓ **Scrum Roles**
  - ✓ Product Owner
  - ✓ Development Team
  - ✓ Scrum Master
- ✓ **Scrum Events**
  - ✓ Sprint
  - ✓ Sprint Planning
    - ✓ What
      - ✓ Estimations
    - ✓ How
      - ✓ Task break down
  - ✓ Daily Scrum
  - ✓ Sprint Review
  - ✓ Sprint Retrospective
- ✓ **Scrum Artifacts**
  - ✓ Product Backlog
  - ✓ Sprint Backlog
  - ✓ Product Increment
- ✓ **Agreements**
  - ✓ Definition of Ready
  - ✓ Definition of Done
- ✓ **Metrics**
  - ✓ Burn-down
  - ✓ Burn-up
  - ✓ Cumulative Flow Diagram
  - ✓ Lead time
  - ✓ Cycle Time
- ✓ **Estimations**
  - ✓ Relative Sizing
  - ✓ Modified Fibonacci Series
  - ✓ T Shirt Sizing
  - ✓ Planning Poker
  - ✓ CUE Factor
- ✓ **Product Backlog Refinement**
  - ✓ Prioritization Techniques
  - ✓ DEEP
  - ✓ User Stories
    - ✓ INVEST
  - ✓ SPIKES
  - ✓ EPICS/ Features
- ✓ **Scrum of Scrums**
- ✓ **Q&A**
- ✓ **Mock Test**

Activities

- Agile Manifesto
- Inspect & Adapt
- Scrum Values
- Simulation of Scrum

Level 2

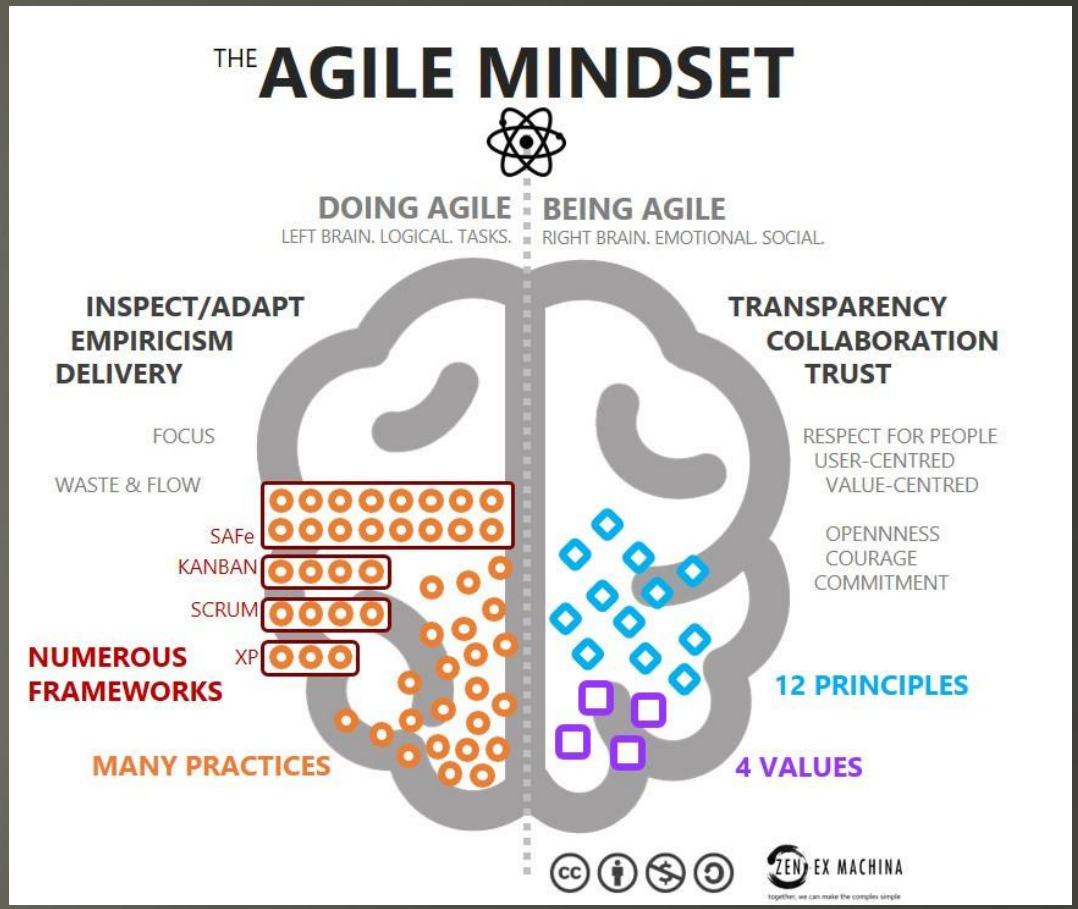
- Kanban – 2 hours
- Scrumban- 1 Hour
- SAFe Overview- 2 hours
- Agile Transformation Challenges- 2 hours
- Anti Agile Patterns- 6 hours
- Scrum Master Challenges- 2 hours
- Effective Facilitation Techniques- 1 hours
- JIRA or any other Tools- 1 hour
- Resume Prep- Offline
- Interview Questions- PDF's to be Sent
- Mock interviews- 2 hours per person
- Interview Support and Retrospectives – On Need basis

- ✓ **LEAN**
  - ✓ Introduction
  - ✓ 7 Lean Principles
  - ✓ 7 TPS Wastes
  - ✓ 7 Wastes of Software Development
  - ✓ Lean Leadership
- ✓ **XP**
  - ✓ Introduction
  - ✓ 5 Values
  - ✓ 12 Practices
  - ✓ Roles
- ✓ **KANBAN**
  - ✓ Introduction
  - ✓ 4 Basic Principles
  - ✓ 6 Core Practices
  - ✓ Metrics
- ✓ **SCRUMBAN**

What is Agile


No manager
Incremental
Self Organizing
To Deliver Fast
Iterative
Speed to Market
Methodology
No Documentation
CHANGE YOUR MIND
BY CHANGING YOUR PERCEPTION

Agile is a Mind-set



The classic problem

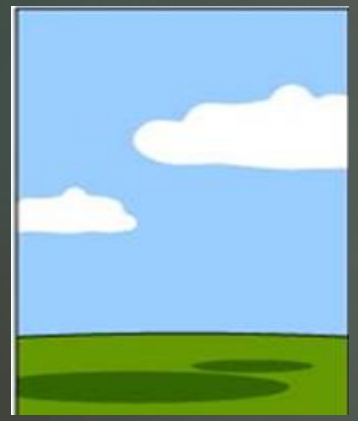
How the customer explained it

How the Project Leader understood it

How the analyst designed it

How the programmer wrote it

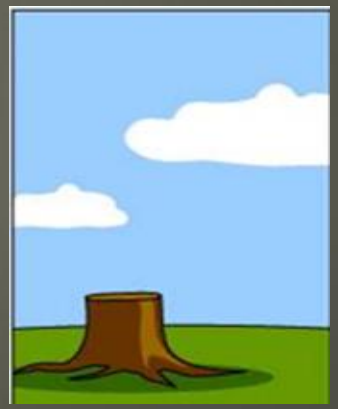
How the business consultant described it



How the project was documented

What operations installed

How the customer was billed



How it was supported

What the customer really needed

It’s not waterfall

The Traditional Way

TIME
JAN
FEB
MAR
APR
MAY
JUN
JUL
AUG
SEP
OCT
NOV
DEC

Plan

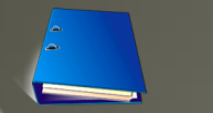
Analysis

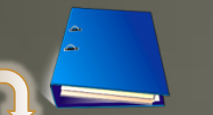
Design

Code

Test

Deploy





The Agile Way

INSPECT & ADAPT

WHAT IS THE DIFFERENCE?

~~Large group spending a long time building a huge thing~~

Small team spending a little time building a small thing

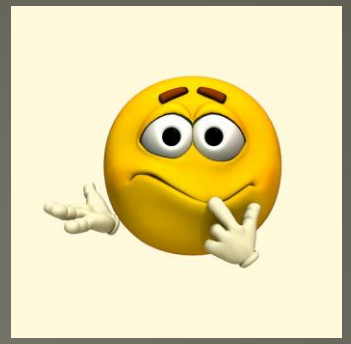
... but integrating regularly to see the whole

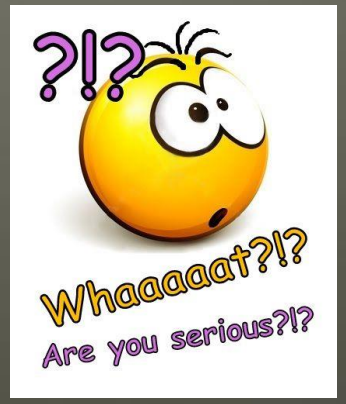
Challenges in Waterfall Model

Leadership	• Project manager Controls the Project and Team has Zero Decision making
Feedback	• Received late in the Project Life Cycle, Adds lot of Rework
Risk	• High Risks pop up due to communication Gap
Poor Visibility	• It is difficult to see the exact status of project at any point in-time
Poor Quality	• Generally Quality takes a hit in Traditional Model due Time Constraints
Larger Teams	• Teams are huge in size and there is more tendency to work in silos and deviate
Late Value Delivery	• Customer Cannot see the Product until the end of Project
Planning	• Big Upfront Planning
Changes	• Changes are not accepted Easily
Customer Involvement	• Customer is only involved in the beginning and again during the UAT
Cost of Changes	• Very High

Agile Myths

- Agile solves every problem in this world
- Agile means no architecture and no documentation
- Agile just an iterative and incremental model
- Agile works for only small projects
- Agile Teams get more stress and early burn-out
- Agile requires very experienced people
- Agile do not need project manager
- Agile does not support CMMI model
- Agile model will not fit with scaling
- Agile do not have risk management





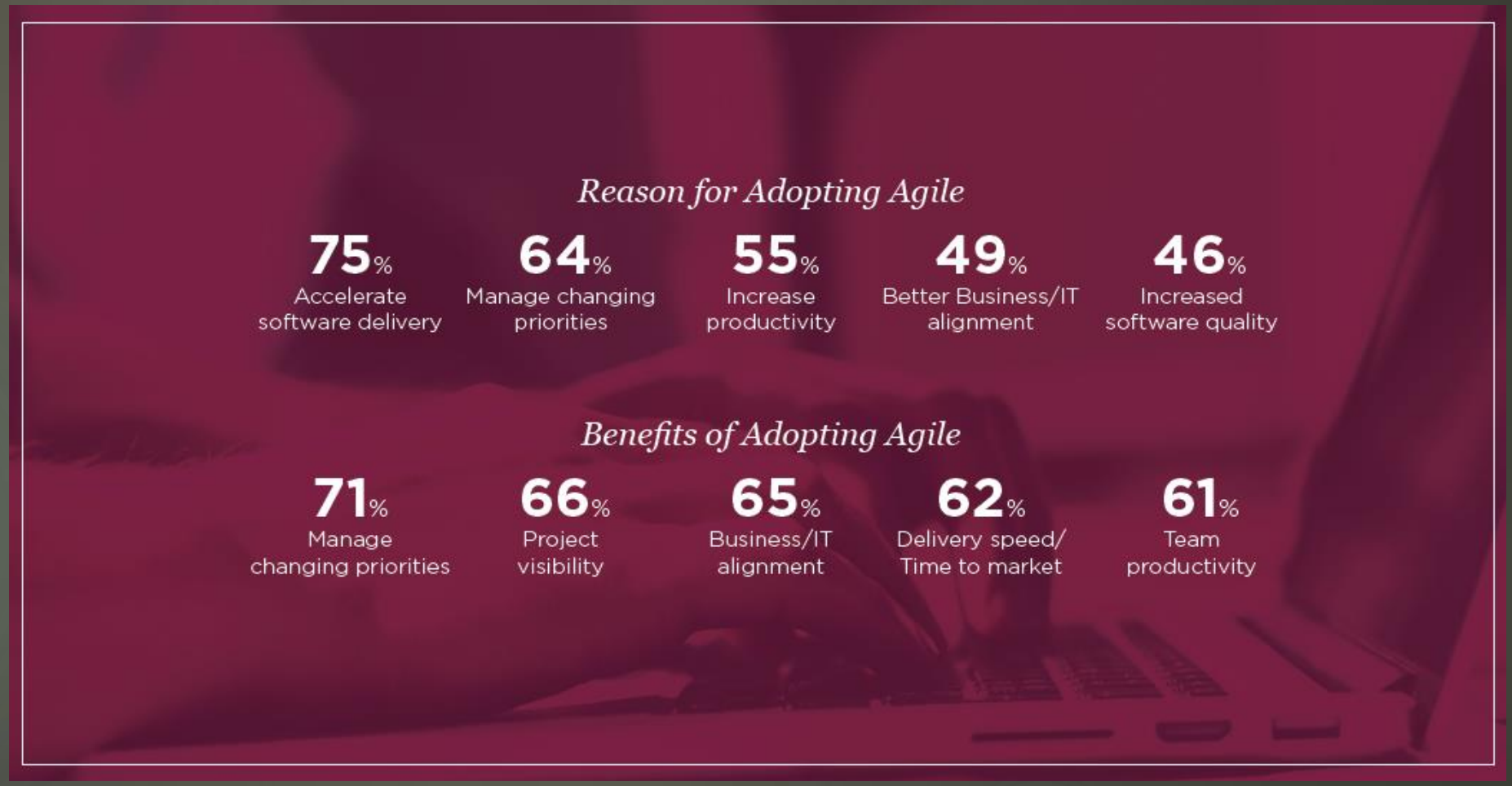
Reasons

- Not understanding the concepts
- Wrong definitions of the concepts
- Bad use of concepts
- No experience in distributed projects
- No experience on big sized projects
- Blaming the process
- Not being agile

Why Agile

Q:Why Agile

Answer: Look at The Numbers. They don't lie



Why Agile

Q:Why Agile

Answer: Look at The Numbers. They don't lie

COMPANY EXPERIENCE AND ADOPTION

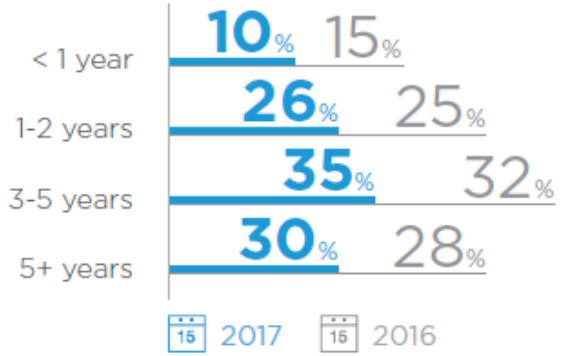
Company Experience

HOW MANY?

97%
The percentage of respondents' organizations that practice agile development methods:

&

HOW LONG?
The length of time respondents' organizations have been practicing agile development methods:



Percentage of Teams Using Agile

52% of respondents stated that more than half of teams in their organizations are using agile practices.

25% All of our teams are agile

27% More than ½ of our teams are agile

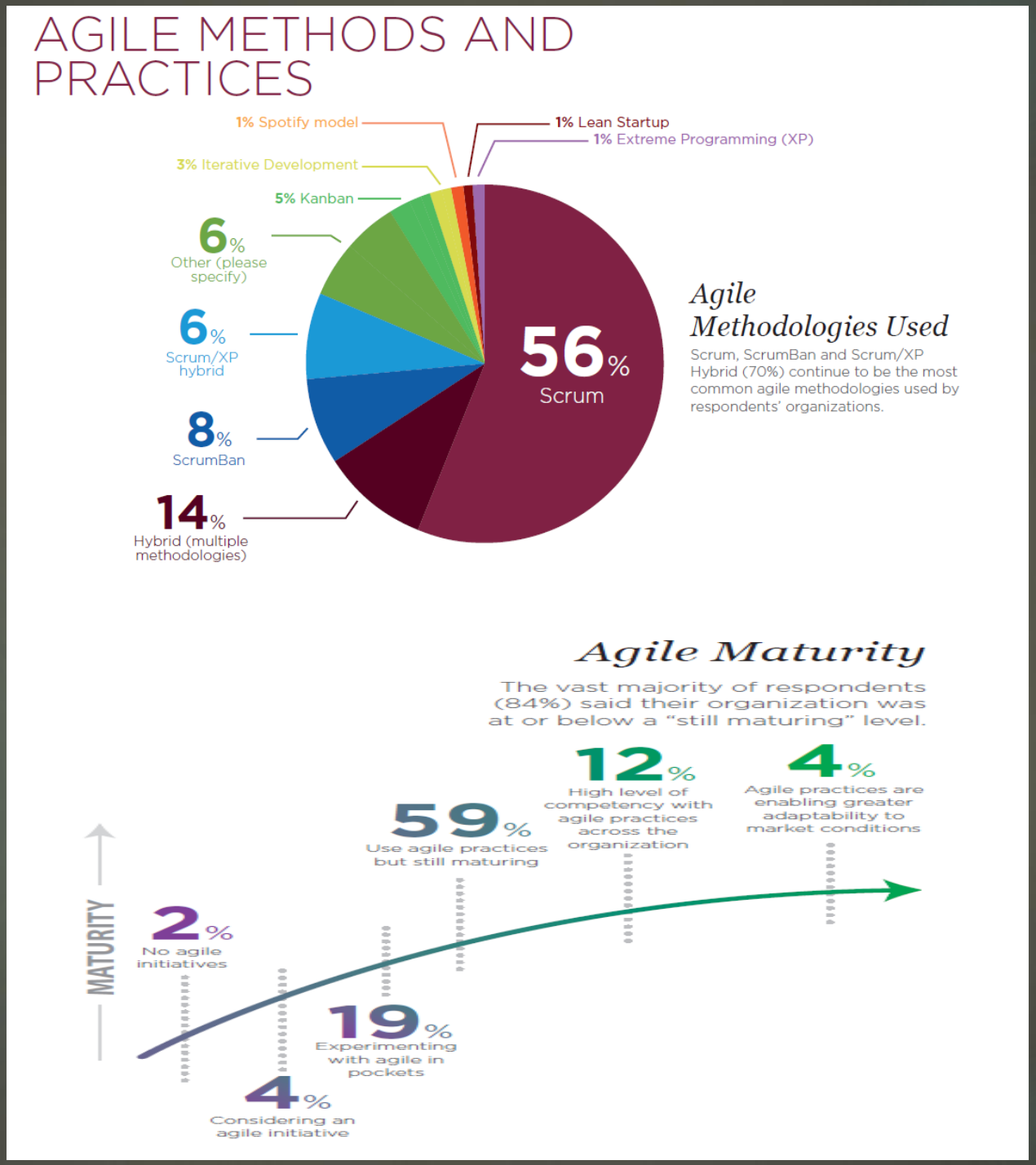
46% Less than ½ of our teams are agile

2% None of our teams are agile

Q: Why Scrum

Answer: Majority Feels its Working for them

Q:Why Scrum Master Answer: 84% of the Teams Need you



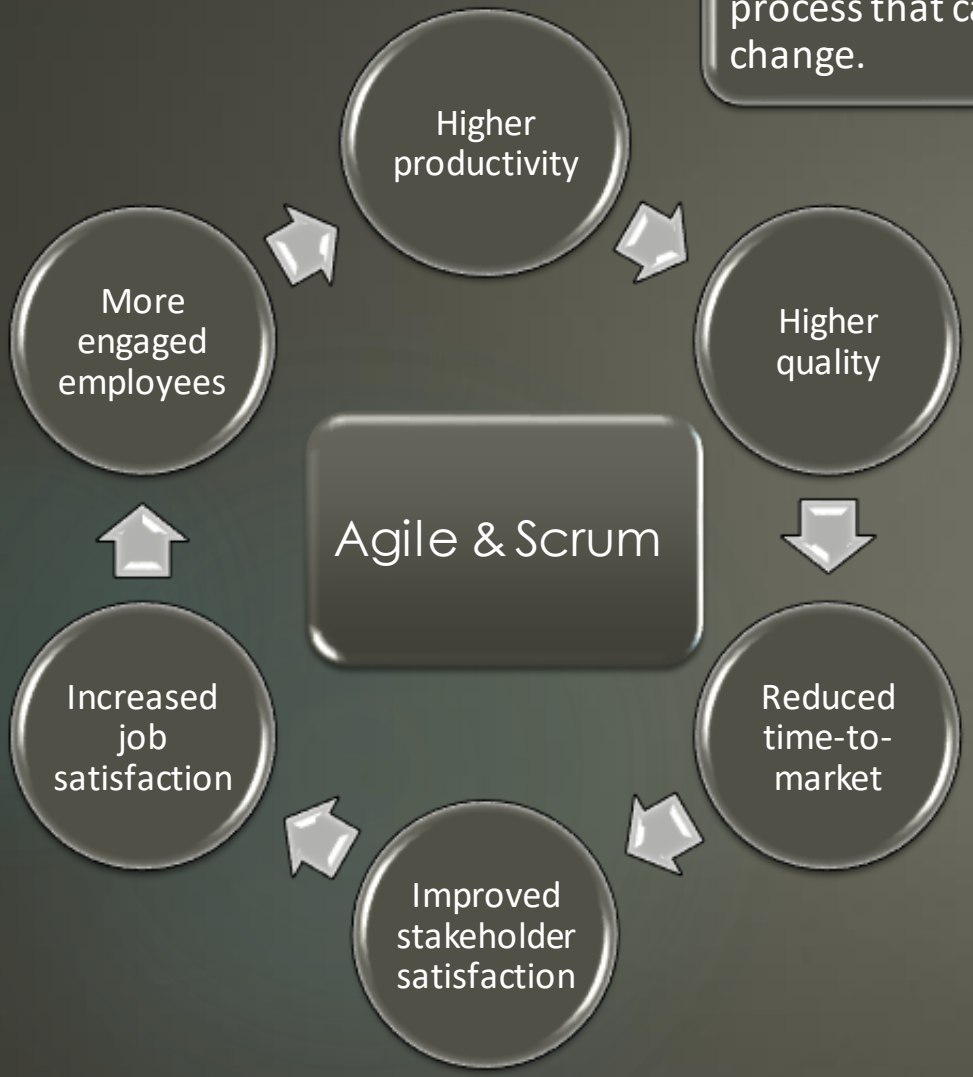
Source:

