

Welcome!



# Certified ScrumMaster Workshop

Facilitated by Bonsy Yelsangi

Email: [bonsy.yelsangi@bonsyinc.com](mailto:bonsy.yelsangi@bonsyinc.com)

LinkedIn: Bonsy Yelsangi

Twitter: @BonsyYelsangi



# WELCOME TO SCRUM ALLIANCE®

**Congratulations on completing the Certified ScrumMaster® (CSM®) course and taking the next step on your Agile journey.**

.....



## Next: Pass The Test To Earn Your CSM Certification

1. Watch for your official welcome email from Scrum Alliance.
2. Log in to your Scrum Alliance dashboard at [scrumalliance.org](https://scrumalliance.org) to access the CSM test. Your dashboard is also where you review your profile, manage your Scrum Education Units® (SEUs), and (if you're a first-time certificant) claim your complimentary membership.
3. Pass the CSM test by correctly answering at least 37 of the 50 questions within 60 minutes. You have 90 days and two free attempts to pass the test. Please see details online.
4. Once you pass the test, go back into your dashboard to accept the License Agreement.

## You Earned Your Certification. Now Maintain It!

Your certification is valid for two years. We'll send reminders as your renewal date approaches. **To maintain