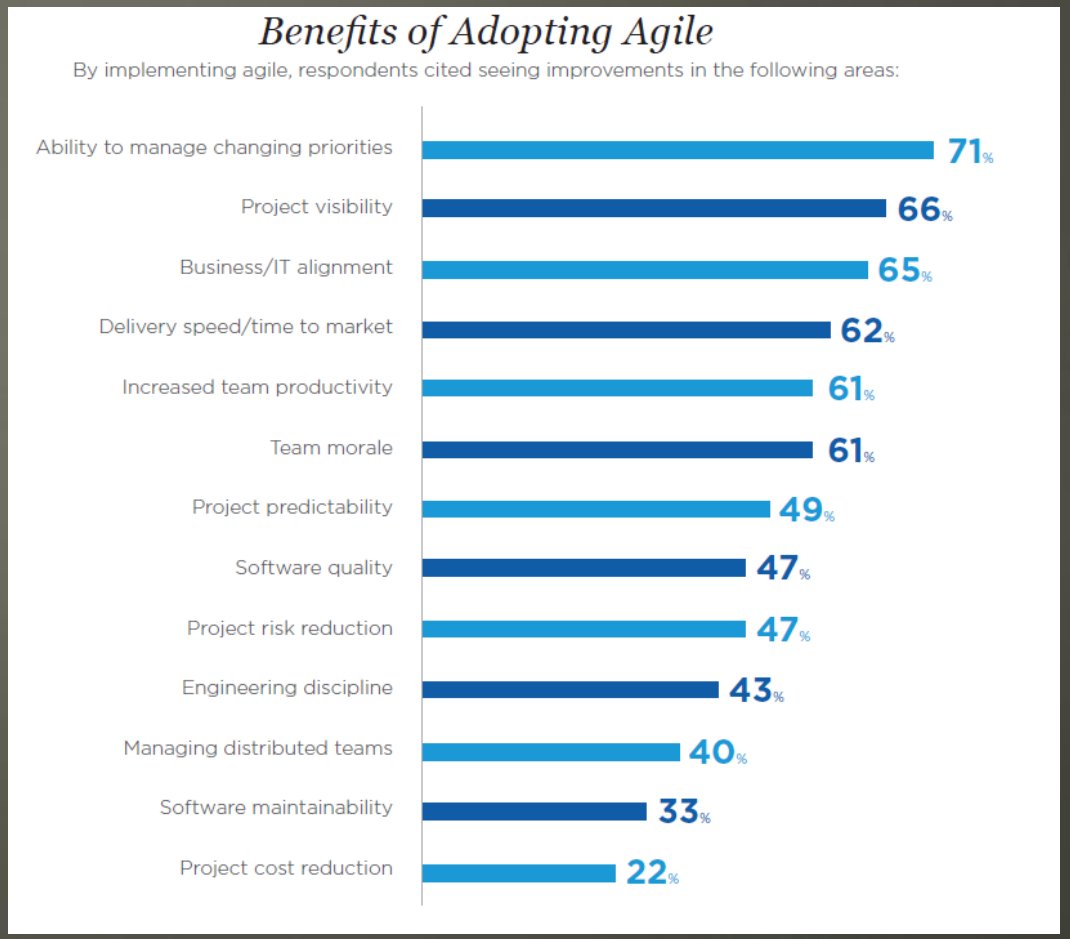
Adobe Acrobat Document

Benefits

Today's “fast enough” will likely not be fast enough tomorrow. In order to remain competitive, companies developing software need an agile process that can help them keep up with the accelerating rate of change.

Agile and Scrum helps teams develop software quicker, and at lower costs, giving them a competitive advantage in a fast-paced market.





Agile Manifesto

History of Agile Manifesto
http://agilemanifesto.org/history.html

Comprehensive Documentation

Working Software

Customer Collaboration

Individuals Interactions

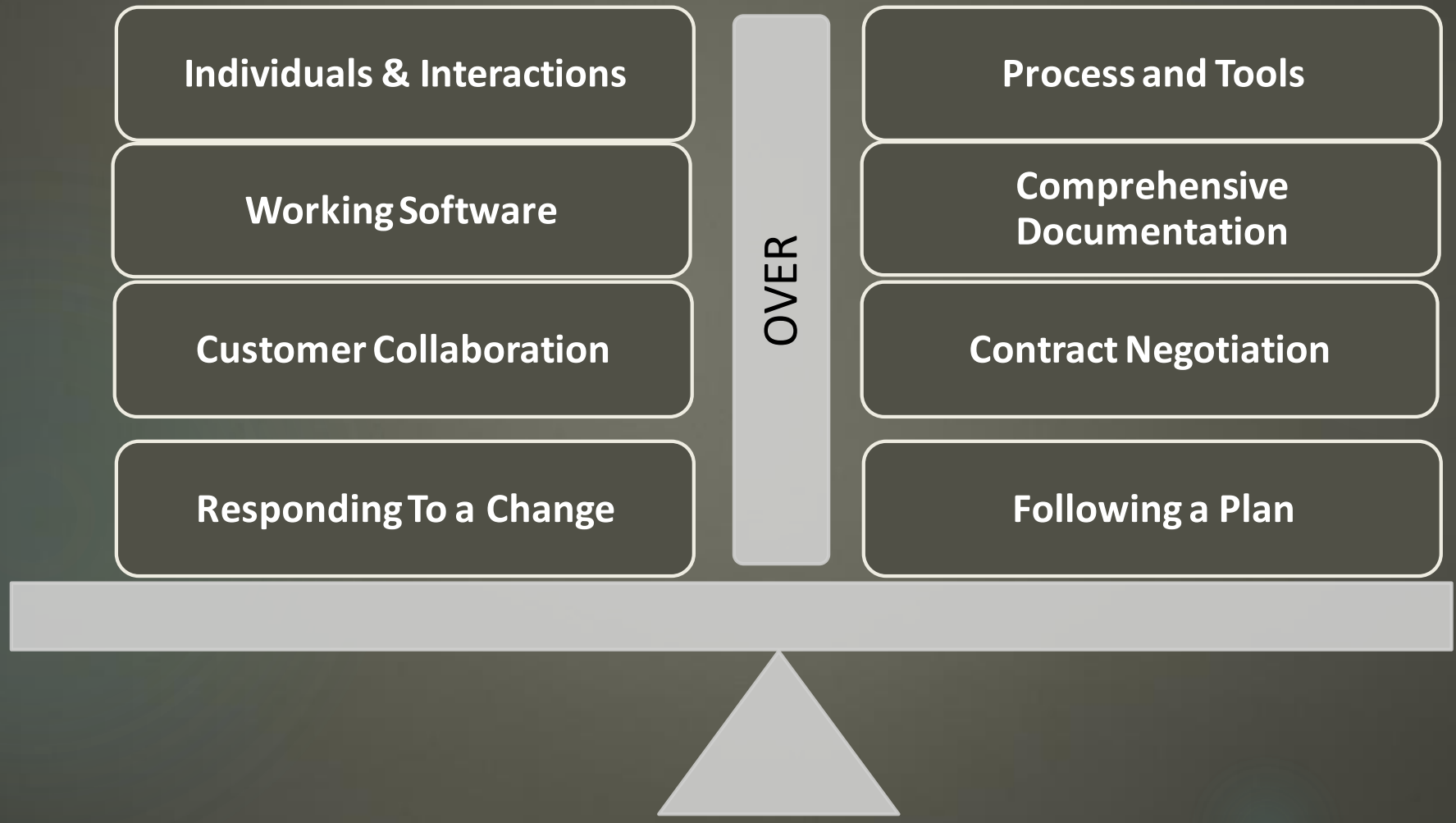
Contract Negotiation

Following a Plan

Responding to Change

Process and Tools

Agile Manifesto



12 Principles of Agile

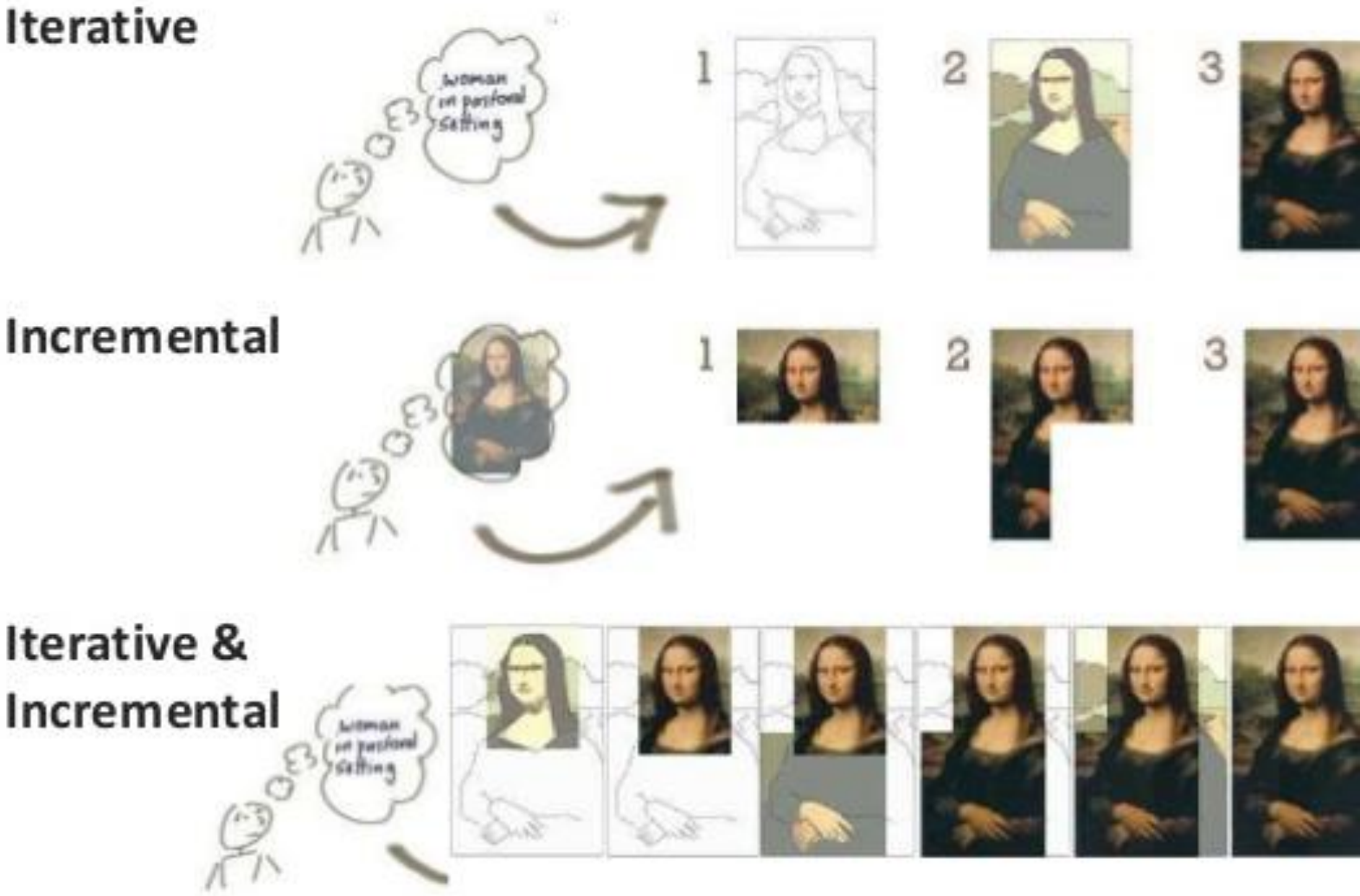
1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software [CUSTOMER SATISFACTION]
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage. [WELCOMING CHANGE]
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale. [DELIVER FREQUENTLY]
4. Business people and developers must work together daily throughout the project. [COMMUNICATION IS THE KEY]
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done. [ENIVRONMENT AND TRUST]
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation. [FACE TO FACE COMMUNICATION]
7. Working software is the primary measure of progress. [SOFTWARE AS A MEASURE OF PROGRESS]
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely. [SUSTAINABLE DEVELOPMENT]
9. Continuous attention to technical excellence and good design enhances agility. [ATTENTION TO DETAILS]
10. Simplicity--the art of maximizing the amount of work not done--is essential. [SIMPLICITY]
11. The best architectures, requirements, and designs emerge from self-organizing teams. [SELF-ORGANIZING TEAMS]
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly. [INSPECT AND ADAPT]

Agile Methods and Practices

Term	Definition
Agile Manifesto	A public Declaration of the philosophy and Principles of Agile Software Development, created in Feb 2001 in Snowbird, Utah
Agile Methodologies	Frameworks and Process whose practices support the Agile Manifesto Principles That includes SCRUM, XP,CRYSTAL,DSDM, FDD, KANBAN
Agile Practices	Activities that apply Agile Principles
Agile Principles	Fundamental truth and Shared Values that drive behaviour in Agile Methodologies
Iterative and Incremental	The Approach of Implementing a work product in successive pieces(INCREMENTS) While also gradually refining the Product through Target Improvements(ITERATIONS)

Iterative and Incremental Delivery

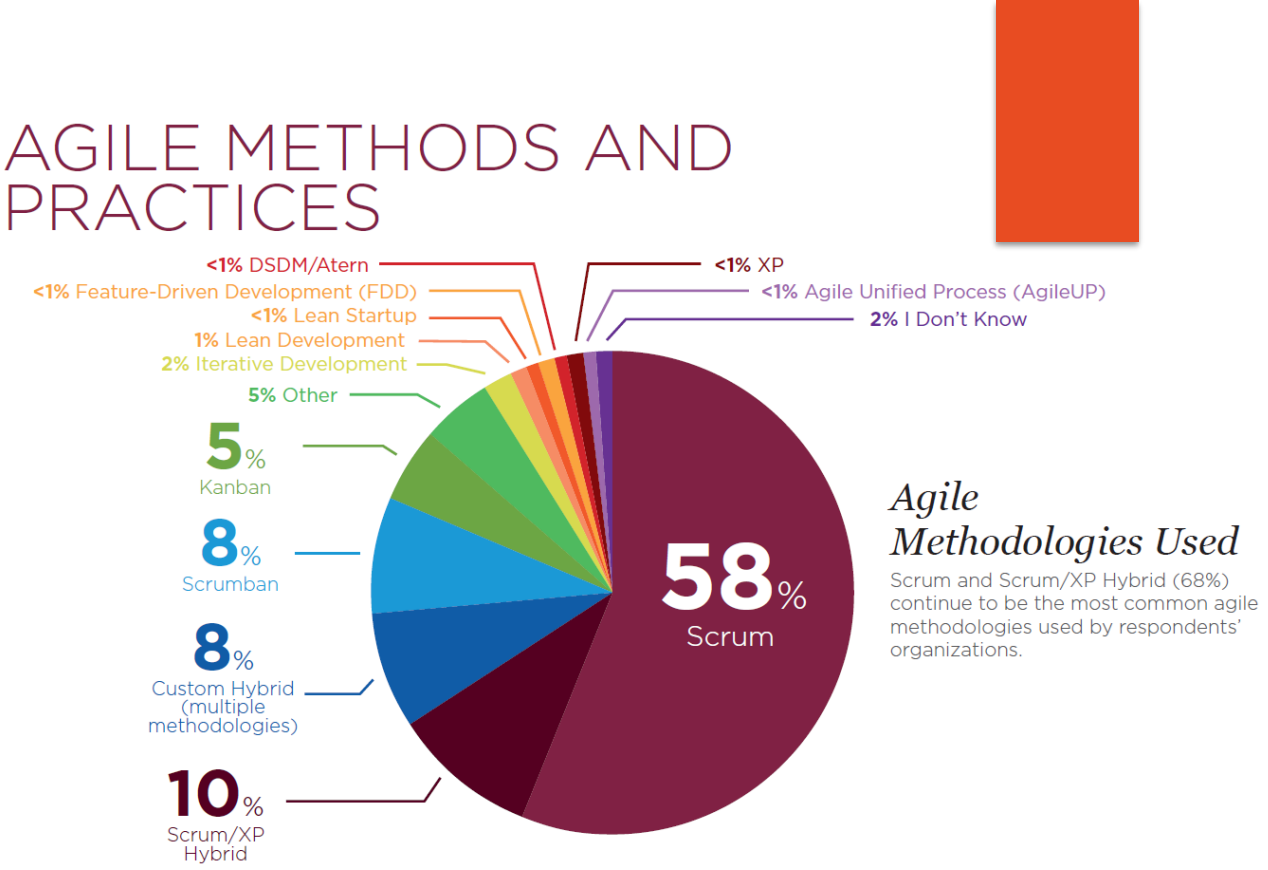
Concept: Iterative-Incremental

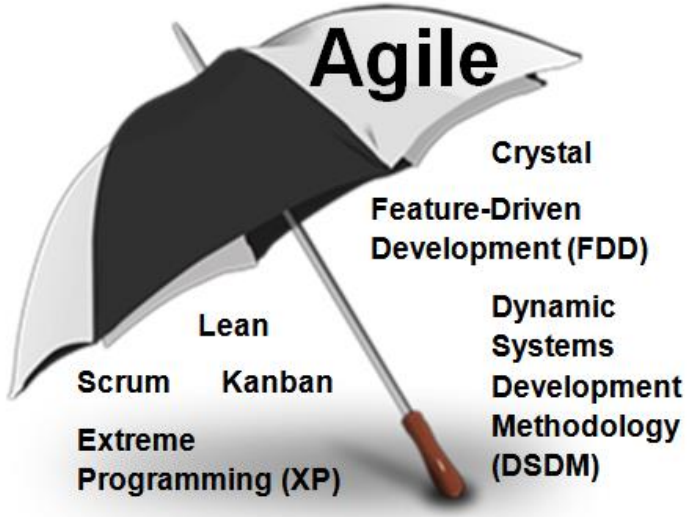


Agile Umbrella

Agile is a Conceptual (Mind-set) level where as the different frameworks are at (Implementation) level.

These Frameworks use Agile Values and Prnciples

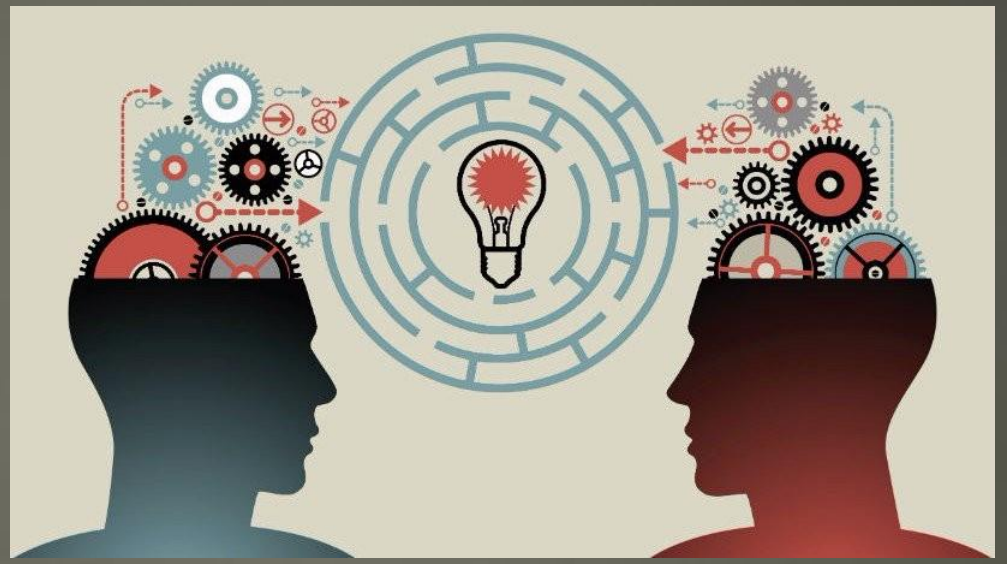





Scrum
Scrum

Scrum

- Scrum (n): A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.
  - Scrum is:
    - Lightweight
    - Simple to understand
    - Difficult to master



Scrum is founded on empirical process control theory, or empiricism. Empiricism asserts that knowledge comes from experience *and* making decisions based on what is known. Scrum employs an iterative, incremental approach to optimize predictability and control risk.

Scrum Pillars

Scrum users must frequently inspect Scrum artifacts and progress toward a Sprint Goal to detect undesirable variances.

Significant aspects of the process must be visible to those responsible for the outcome.

Those performing the work and those inspecting the resulting increment must share a common definition of "Done".

Transparency

Inspection

Adaption

If an inspector determines that one or more aspects of a process deviate outside acceptable limits, and that the resulting product will be unacceptable, the process or the material being processed must be adjusted. An adjustment must be made as soon as possible to minimize further deviation.

The Agile Scrum Framework at a Glance

Inputs from Executives, Team, Stakeholders, Customers, Users

Product Owner

The Team

Scrum Master

Burndown/up Charts

Daily Scrum Meeting

Every 24 Hours

1-4 Week Sprint

Sprint Review

1
2
3
4
5
6
7
8

Ranked list of what is required: features, stories, ...

Product Backlog

Team selects starting at top as much as it can commit to deliver by end of Sprint

Sprint Planning Meeting

Task Breakout

Sprint Backlog

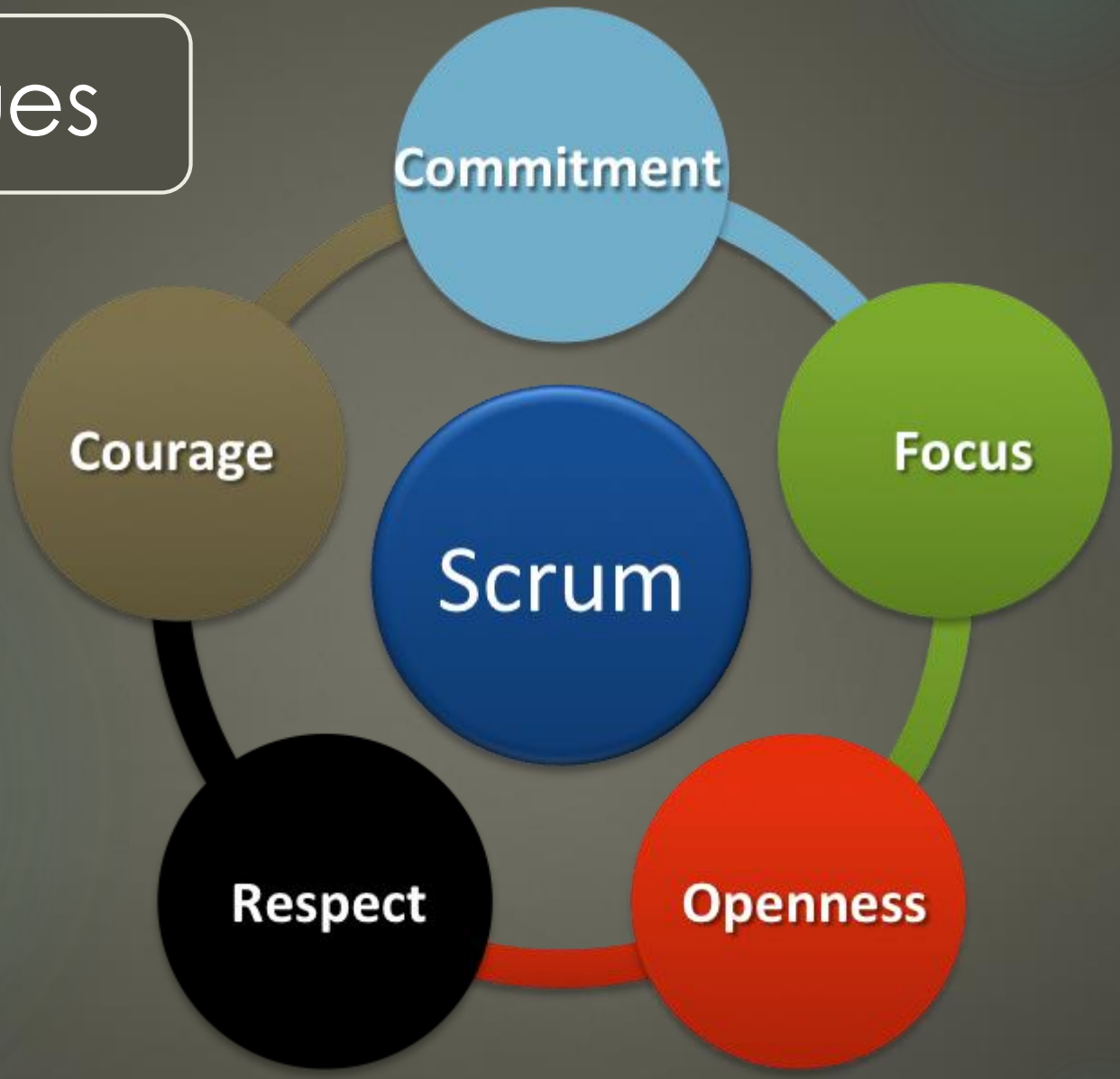
Sprint end date and team deliverable do not change

Finished Work

Sprint Retrospective

AGILE FOR ALL
Making Agile a Reality®

Scrum Values



https://guntherverheyen.com/2013/05/03/theres-value-in-the-scrum-values/

COURAGE

Scrum Team members have courage to do the right thing and work on tough problems

FOCUS

Everyone focuses on the work of the Sprint and the goals of the Scrum Team

COMMITMENT

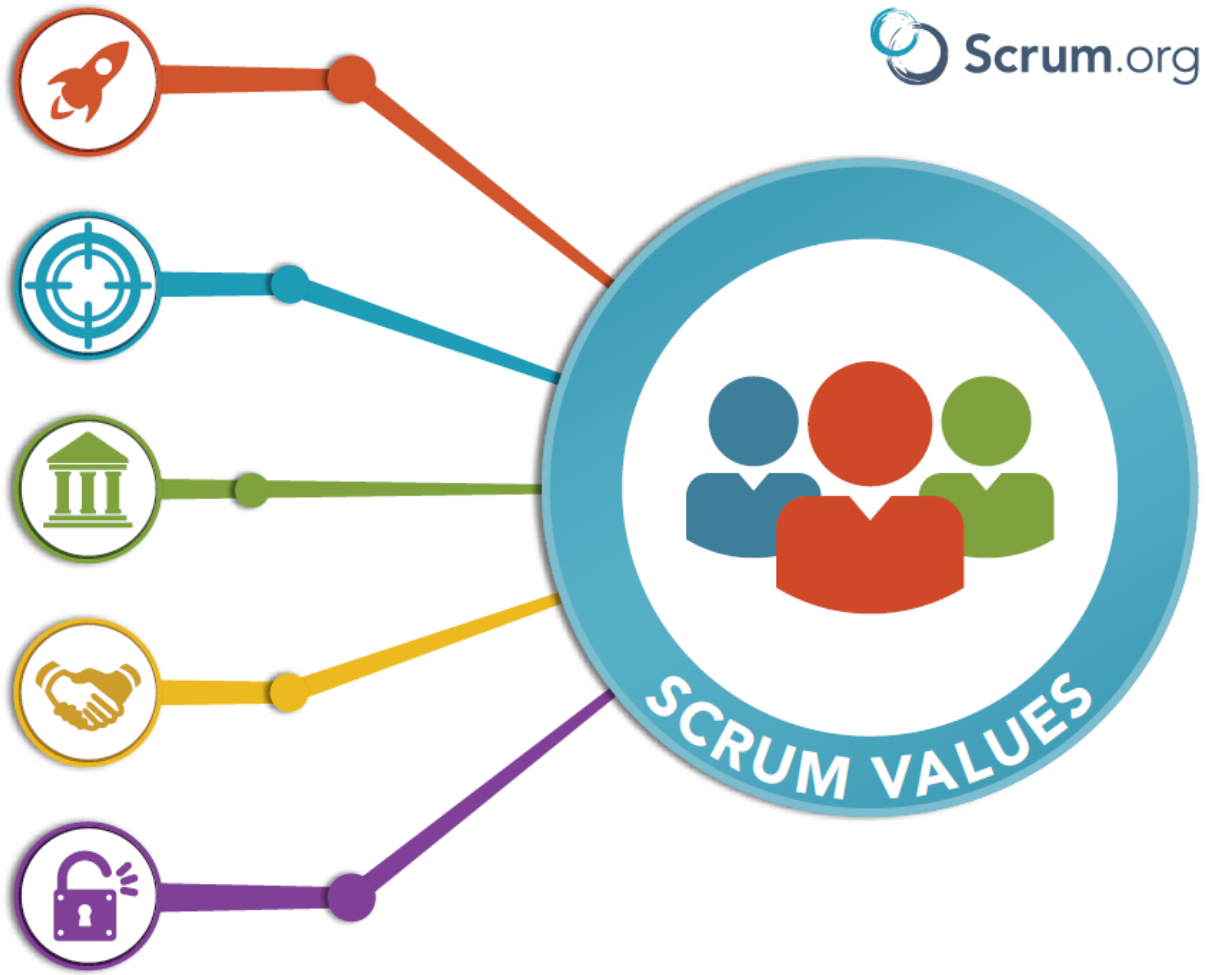
People personally commit to achieving the goals of the Scrum Team

RESPECT

Scrum Team members respect each other to be capable, independent people

OPENNESS

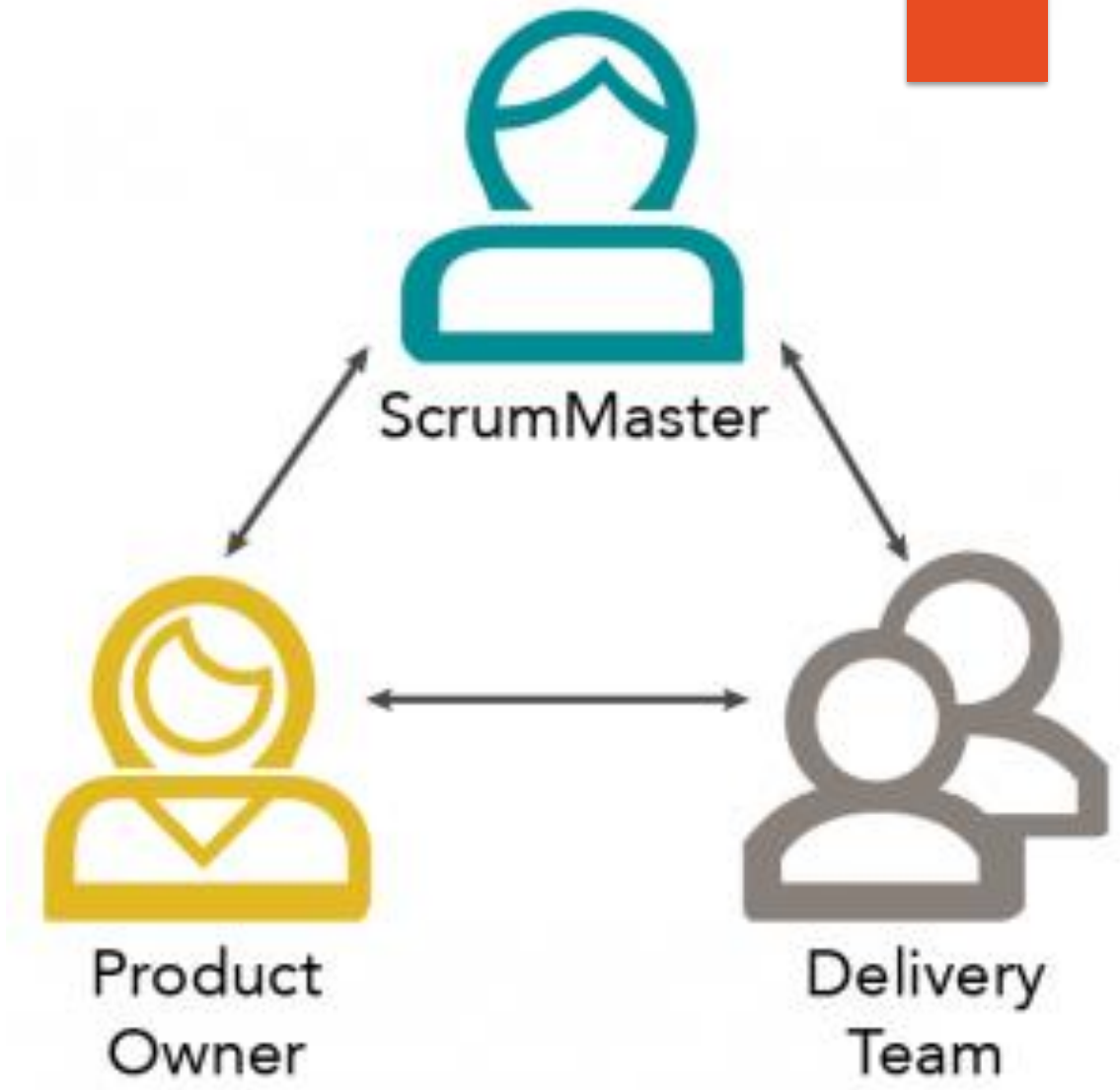
The Scrum Team and its stakeholders agree to be open about all the work and the challenges with performing the work



Scrum Roles

Scrum Team

- Product Owner
- Development Team
- Scrum Master



Product Owner

The Product Owner is responsible for maximizing the value of the product resulting from work of the Development Team. How this is done may vary widely across organizations, Scrum Teams, and individuals.

The Product Owner is the sole person responsible for managing the Product Backlog. Product Backlog management includes:

Clearly expressing Product Backlog items	Ordering the items in the Product Backlog to best achieve goals and missions	Optimizing the value of the work the Development Team performs;	Ensuring that the Product Backlog is visible, transparent, and clear to all, and shows what the Scrum Team will work on next	Ensuring the Development Team understands items in the Product Backlog to the level needed.

Product Owner

The Product Owner is Accountable for the Product Backlog

The Product Owner is one person, not a committee. The Product Owner may represent the desires of a committee in the Product Backlog, but those wanting to change a Product Backlog item's priority must address the Product Owner.

For the Product Owner to succeed, the entire organization must respect his or her decisions. The Product Owner's decisions are visible in the content and ordering of the Product Backlog. No one can force the Development Team to work from a different set of requirements.

A Good Product owner should be/have ..

ACCOUNTABLE

AVAILABILITY

EMPOWERED

INNOVATIVE

COMMUNICATION

KNOWLEDGEABLE

Development Team

Team size (3-9)

Responsible for delivering a potentially releasable Increment of “Done” product at the end of each Sprint.

Only members of the Development Team create the Increment.

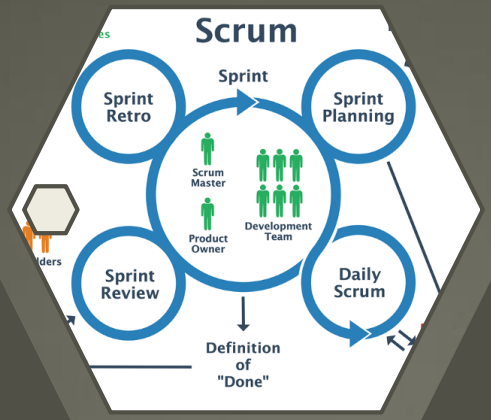
Development Teams have the following characteristics:

- They are self-organizing. No one (not even the Scrum Master) tells the Development Team how to turn Product Backlog into Increments of potentially releasable functionality.
- Development Teams are cross-functional, with all the skills as a team necessary to create a product Increment.
- Scrum recognizes no titles for Development Team members, regardless of the work being performed by the person;
- Scrum recognizes no sub-teams in the Development Team, regardless of domains that need to be addressed like testing, architecture, operations, or business analysis; and,
- Individual Development Team members may have specialized skills and areas of focus, but accountability belongs to the Development Team as a whole.

Scrum Master

The Scrum Master is a servant-leader for the Scrum Team.

The Scrum Master is responsible for promoting and supporting Scrum.

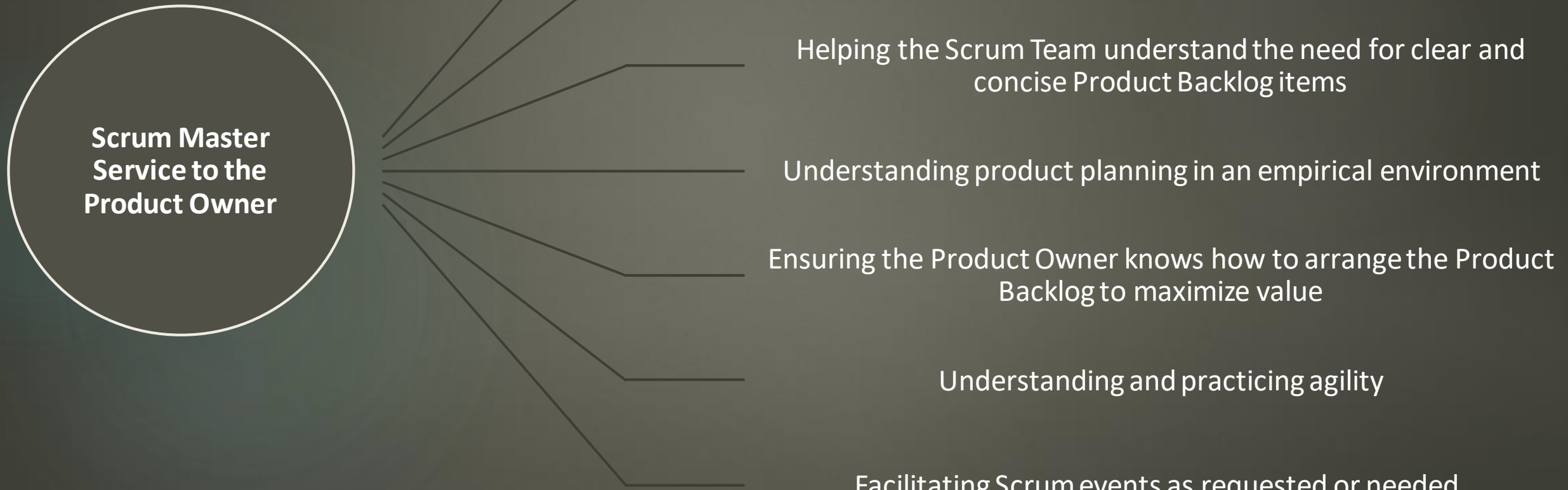


Scrum Masters do this by helping everyone understand Scrum theory, practices, rules, and values.

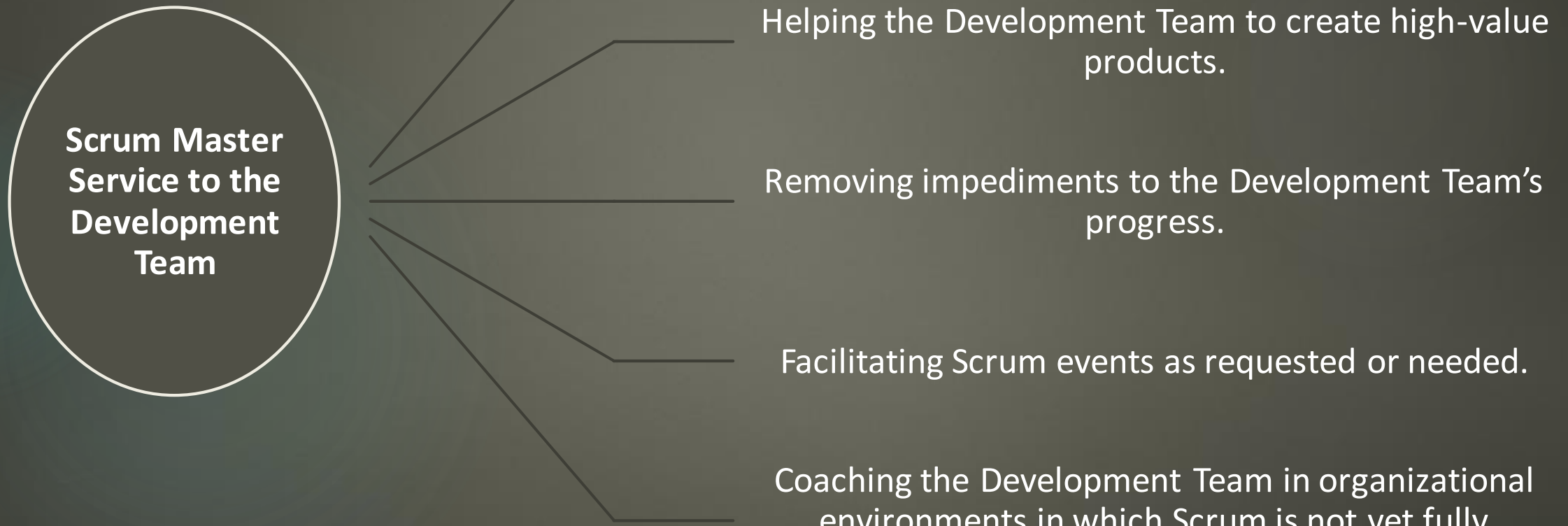
The Scrum Master helps those outside the Scrum Team understand which of their interactions with the Scrum Team are helpful and which aren't.

The Scrum Master helps everyone change these interactions to maximize the value created by the Scrum Team.

Scrum Master



Scrum Master



Scrum Master

Scrum Events

Sprint

(Not More than 1 Calendar Month)

Sprint Planning

(8hrs for 1 Month Sprint)

Daily Scrum

(15 min)

Sprint Review

(4hrs for 1 Month Sprint)

Sprint Retrospective

(3hrs for 1 Month Sprint)

Sprint

Sprint

(Not More than 1 Calendar Month)

The heart of Scrum is a Sprint,

Time-box of one month or less during which a “Done”, useable, and potentially releasable product Increment is created.

Sprints have consistent durations throughout a development effort.

A new Sprint starts immediately after the conclusion of the previous Sprint.

Sprints contain and consist of the Sprint Planning, Daily Scrums, the development work, the Sprint Review, and the Sprint Retrospective.

During the Sprint

No changes are made that would endanger the Sprint Goal

Quality goals do not decrease

Scope may be clarified and re-negotiated between the Product Owner and Development Team as more is learned.

Cancelling a Sprint:

A Sprint can be cancelled before the Sprint time-box is over. Only the Product Owner has the authority to cancel the Sprint, although he or she may do so under influence from the stakeholders, the Development Team, or the Scrum Master.

Sprint Planning

Sprint Planning

(8hrs for 1 Month Sprint)

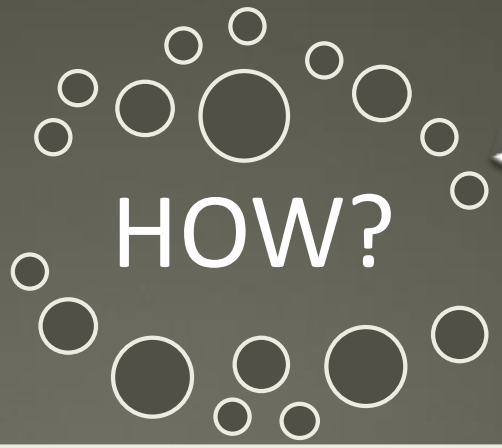
Sprint Planning Answers the following:	
What Can Be Delivered in the Increment resulting from the upcoming Sprint	How will the work needed to deliver the Increment be achieved

What

The Development Team works to forecast the functionality that will be developed during the Sprint	The Product Owner discusses the objective that the Sprint should achieve and the Sprint Goal	Inputs for this • Product Backlog • Latest Product Backlog • Projected Capacity of the Development Team • Past Performance	Only Development Team chooses what they can accomplish over the upcoming sprint	The Sprint Goal is Crafted during the Sprint Planning

The Sprint Goal is an objective that will be met within the Sprint through the implementation of the Product Backlog, and it provides guidance to the Development Team on why it is building the Increment.

Sprint Planning



The Product Backlog items selected for this Sprint plus the plan for delivering them is called the Sprint Backlog.

The Development Team decides how it will build this functionality into a “Done” product Increment during the Sprint

The Development Team self-organizes to undertake the work in the Sprint Backlog, both during Sprint Planning and as needed throughout the Sprint.

Work planned for the first days of the Sprint by the Development Team is decomposed by the end of this meeting, often to units of one day or less.

The Product Owner can help to clarify the selected Product Backlog items and make trade-offs. If the Development Team determines it has too much or too little work, it may renegotiate the selected Product Backlog items with the Product Owner.

The Development Team may also invite other people to attend to provide technical or domain advice

By the end of the Sprint Planning, the Development Team should be able to explain to the Product Owner and Scrum Master how it intends to work as a self-organizing team to accomplish the Sprint Goal and create the anticipated Increment.

Sprint Goal

- S- Specific
- M- Measurable
- A-Attainable
- R- Relevant
- T- Time Bound

The Sprint Goal is an objective set for the Sprint that can be met through the implementation of Product Backlog

It provides guidance to the Development Team on why it is building the Increment. It is created during the Sprint Planning meeting

The Sprint Goal gives the Development Team some flexibility regarding the functionality implemented within the Sprint.

As the Development Team works, it keeps the Sprint Goal in mind. In order to satisfy the Sprint Goal, it implements functionality and technology.

If the work turns out to be different than the Development Team expected, they collaborate with the Product Owner to negotiate the scope of Sprint Backlog within the Sprint.

Daily Scrum

Daily Scrum
(15 min)

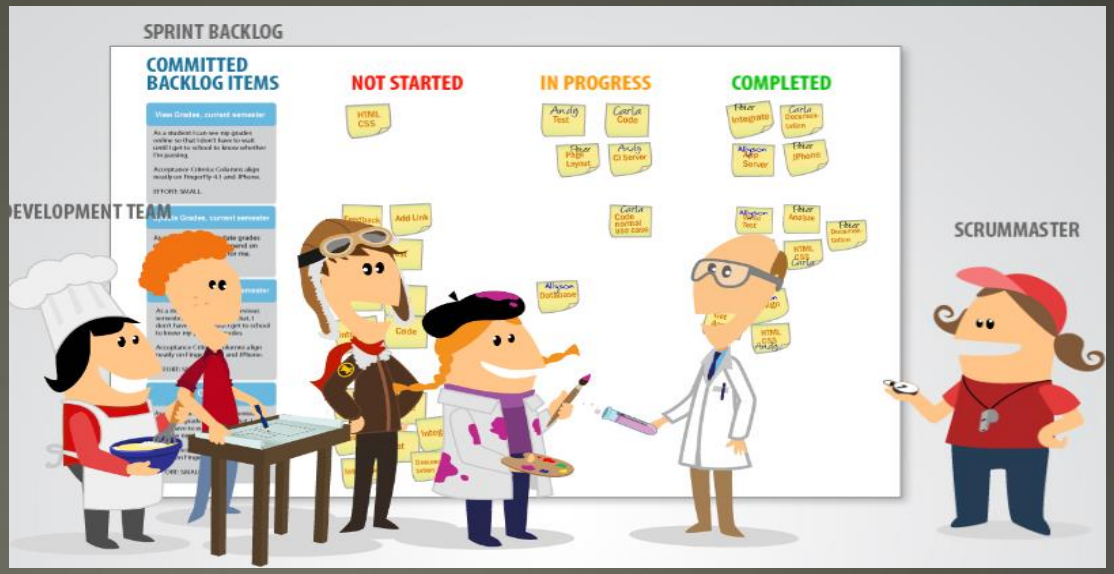
Daily Scrum is held every day of the Sprint.

Development Team plans work for the next 24 hours. This optimizes team collaboration and performance by inspecting the work since the last Daily Scrum and forecasting upcoming Sprint work.

The Daily Scrum is held at the same time and place each day to reduce complexity.

- What did I do yesterday that helped the Development Team meet the Sprint Goal?
- What will I do today to help the Development Team meet the Sprint Goal?
- Do I see any impediment that prevents me or the Development Team from meeting the Sprint Goal?

Daily Scrum



The Development Team or team members often meet immediately after the Daily Scrum for detailed discussions, or to adapt, or re-plan, the rest of the Sprint's work.

The Scrum Master ensures that the Development Team has the meeting, but the Development Team is responsible for conducting the Daily Scrum.

The Scrum Master teaches the Development Team to keep the Daily Scrum within the 15-minute time-box.

The Daily Scrum is an internal meeting for the Development Team. If others are present, the Scrum Master ensures that they do not disrupt the meeting.

Daily Scrums improve communications, eliminate other meetings, identify impediments to development for removal, highlight and promote quick decision-making, and improve the Development Team's level of knowledge.

This is a key inspect and adapt meeting.

Sprint Review

Sprint Review
(4 hrs for 1 Month Sprint)

A Sprint Review is held at the end of the Sprint to inspect the Increment and adapt the Product Backlog if needed.

During the Sprint Review, the Scrum Team and stakeholders collaborate about what was done in the Sprint.

Decide on what are the next things to be done to Optimize Value

This is an informal meeting, not a status meeting, and the presentation of the Increment is intended to elicit feedback and foster collaboration.

Sprint Review

Attendees include the Scrum Team and key stakeholders invited by the Product Owner

The Product Owner explains what Product Backlog items have been “Done” and what has not been “Done”

The Development Team discusses what went well during the Sprint, what problems it ran into, and how those problems were solved

The Development Team demonstrates the work that it has “Done” and answers questions about the Increment

The Product Owner discusses the Product Backlog as it stands. He or she projects likely target and delivery dates based on progress to date (if needed)

The entire group collaborates on what to do next, so that the Sprint Review provides valuable input to subsequent Sprint Planning

Review of how the marketplace or potential use of the product might have changed what is the most valuable thing to do next

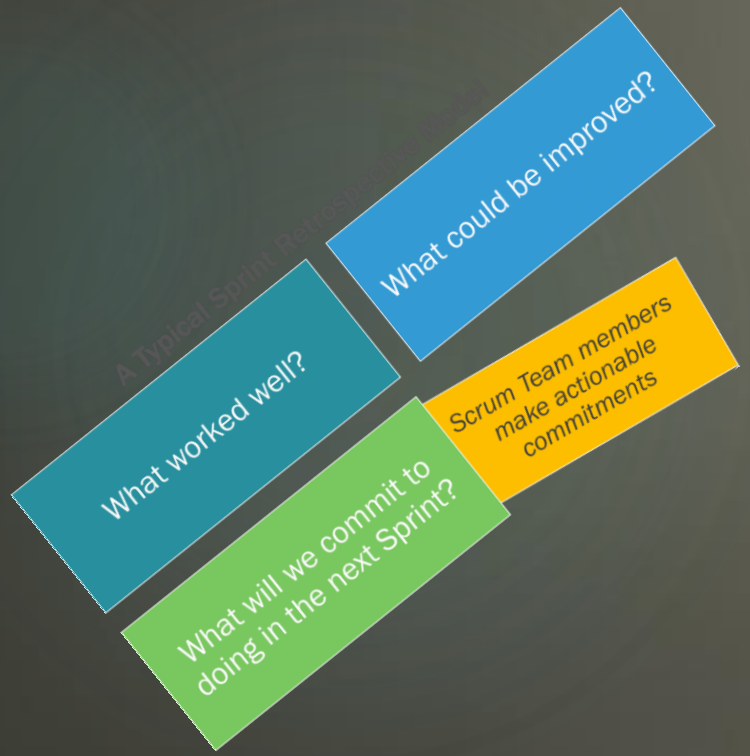
Review of the timeline, budget, potential capabilities, and marketplace for the next anticipated releases of functionality or capability of the product

Sprint Retrospective

Sprint Retrospective
(3 hrs for 1 Month Sprint)

The Sprint Retrospective is an opportunity for the Scrum Team to inspect itself and create a plan for improvements to be enacted during the next Sprint

The Scrum Master ensures that the meeting is positive and productive. The Scrum Master teaches all to keep it within the time-box. The Scrum Master participates as a peer team member in the meeting from the accountability over the Scrum process.



The purpose of the Sprint Retrospective

Inspect how the last Sprint went with regards to people, relationships, process, and tools.	Identify and order the major items that went well and potential improvements.	Create a plan for implementing improvements to the way the Scrum Team does its work.

Sprint Retrospective

The Scrum Master encourages the Scrum Team to improve, within the Scrum process framework, its development process and practices to make it more effective and enjoyable for the next Sprint.

During each Sprint Retrospective, the Scrum Team plans ways to increase product quality by improving work processes or adapting the definition of “Done”, if appropriate and not in conflict with product or organizational standards

By the end of the Sprint Retrospective, the Scrum Team should have identified improvements that it will implement in the next Sprint.

Implementing these improvements in the next Sprint is the adaptation to the inspection of the Scrum Team itself.

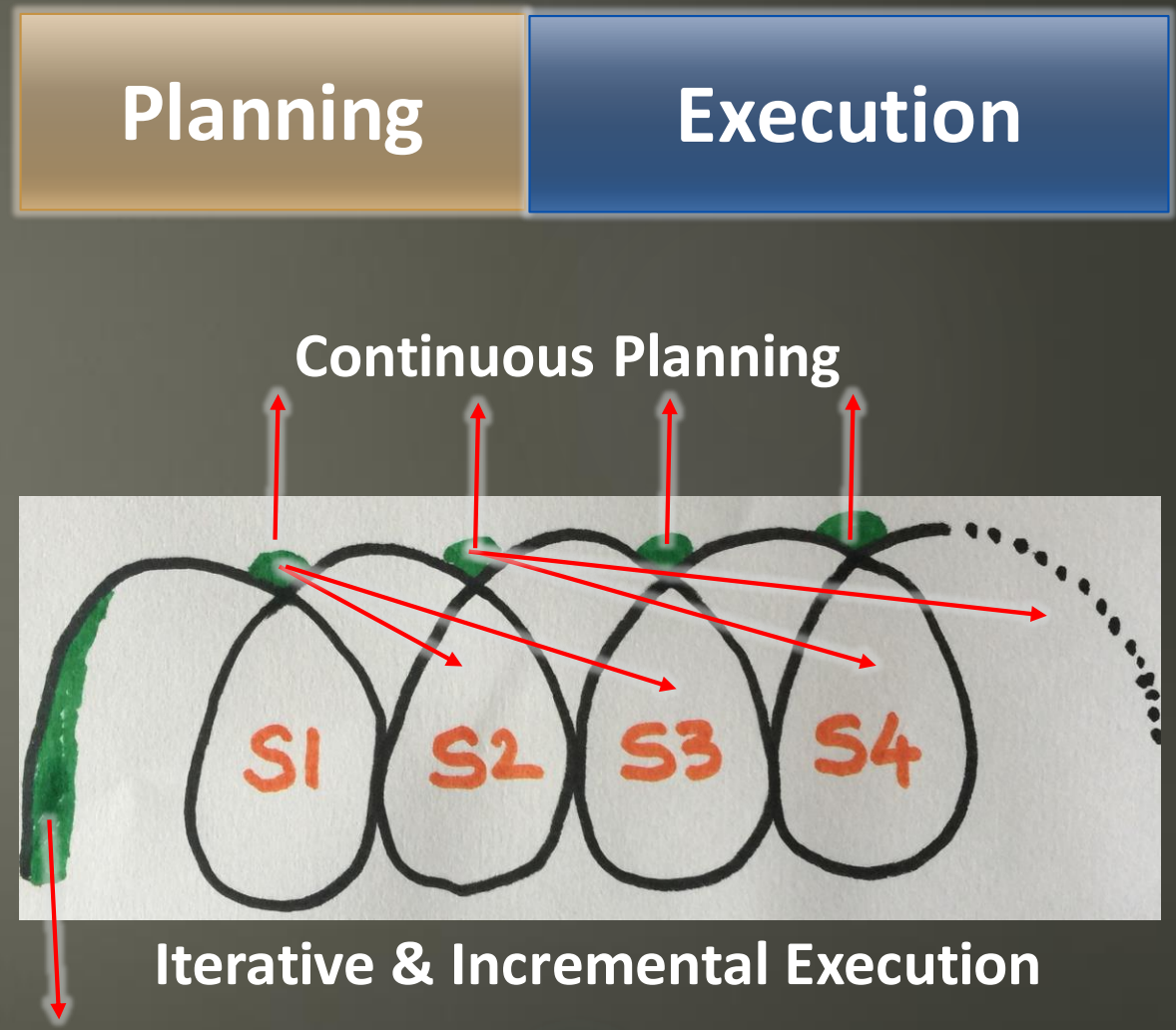
Although improvements may be implemented at any time, the Sprint Retrospective provides a formal opportunity to focus on inspection and adaptation.

Recap

Event	Time Box	Input	Output	Audience
Sprint	1 Calendar month	Ready product Backlog Items	Potentially Releasable Product Increment	Scrum Team and Dependant Teams
Sprint planning	8hrs (1 month Sprint)	Product Backlog Latest Product Backlog Definition of Done Past Performance Teams Capacity	Sprint Backlog Sprint Goal	Scrum Team(Mandatory) Dependant Team's (If Required) SME's (if Required)
Daily Scrum	15 min	Updated Sprint Board	Plan for Next 24 hours	Development Team (Mandatory) Others Optional
Sprint Review	4 hours (1 Month Sprint)	Product Increment	Feedback Enhancements Updated Product Backlog	Stakeholders and Scrum Team
Sprint Retrospective	3 hours (1 Month Sprint)	What went will What Could have been Better Action Items w.r.t (People,Process,Tools)	Action Items to be Implemented. Revised "Definition of DONE"	Scrum Team.

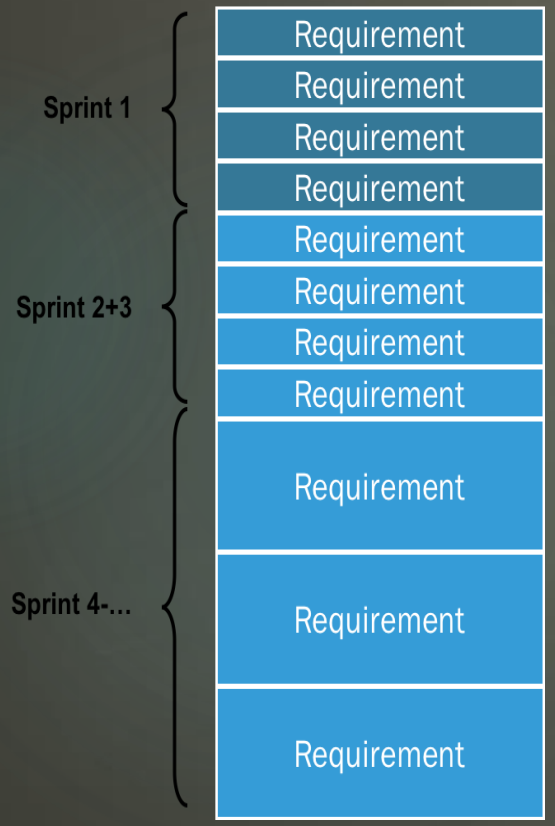
Product Backlog Refinement

- Is an activity, not a meeting
- Can take 10% of sprint capacity
- PO and Development team attend
- Done during the sprint
- PO leads the activity
- Not for current sprint, for future sprints
- Activities of PBR:
  - Split stories
  - Size stories
  - Discuss design
  - Write acceptance tests
  - Merge stories
  - Remove stories
  - …



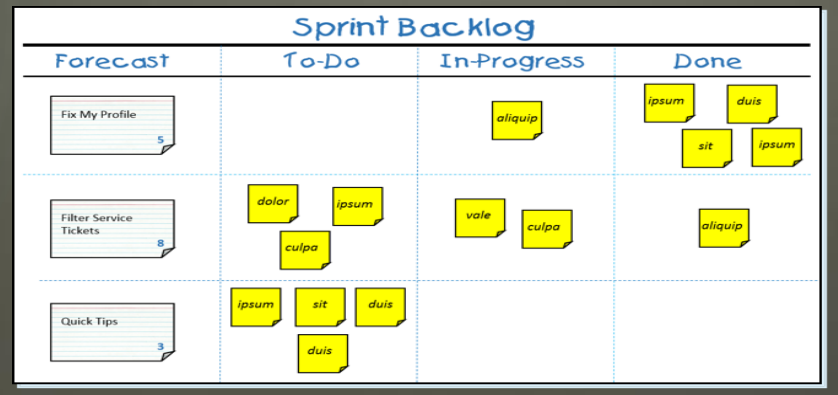
Scrum Artificats

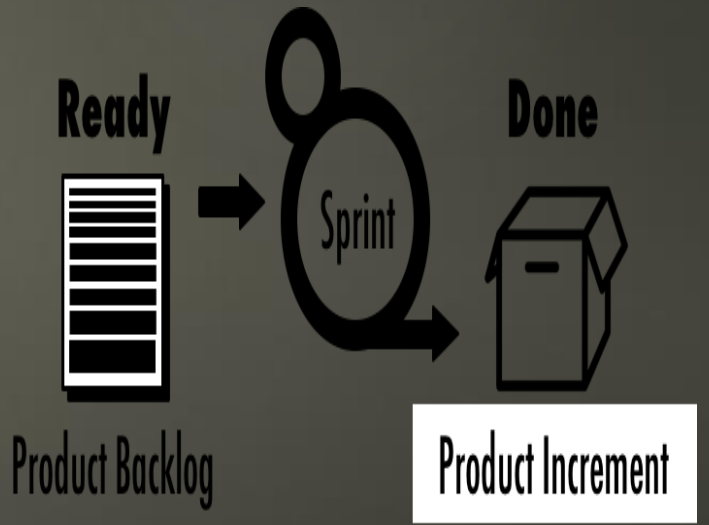
Scrum's artifacts represent work or value to provide transparency and opportunities for inspection and adaptation. Artifacts defined by Scrum are specifically designed to maximize transparency of key information so that everybody has the same understanding o f the artifact.



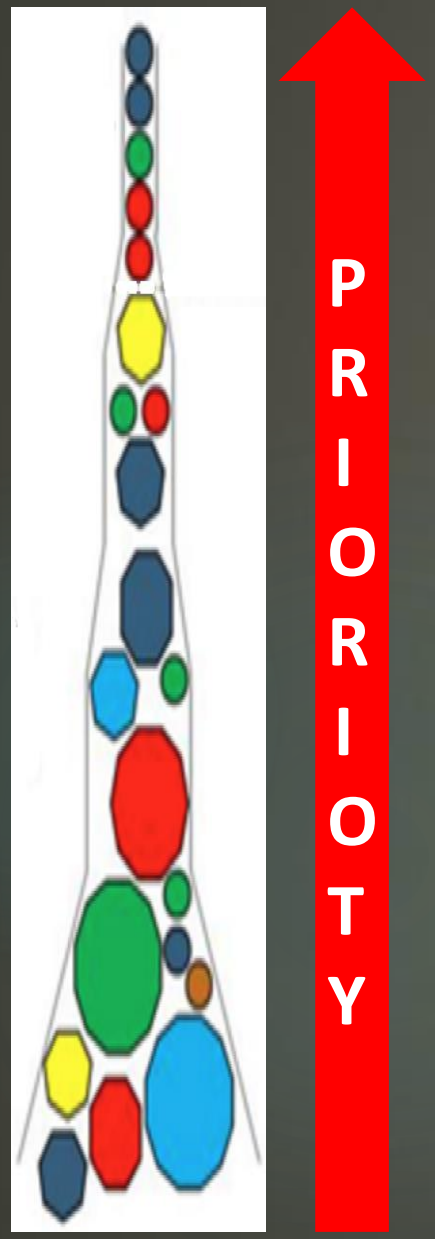
Scrum Artifacts

- Product Backlog
- Sprint Backlog
- Product Increment





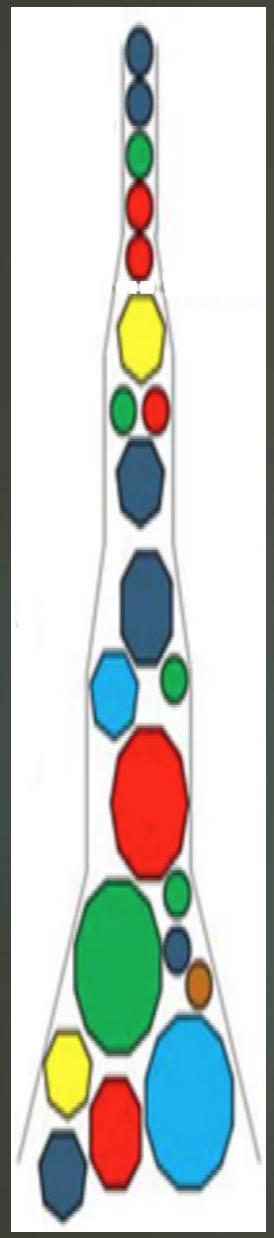
Product Backlog



- Prioritized list of items
- Contains:
  - New features
  - Defects
  - Technical Work
  - Infrastructural
  - Knowledge work
- Owns by Product owner
- Any one can add items

- Only PO can change priority
- Progressively elaborated
- All items are stack ranked
- Items should have:
  - Order
  - Description
  - Value
  - Size
  - Acceptance criteria
  - …

Product Backlog



PRIORITY

- Living document
- Visible and accessible to everyone
- Continuously gets updated
- Managed through backlog refinement
- Top items are “User stories”
- Top items are:
  - More granular
  - High value
- Bottom items are generally “Epics”
- Bottom items are:
  - Less granular
  - Low value

Mike Cohn Says a Backlog should be:

- **D – Detailed appropriately**
- **E – Emergent**
- **E – Estimated**
- **P – Prioritized**

Product Backlog - Recap

The development team is responsible for estimating the product backlog items

Anyone can create backlog items, but product owner has overall responsibility to prioritize the backlog items

Development team may work on critical engineering items without placing them in product backlog

A single development team works from multiple product backlogs

Product Backlog - Recap

Each product backlog item has description, value, estimate, and order associated with it

Once an item is placed in the product backlog, it is never re-ordered

Product owner keeps the product backlog somewhere secretly so that no one can see it

Product Backlog - Recap

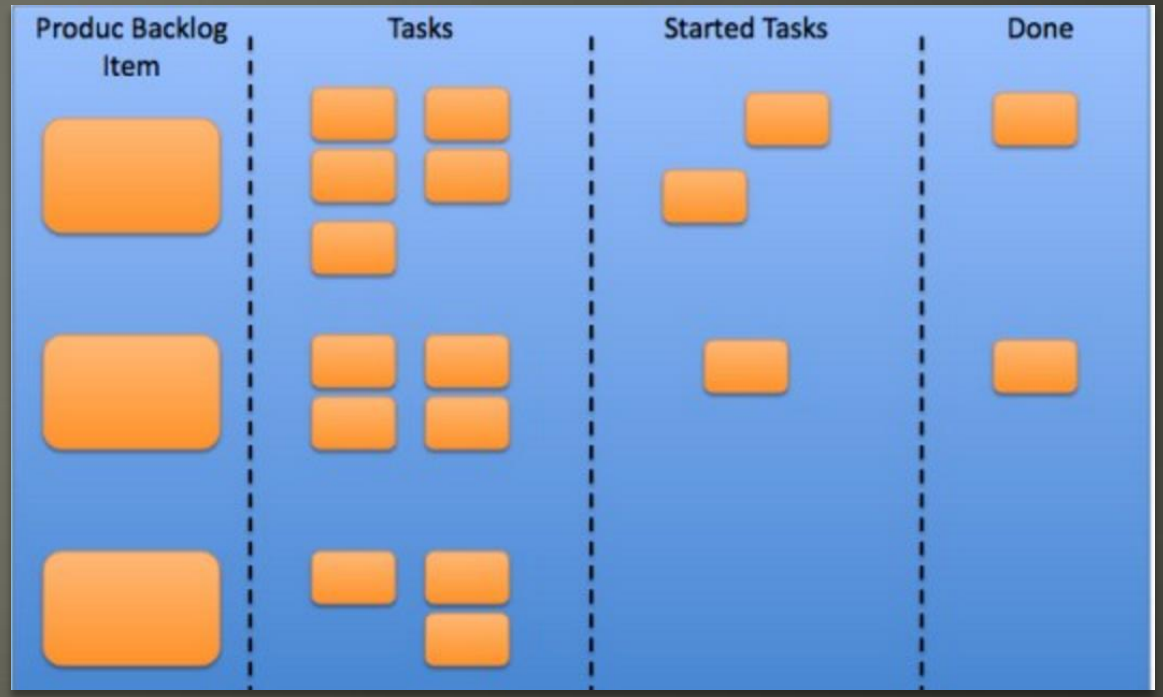
Higher order product backlog items are usually clear and more detailed than lower ordered backlog items

Product backlog contains functional, non-functional, infrastructural and defects

The product backlog is always sorted from small items at the top to large items at the bottom

Sprint Backlog

- Contains the backlog items & corresponding tasks of a sprint
- Output of Sprint planning meeting
- Owns by Development team
- Helps Development team to plan and organize their work
- Gets updated during the sprint every day
- Is a plan for Development team
- Is a forecast of what will be delivered in the sprint
- It emerges during the sprint
- Tasks will be updated with remaining effort
- Completed tasks will be moved to “Done” state
- Completed stories will be moved to “Done”
- Only Development team can change Sprint backlog
- Highly visible and frequently updated during the sprint
- Helps to monitor the sprint progress



Product Increment that is Potentially Releasable

- Collection of all Sprint backlog items that are completed
- The new increment that comes out of a Sprint must meet DOD
- The increment must be in a usable condition
- It should be demo-able to the product owner
- It must satisfy all the acceptance test cases
- Current increment must not disturb the existing features
- It should create business value and complete
- If required, it should be pushed to production “as-is” and without any additional work required

Potentially Releasable Product Increment:

- Increases the Business Value
- Reduces the Risk by receiving feedback
- Will create transparency to the stakeholders

Artificat Transparency

Scrum relies on transparency.

Decisions to optimize value and control risk are made based on the perceived state of the artifacts.

To the extent that transparency is complete, these decisions have a sound basis.

To the extent that the artifacts are incompletely transparent, these decisions can be flawed, value may diminish and risk may increase.

The Scrum Master must work with the Product Owner, Development Team, and other involved parties to understand if the artifacts are completely transparent

A Scrum Master can detect incomplete transparency by inspecting the artifacts, sensing patterns, listening closely to what is being said, and detecting differences between expected and real results.

The Scrum Master's job is to work with the Scrum Team and the organization to increase the transparency of the artifacts.

This work usually involves learning, convincing, and change. Transparency doesn't occur overnight, but is a path.

Agreements

Definition of Ready

Definition of Done

Definition of Ready

Criteria Defined and Agreed by the Team

What should Come into the Sprint

"Ready"
"Ready"

DOR Can be for Sprints, EPIC/ Features, User Stories

Improves the Transparency and Quality of Product/Sprint Backlog

https://www.mountaingoatsoftware.com/blog/the-dangers-of-a-definition-of-ready

https://www.scrum.org/resources/blog/walking-through-definition-ready

Definition of Done

Scrum Team to have Shared understanding of what it means for work to be complete,

This ensures transparency.

Help Team in Picking up the Backlog Items for Next Sprints

The purpose of each Sprint is to deliver Increments of potentially releasable functionality that adhere to the Scrum Team's current definition of "Done."

Development Team should adhere all the Points Agreed in the "Done" Criteria

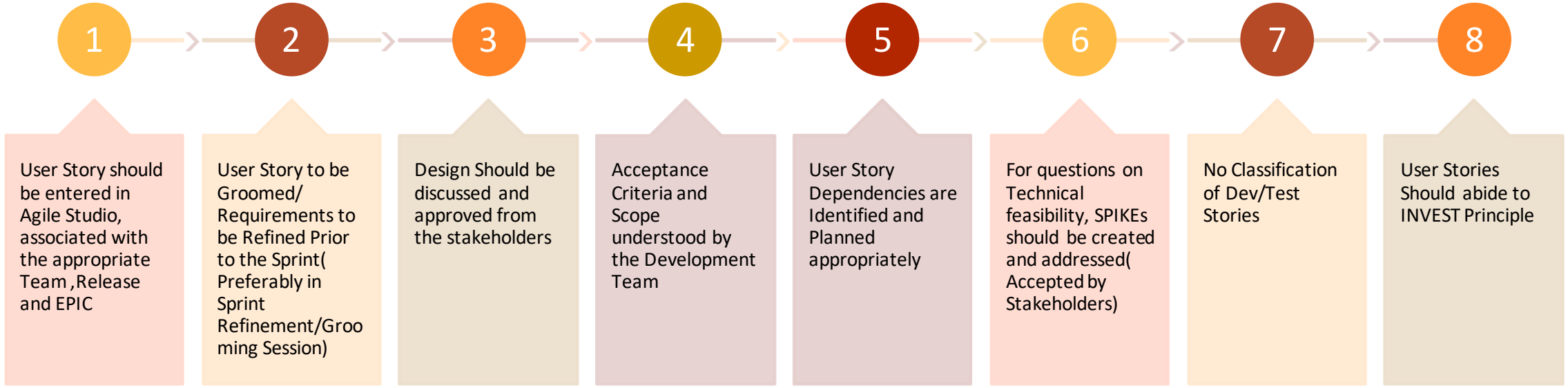
If there are multiple Scrum Teams working on the system or product release, the Development Teams on all the Scrum Teams must mutually define the definition of "Done."

As Scrum Teams mature, it is expected that their definitions of "Done" will expand to include more stringent criteria for higher quality.

EPIC-Definition of Ready

- EPIC Should be associated to Team, GOAL and Release in Agile Studio.
- Identify the Stakeholders, Dependent Teams and Engineering Owner
- Scope [In-Scope, Out of Scope] and Acceptance Criteria of EPIC's to be clearly drafted in Agile Studio
- High-Level Design for Epic to be added in Agile Studio and be reviewed by Stakeholders (Approvers for Design to be identified)
- Notify the Dependencies to respective teams preferably before EDOR.
- EPIC should be broken down into Detailed User Stories that can be Sized, Groomed/Refined and Prioritized
- High level Test Plan to be written and approved by EDOR time
- Estimated Story Points/ Sprints to be entered in Agile Studio
- Delivery Dates [Design, Development, Adoption] of EPIC should be entered by Product Owner in Agile Studio

USER STORY-Definition of Ready



EPIC-Definition of Done

- All the User Stories related to the EPIC must be Closed/ Resolved
- No outstanding Issues/ Bugs
- All the Test Cases pertaining to EPIC should be Passed
- Necessary documents to be published and uploaded onto Agile Studio
- Update Actual Delivery Dates [Design, Development, Adoption] of EPIC should be entered by Product Owner in Agile Studio.
- EPIC should be moved to Adoption Stage
- EDOD- Done, Approved from all the Stakeholders

USER STORY-Definition of Done

Acceptance Criteria of User story Met	Code Review done, Code Review Comments Addressed	Code Merged to Master	Percentage of Code Coverage- To be identified by Team
All the Test Cases to be uploaded onto Agile Studio	All the Test Cases should be “Passed”	Test Results to be uploaded (if required)	All the Tasks should be in “DONE” State
No Open Issues/bugs associated to User story.	Design Docs/ Relevant docs to be uploaded onto Agile Studio and tagged as part of User story documents	Quality leads from the team to provide a signoff before the story is moved to “Pending - verification”	Any of the above points not met before the Sprint Review, the User Story is not considered as “DONE”

Metrics [Measure Everything That Results In Customer Satisfaction]

Burn-down

- Sprint Burndown
- Release Burndown

A **Burn down chart** is a graphical representation of work left to do versus time

Burnup

- Sprint Burnup
- Release Burnup

A **Burn up chart** is a graphical representation of work Completed vs Total work to be Completed

Cumulative Flow Diagram

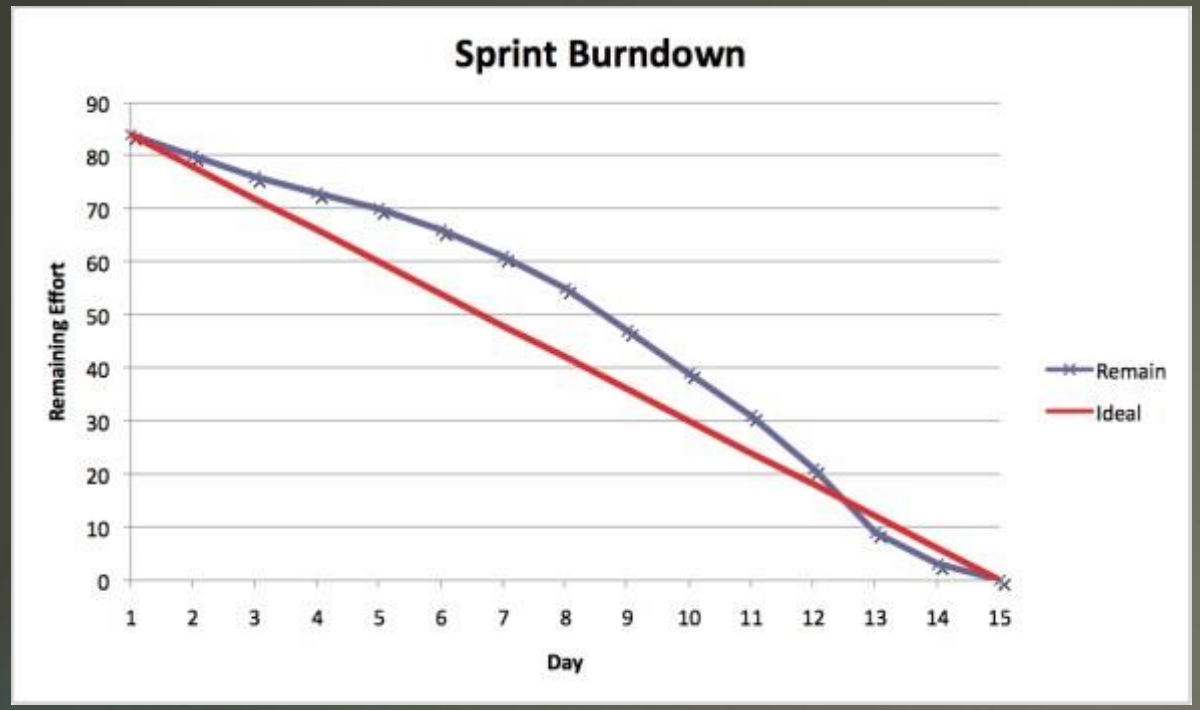
A **cumulative flow diagram** is a tool used in queuing theory. It is an area graph that depicts the quantity of work in a given state, showing arrivals, time in queue, quantity in queue, and departure.

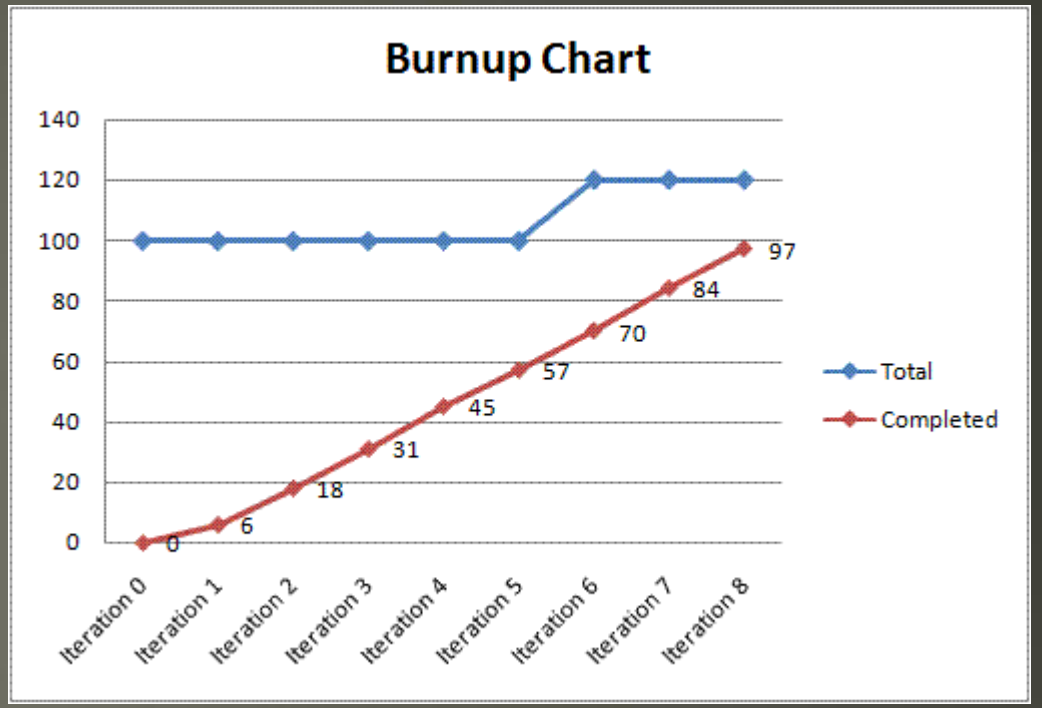
Metrics [Measure Everything That Results In Customer Satisfaction]

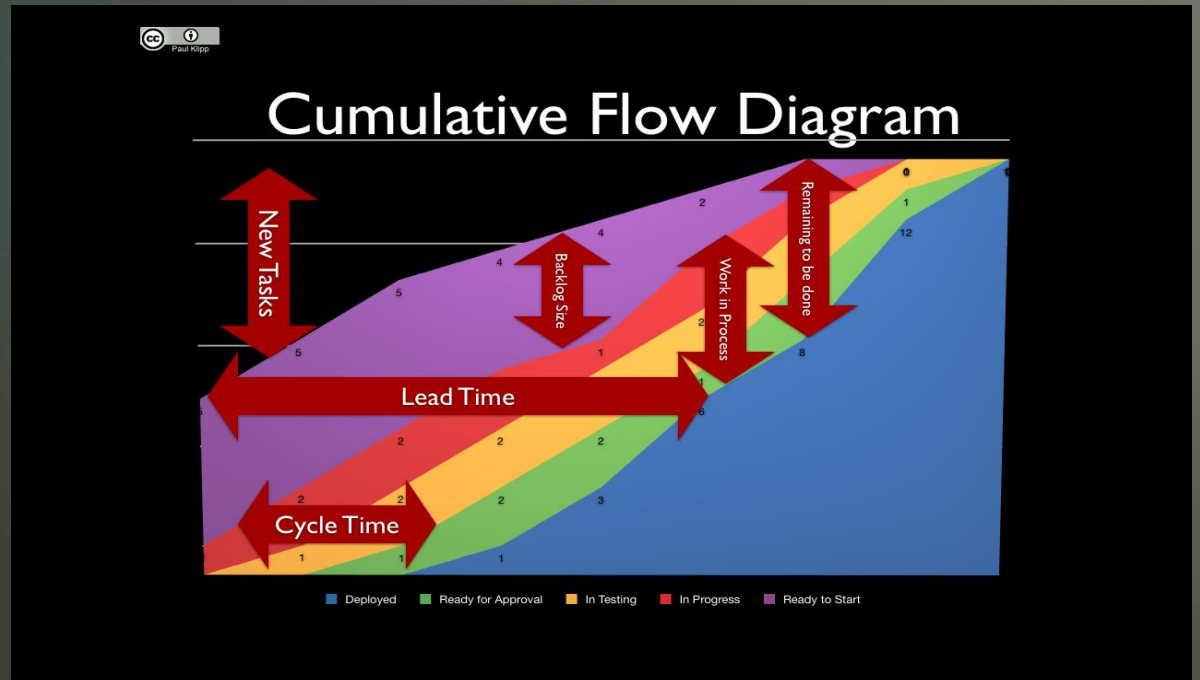
Lead Time → Cycle Time

Lead time measures the **time** elapsed between order and **delivery**, thus it measures your production process from your customer's perspective.

Cycle Time starts when the actual work begins on the unit and ends when it is ready for delivery.


Sprint Burndown
Remaining Effort
90
80
70
60
50
40
30
20
10
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Day
Remain
Ideal


Burnup Chart
140
120
100
80
60
40
20
0
0
6
18
31
45
57
70
84
97
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Iteration 7
Iteration 8
Total
Completed


Cumulative Flow Diagram
New Tasks
Backlog Size
Work in Process
Remaining to be done
Lead Time
Cycle Time
Deployed
Ready for Approval
In Testing
In Progress
Ready to Start

Estimations

Estimation is predicting the size, Cost, Schedule etc..

Estimations are guess values that are derived from Conditions , parameters and Criteria.

Estimation Generally has

- Accuracy: How Something is close to Reality
- Precision : Degree to Which Repeated measurements shown (consistency)

Definition of Done Plays a Vital role in Estimations,

- DOD gives you a clear picture on the Scope of the Work

3 Common Techniques for Estimations

- Expert Opinion
- Analogy
- Disaggregation

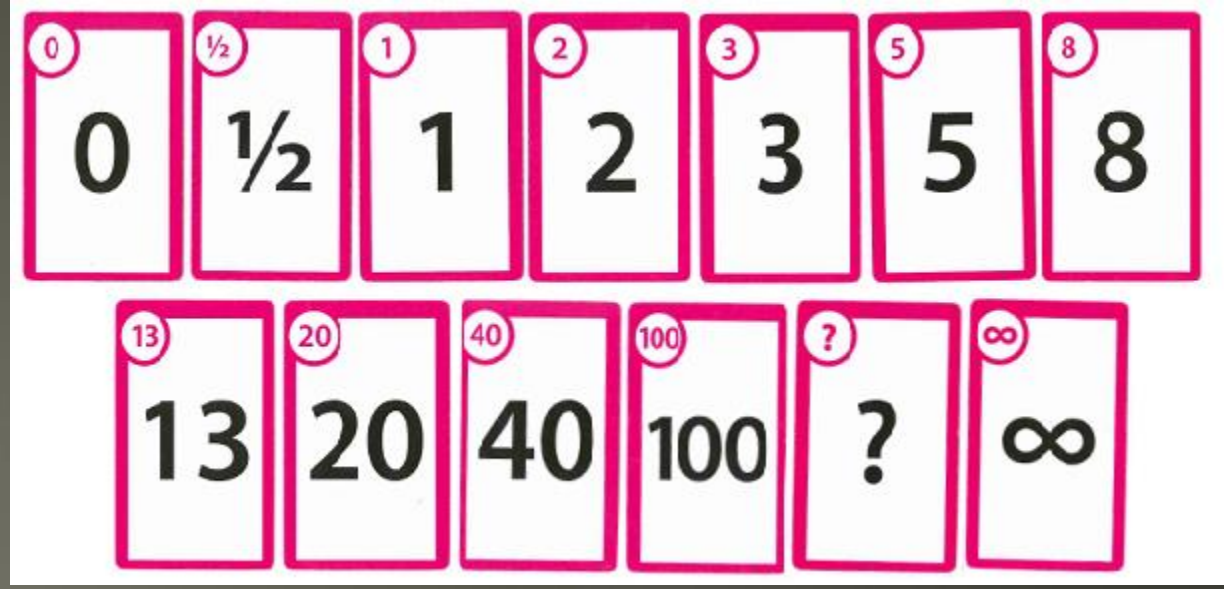
3 Factors that Influence Estimations

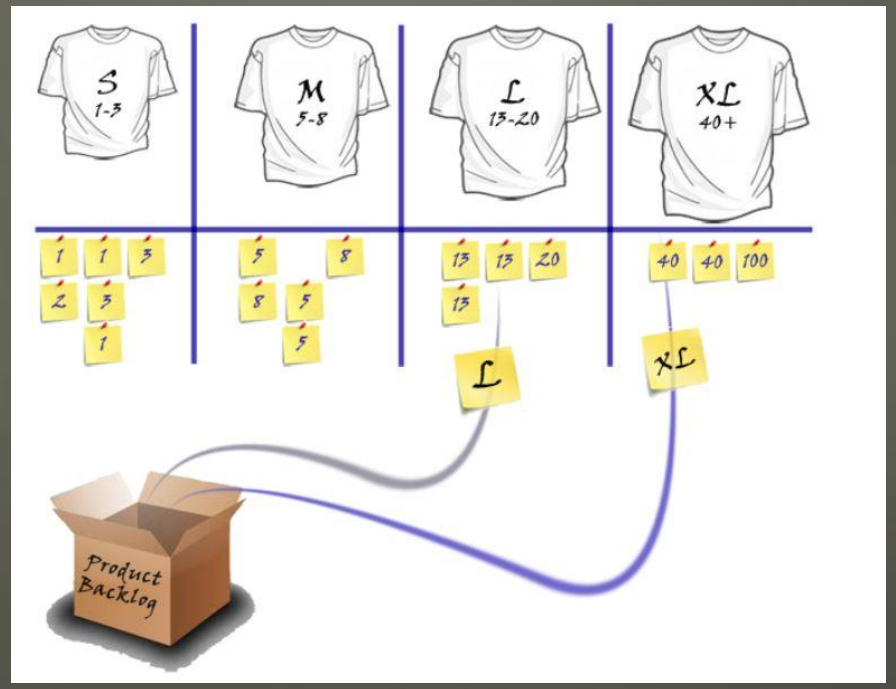
- Complexity
- Uncertainty
- Effort

Relative Sizing

Estimation Techniques

- Story Points
- Modified Fibonacci Series
- Poker Planning
- T-Shirt Sizing
- Ideal Days
- Bucket System
- Affinity Mapping
- Big/Uncertain/ Small





Velocity & Release Planning

- **Velocity** is a measure of the amount of work a Team can tackle during a single Sprint and is the key metric in Scrum.
- Average number [Preferably Last 3 Sprints] of Story Points Delivered in a Sprint/Iteration
- Velocity is a key feedback mechanism for the Team. It helps them measure whether process changes they make are improving their productivity or hurting it. While a Team's velocity will oscillate from Sprint to Sprint, over time, a well-functioning Scrum Team's velocity should steadily trend upward by roughly 10% each Sprint.
- Without Velocity, Release Planning is impossible. By knowing Velocity, a Product Owner can figure out how many Sprints it will take the Team to achieve a desired level of functionality that can then be shipped. Depending on the length of the Sprint, the Product owner can fix a date for the release.

For most agile development teams velocity will typically stabilize between 3 and 6 iterations.

User Stories

- **User stories** are short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system. They typically follow a simple template:
  - *As a < type of user >, I want < some goal > so that < some reason >.*
- Who writes user stories
  - Anyone can write user stories. It's the product owner's responsibility to make sure a product backlog of agile user stories exists, but that doesn't mean that the product owner is the one who writes them. Over the course of a good agile project, you should expect to have user story examples written by each team member.

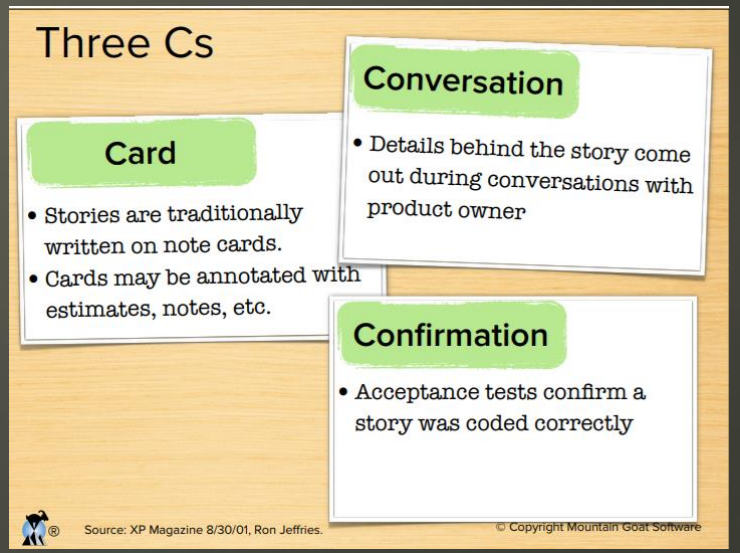
➢ **What is an Acceptance Criteria?**

An acceptance criterion is a set of accepted conditions or business rules which the functionality or feature should satisfy and meet, in order to be accepted by the Product Owner/Stakeholders.

➢ **What are Acceptance Criteria Used For**?

- ✓ To define boundaries
- ✓ To reach consensus.
- ✓ To serve as a basis for tests.
- ✓ To allow for accurate planning and estimation. .

User Stories

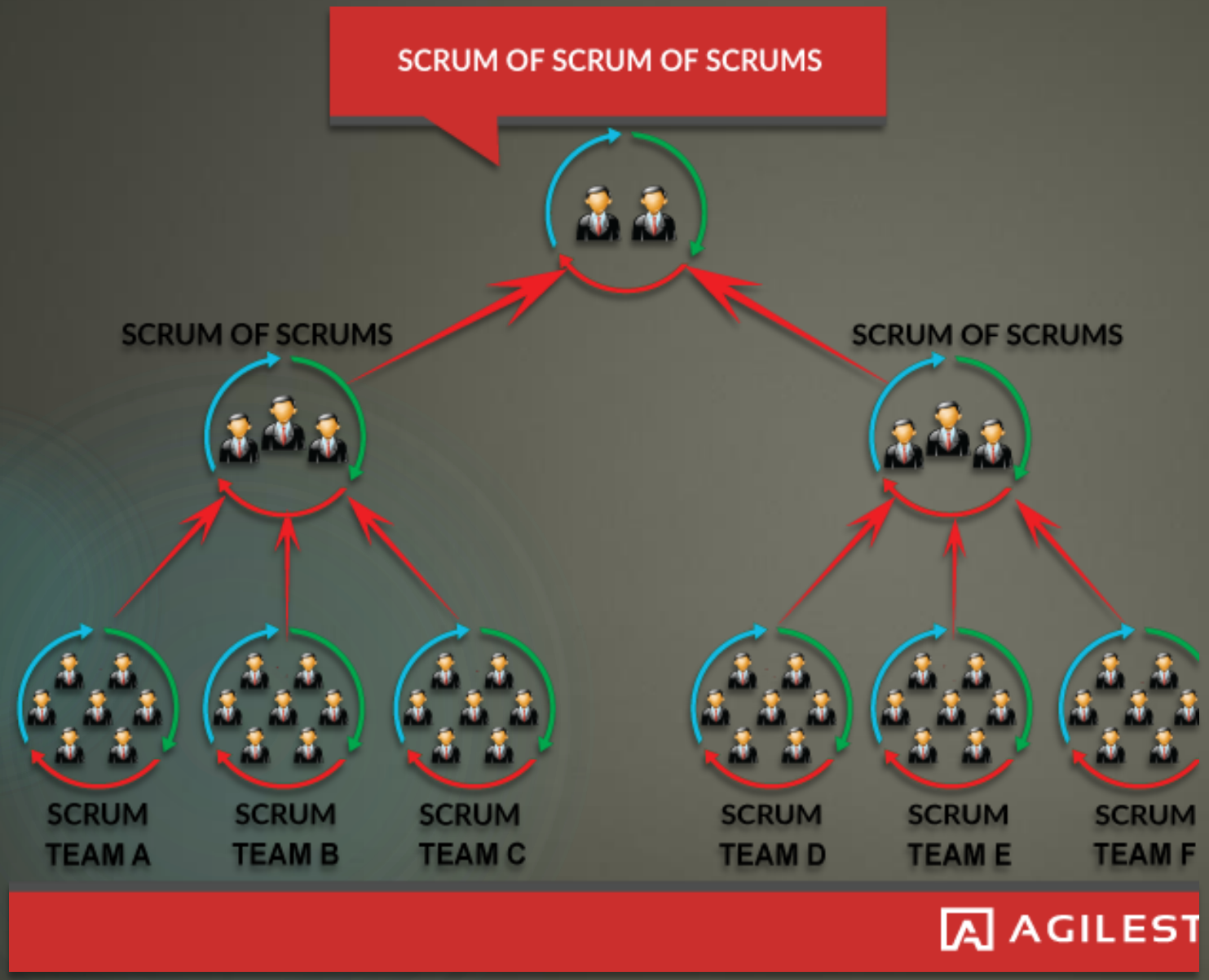


INVEST

The acronym INVEST helps to remember a widely accepted set of criteria, or checklist, to assess the quality of a user story. If the story fails to meet one of these criteria, the team may want to reword it, or even consider a rewrite (which often translates into physically tearing up the old story card and writing a new one).

A good user story should be:

- "I" ndependent (of all others)
- "N" egotiable (not a specific contract for features)
- "V" aluable (or vertical)
- "E" stimable (to a good approximation)
- "S" mall (so as to fit within an iteration)
- "T" estable (in principle, even if there isn't a test for it yet)



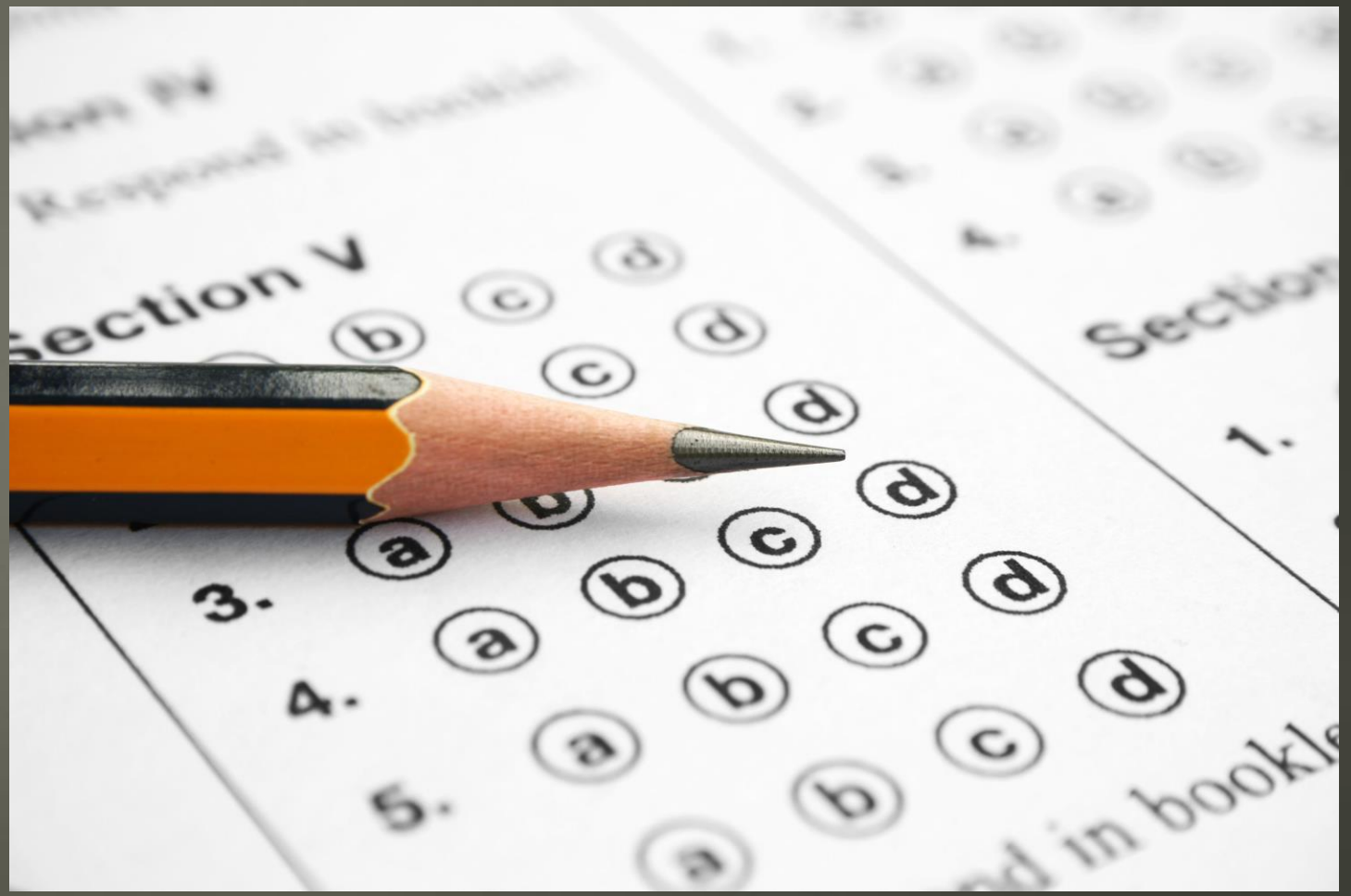
Scrum of Scrums


Any Questions?

Open Assessments

https://www.scrum.org/open-assessments/scrum-open

Score 100 % in the Open Assessments that's when you are ready for the Real Test



Level 2

- Kanban – 2 hours
- Scrumban- 1 Hour
- SAFe Overview- 2 hours
- Agile Transformation Challenges- 2 hours
- Anti Agile Patterns- 6 hours
- Scrum Master Challenges- 2 hours
- Effective Facilitation Techniques- 1 hours
- JIRA or any other Tools- 1 hour
- Resume Prep- Offline
- Interview Questions- PDF's to be Sent
- Mock interviews- 2 hours per person
- Interview Support and Retrospectives – On Need basis

Kanban

- https://www.digite.com/kanban/what-is-kanban/#origin-of-kanban
- https://youtu.be/Oux-frfCLCo
- https://leankanban.com/project/ten-things/
- https://kanbanize.com/kanban-resources/getting-started/what-is-kanban/

- 1940- Taiichi Ohno [Toyota]- to Reduce the inventory waste
- 2004- David J. Anderson who was the first to apply the concept to IT, Software development and knowledge work in general in the year 2004
- 2010- First Book on Kanban was Published

"Kanban is not a software development lifecycle methodology or an approach to project management. It requires that some process is already in place so that Kanban can be applied to incrementally change the underlying process." – David J. Anderson

Kanban for maximizing the benefits to your business process – improve flow, reduce cycle time, increase value to the customer, with greater predictability – all of which are crucial to any business today.

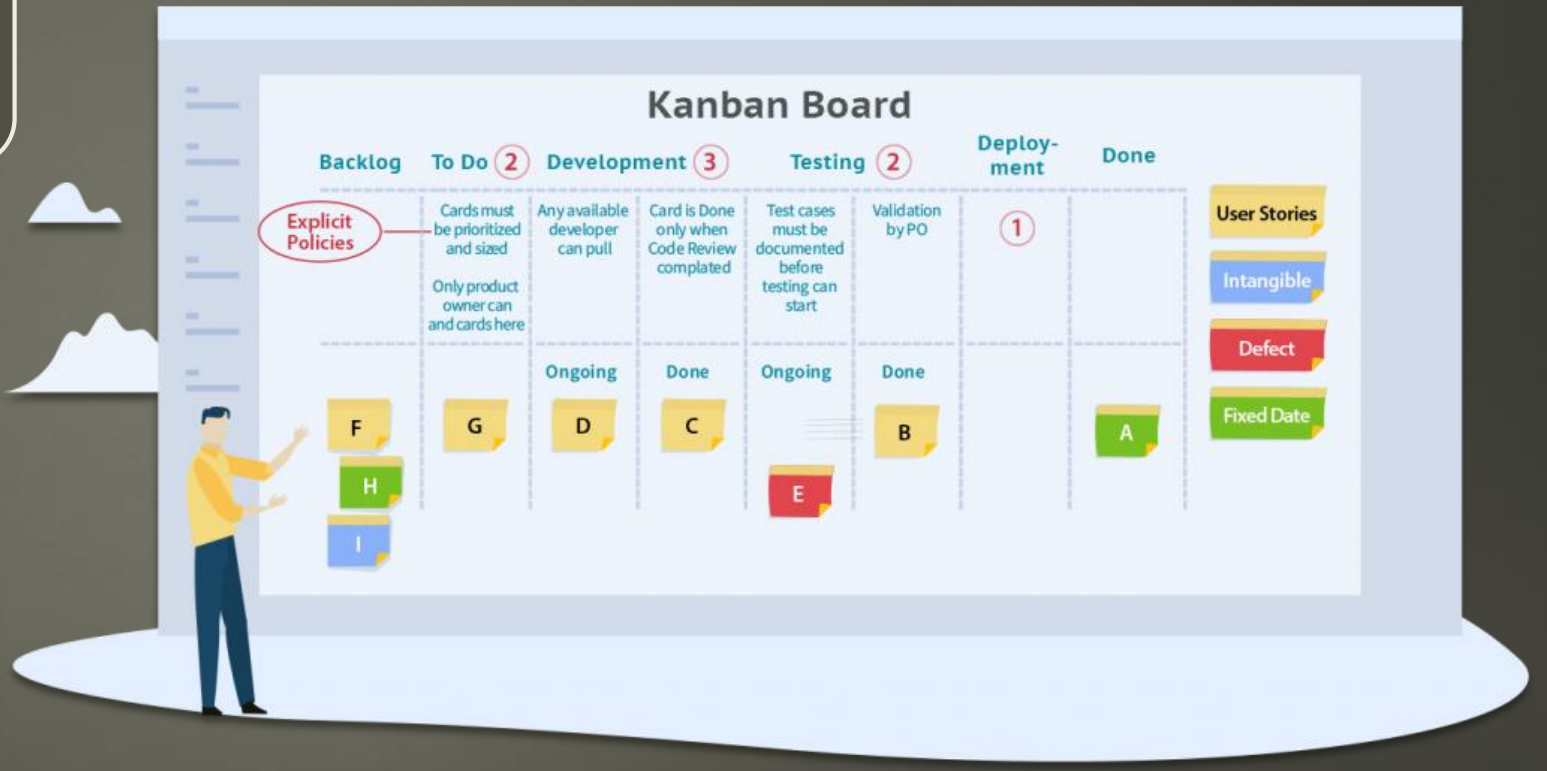
Kanban Principles

- Foundational Principles
  - **Start with what you are doing now:** Kanban strongly emphasizes not making any change to your existing setup/ process right away. Kanban must be applied directly to current workflow. Any changes needed can occur gradually over a period of time at a pace the team is comfortable with.
  - **Agree to pursue incremental, evolutionary change:** Kanban encourages you to make small incremental changes rather than making radical changes that might lead to resistance within the team and organization.
  - **Initially, respect current roles, responsibilities and job-titles:** Unlike other methods, Kanban does not impose any organizational changes by itself. So, it is not necessary to make changes to your existing roles and functions which may be performing well. The team will collaboratively identify and implement any changes needed. These three principles help the organizations overcome the typical emotional resistance and the fear of change that usually accompany any change initiatives in an organization.
  - **Encourage acts of leadership at all levels:** Kanban encourages continuous improvement at all the levels of the organization and it says that leadership acts don't have to originate from senior managers only. People at all levels can provide ideas and show leadership to implement changes to continually improve the way they deliver their products and services.
- *"Asking people to change behavior is difficult!" – David J. Anderson*

Kanban Practices

6 Core Practices of the Kanban Method

1. Visualize the flow of work
2. Limit WIP (Work in Progress)
3. Manage Flow
4. Make Process Policies Explicit
5. Implement Feedback Loops
6. Improve Collaboratively, Evolve Experimentally (using the scientific method)



Kanban Practices

Visualize the flow of work

- You need to visualize – either on a physical board or an electronic Kanban Board
- Once you visualize your process, then you can visualize the current work that you and your team are doing
- Kanban board can have different Swim Lanes, one for each class of service or for each work item type

Limit WIP (Work in Progress)

- Limiting work-in-progress (WIP) is fundamental to implementing Kanban – a 'Pull-system'.
- By limiting WIP, you encourage your team to complete work at hand first before taking up new work.
- Deciding on WIP Limits

Manage Flow

- Managing and improving flow is the crux of your Kanban system. It helps you manage flow by highlighting the various stages of the workflow and the status of work in each stage.
- Depending on how well the workflow is defined and WIP Limits are set, you will observe either a smooth flow within WIP limits or work piling up as something gets held up and starts to hold up capacity.
- Kanban helps your team analyze the system and make adjustments to improve flow so as to reduce the time it takes to complete each piece of work.

Kanban Practices

Make Process Policies Explicit

- As part of visualizing your process, it makes sense to also define and visualize explicitly, your policies (process rules or guidelines) for how you do the work you do. By formulating explicit process guidelines, you create a common basis for all participants to understand how to do any type of work in the system.
- The policies can be at the board level, at a swim lane level and for each column. They can be a checklist of steps to be done for each work item-type, entry-exit criteria for each column, or anything at all that helps team members manage the flow of work on the board well.
- The policies must be defined explicitly and visualized usually on the top of the board and on each lane and column.

Implement Feedback Loops

- The Kanban Method encourages and helps you implement feedback loops of various kinds – review stages in your Kanban board workflow, metrics and reports and a range of visual cues that provide you continuous feedback on work progress – or the lack of it – in your system

Improve Collaboratively, Evolve Experimentally (using the scientific method)

- As a team implementing Lean/ Agile principles, your key task is to evaluate your process constantly and improve continuously as needed and as possible.
- Metrics will hep you to evaluate your performance and tweak your system as needed

Kanban Cont..

Kanban in IT

- Kanban is not a software development or a project management methodology – David makes that very clear in his 'Blue Book'. Kanban does not say anything about how a Software should be developed. It does not even say anything about how Software projects should be planned and implemented. Therefore, Kanban is not a management framework such as Scrum. Instead, the purpose of Kanban is to continually improve one's own work process.

Kanban in Lean/ Agile software/ product development

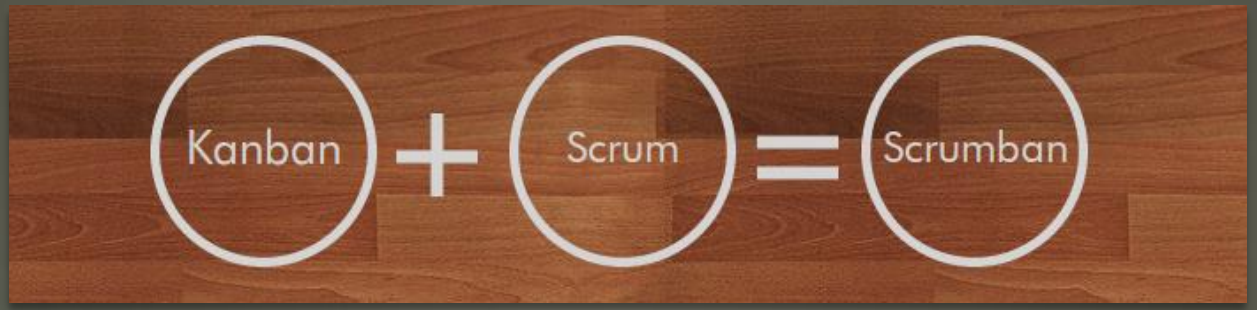
- The Kanban Method provides technology teams a great set of principles for visualizing their work, delivering products and services continuously and getting customer feedback more often and with greater speed. Consequently, it is helping teams get to market faster with greater fidelity to what the customers want from those products and services.

Kanban as an Alternative Path to Enterprise Agility

- The Kanban Method helps you gradually improve the delivery of your products and services. It does so by helping you eliminate bottlenecks in your system, improve flow and reduce cycle time. It helps you deliver more continuously and get faster feedback to make any changes that may be needed by your customer. It helps you become more responsive.

ScrumBan

- ✓ https://leankit.com/learn/agile/what-is-scrumban/
- ✓ https://www.agilealliance.org/what-is-scrumban/



- Scrumban = Scrum + Kanban
- Scrumban is an Agile development methodology that is a hybrid of Scrum and Kanban
- Scrum is best-suited for products and development projects. Kanban is best for production support. We use Scrumban – which combines the best features of both – for maintenance projects. Scrumban is becoming very popular these days in service industries, where we have both development and maintenance projects.
- Many teams use Scrumban as a transition point between a less mature and more mature Agile practice.

Scrumban

Advantages

Quality	Just-in-time (decisions and facts just when they are needed)	Short lead time	Kaizen (continuous improvement)	Minimizing waste (everything that is not adding value to the customer)	Process improvement by adding some values of Scrum as and when needed

When to consider Scrumban

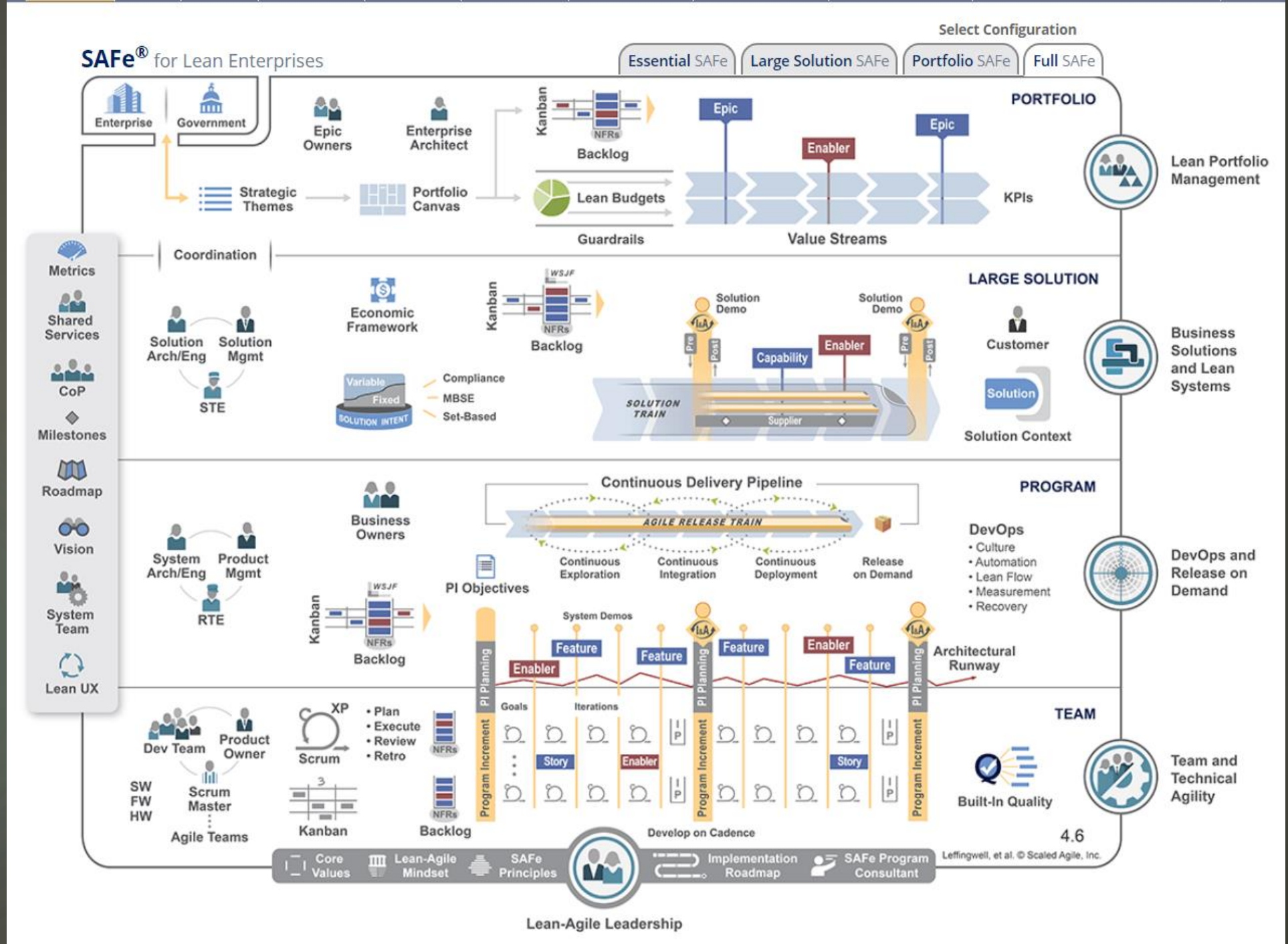
Maintenance projects →Event-driven work →Help desk/support	Hardening/packaging phases	Projects with frequent and unexpected user stories or programming errors	Sprint teams focused on new product development •Work preceding sprint development (backlog, R&D) •Work following sprint development (system testing, packaging, and deployment)	If Scrum is challenged by workflow issues, resources and processes	To manage improvement communities during/after Scrum roll-out

ScrumBan

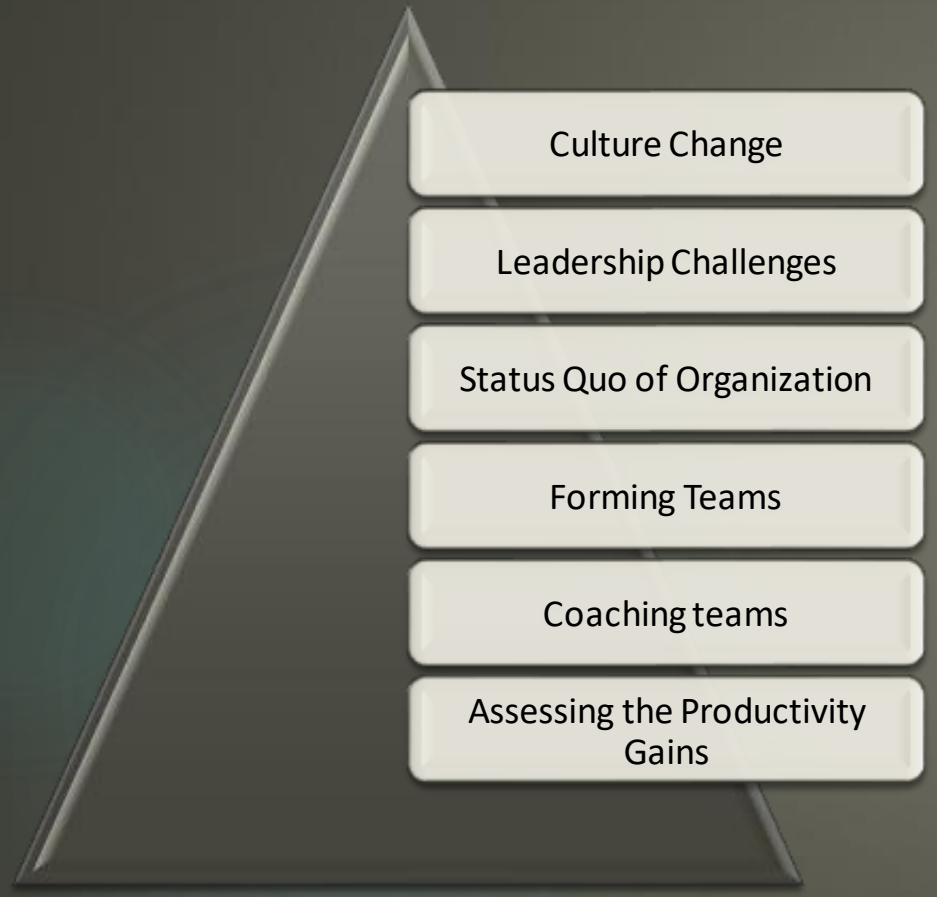
	Kanban	Scrumban
Roles	No prescribed role	Team + needed roles
Daily scrum meeting	No meeting	To make sure continuous work on requirements and reduce idle time of team members.
		Can be done to plan to fill the slots
Review and Retrospective Meeting	Not prescribed.	Can be done as needed for process improvement and share learnings.
Work Flow	Continuous	Same as in Kanban. Just add limit of slots so that pull process will become more comfortable.

	Scrum	Scrumban
Board / Artifacts	board, backlogs, burn-downs	board only
Ceremonies	daily scrum, sprint planning, sprint review, sprint retrospective	daily scrum (planning, review and retrospective as needed)
Iterations	yes (sprints)	no (continuous flow)
Estimation	yes	no (similar size)
Teams	must be cross-functional	can be specialized
Roles	Product Owner, Scrum Master, Team	Team + needed roles
Teamwork	collaborative as needed by task	swarming to achieve goals
WIP	controlled by sprint content	controlled by workflow state
Changes	should wait for the next sprint	added as needed on the board (to do)
Product Backlog	list of prioritized and estimated stories	just in time cards
Impediments	dealt with immediately	avoided

SAFe Overview



Agile Transformation Challenges



It is not the strongest of the species that survives, nor the most intelligent. But it is the one who is most adaptable to change" - Charles Darwin

Agile Transformation Challenges

- Culture Change:
  - Understanding the needs of Agile Transformation
  - Understanding the Roles and Responsibilities
  - Business alignment
  - Agile Mindset
  - Setting up a Road map/ Goals

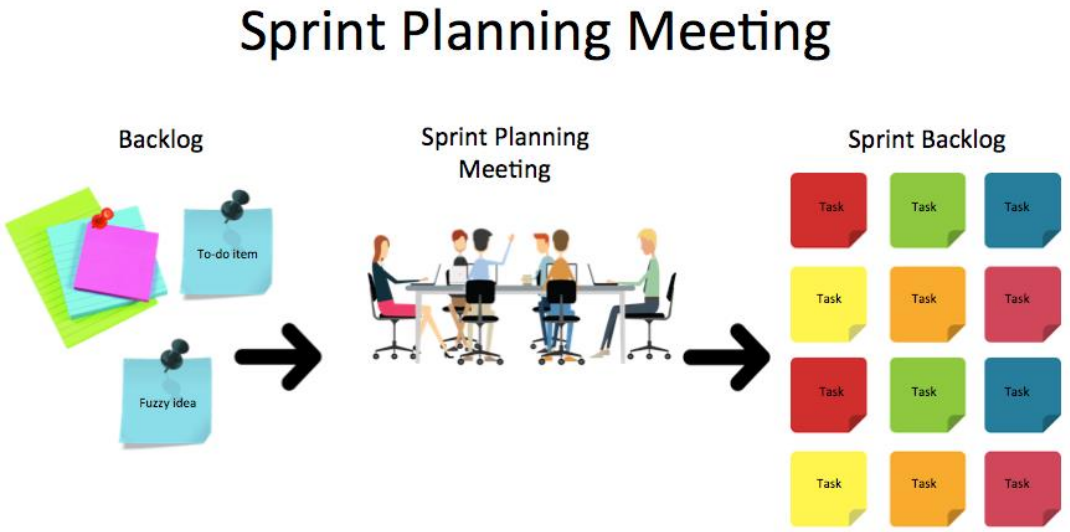
- Leadership Challenges
  - Transformation into an Agile team should first happen at an organization level that they can be cascaded down to the team level.
  - It is the senior leadership's responsibility (owners and CXO's) to embrace agility, and then to create the corporate values, beliefs, behaviors, and practices necessary to activate this culture.
  - Scrum masters as the Agile leadership of the organization must understand this corporate culture and then adapt the same to teams as necessary.

Agile Transformation Challenges

- Status Quo:
  - Roles and Responsibilities
  - Hierarchy
  - Operational Model
- Teams
  - Forming Teams:
    - Component Teams, Feature Teams
  - Coaching Teams
    - Training and Coaching them
- Productivity Gains
  - How do we measure Productivity?
  - Short Term reduction is productivity is acceptable
  - Negotiating with stakeholders

Anti Agile Patterns/ Challenges/Best Practices

- Sprints
  - Not understanding the purpose of Sprints
  - Classification of sprints
    - Dev Sprint, QA Sprint, Hardening Sprint
  - Sprint Cadence
    - Extending the sprint length
- Sprint Planning
  - Estimations
    - Story Points mapping to hours
    - Influencing Estimates
  - Capacity Utilization
    - Not Considering the ad-hoc requests
    - Nature of the Project to be considered
  - Classification of Dev and QA User stories
  - Estimating Tasks
    - Over Estimating / Under Estimating/ Not breaking down the User stories into tasks
  - Sprint Goal
    - Sprint Goal should be defined and understood by the Team



Anti Agile Patterns/ Challenges/ Best Practices

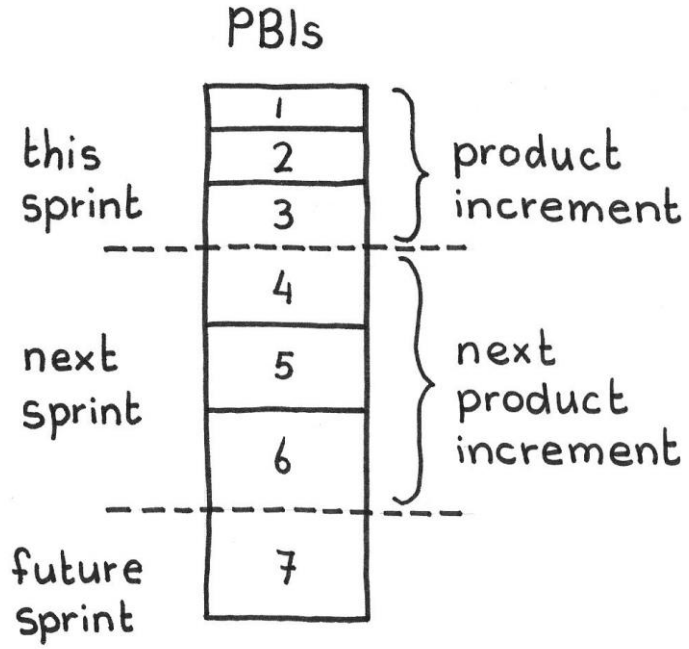
- Daily Scrum
  - Status Meeting
  - Presence of others and their influence
  - Time Boxing the event
  - Updating Tasks in Daily Scrum (not a good practice)
  - Not looking at a broader perspective
  - Risks not highlighted.
- Sprint Review
  - Demo to the Product Owner
  - Not inviting the Right Stakeholders
  - Conflict of Interest between PO and Stakeholders
  - Feedback not formalized
  - Stakeholders not attending the Sprint Reviews
    - Concerns raised after the Sprint Review
  - Release plan/vision/risks not discussed.
    - Negotiation of Scope/ Time
  - Cadence of Sprint Reviews
  - Involve whole team

Anti Agile Patterns/ Challenges/Best Practices

- Sprint Retrospective
  - Cadence of Sprint Retrospectives
  - Purpose of Retrospective
  - Team not opening up
  - Blame Game
  - Discussing the progress of Action Items
  - Tracking Action Items to Closure
  - Restrict it only to Scrum Team(PO, SM, DEV)
  - Teams not showing up interest
- Hardening
  - Hardening Sprint
  - Hardening Patterns in a Sprint
  - Patterns and Mitigation Techniques
  - Reasons for Hardening
  - Addressing the Technical Debt

Anti Agile Patterns/ Challenges/Best Practices

- Product Backlog Refinement
  - Event Vs Activity
  - Affect of not having the PBR
  - Time allocated, defining the DOR
  - Backlog Readiness
- User Story
  - INVEST
  - Dev Story Vs Test Stories
  - Minimizing Dependencies
  - Breaking down User Stories into tasks
  - Ownership and Format of Writing Stories
  - Acceptance Criteria
  - Business Value
  - End User Perspective



I ndependent

N egotiable

V aluable

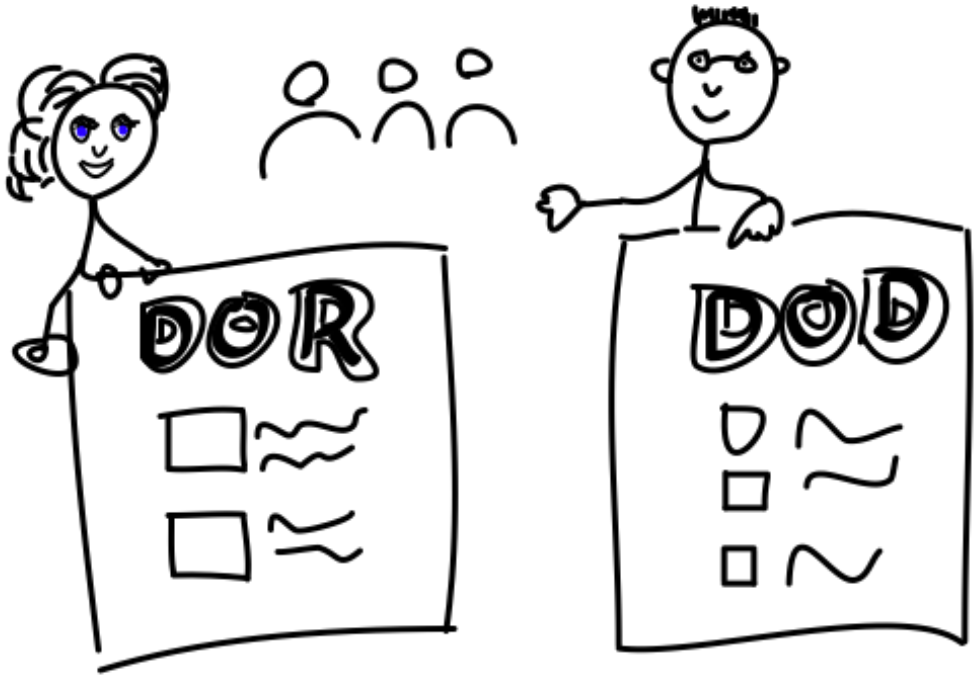
E stimable

S mall (Sized appropriately)

T estable

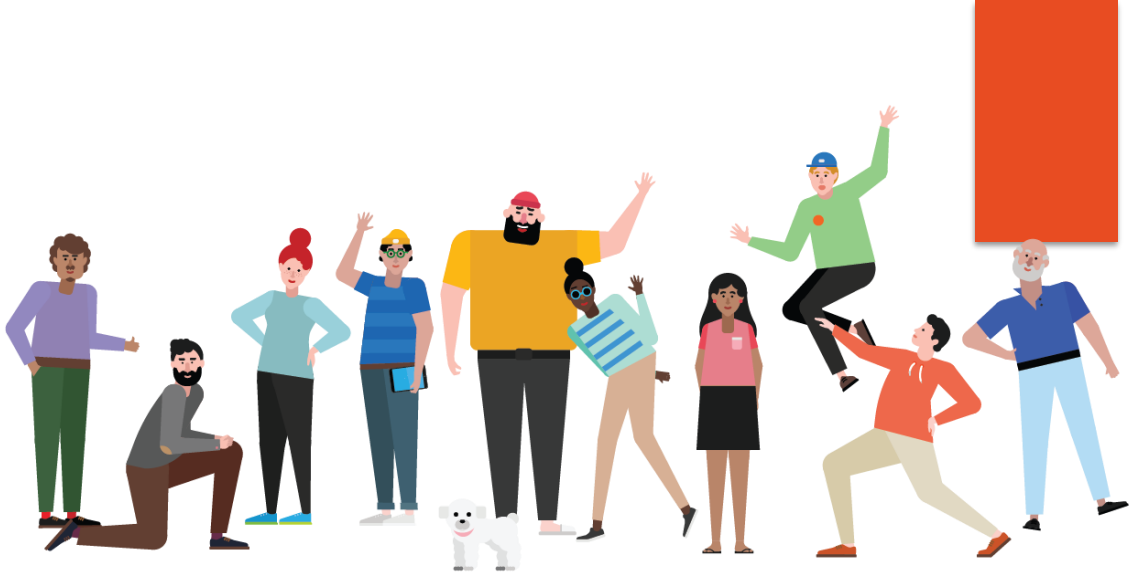
Anti Agile Patterns/ Challenges/Best Practices

- Definition of Ready and Definition of Done
  - Who Defines?
  - Why do we need this?
  - When should these be created and revisited
  - Suggested Formats
  - Affects of not adhering to DOR/DOD
  - Is your DOR a Blocker/ enabler
  - Can you compromise on adhering to DOD
  - How are Acceptance criteria and DOD different
  - Multiple Teams working on Same backlog how do we define DOD
  - DOD= Quality ?



Scrum Master Challenges

- Setting up Teams
  - ✓ Component Teams
  - ✓ Feature Teams
  - ✓ Right mixture of expertise
  - ✓ Dependent Teams
  - ✓ Sprint Cadence (Meetings , Falling on Same Day)
  - ✓ Building Cross functional Teams
- Process
  - ✓ Bug Triaging
  - ✓ Code Ownership/ Sorting out Dependencies/ RACI
  - ✓ Tracking Action Items to Closure
  - ✓ Ownership for Impediments
  - ✓ Reports/ Patterns/ Mitigation Plans



Scrum Master Challenges

Conflict Management

- PO
  - Backlog Readiness
  - Bugs/Issues
  - DOD
  - Pressure from PO
- DEV Team
  - Building Trust
  - Stubborn heads
  - Over/ under Estimations
  - Adherence to Process
  - Breaking the Status Quo
  - Internal Team Conflicts/ with Managers/ external Teams
- Leadership
  - Support of Leadership
  - Culture Change
  - Reports/ Productivity/capacity utilization/ Velocity

Scrum Master Challenges

- **Motivating Teams**
  - Adhering to Scrum Values
  - How do Scrum Master ensure Transparency in Teams
  - Scrum Teams interaction with Stakeholders
  - Building Enthusiasm
- **Release Planning**
  - Facilitating the Release planning
  - Identifying the Dependencies
  - Challenges/ Impediments
  - Scrum of Scrums
  - Release retrospective
  - Convincing the Senior Leadership
  - Metrics

Effective Facilitation Techniques

https://www.scrum.org/resources/blog/scrum-master-master-art-facilitation?gclid=CjwKCAjwmq3kBRB_EiwAJkNDp174NCCqI0GOJwA_2-CkjtOZwJ34ynCtNQW_y5spuh2guh8JmYvy6hoCH0AQAvD_BwE

https://resources.collab.net/blogs/twelve-awesome-interactive-facilitation-techniques-for-agile-teams

- Sprint Planning
  - Poker Planning for Estimations
  - Use Online Tools if the teams are distributed
- Daily Scrum
  - Time Keeper
  - Active Listener
- Sprint Review
  - Feedback tools
- Sprint Retrospectives
  - Dot Voting
  - Games

Effective Facilitation

Tools


VERSIONONE
Agile Made Easier


ca
technologies
CA Agile Central


Microsoft
Visual Studio
Team Foundation Server


Atlassian
JIRA


Thank you!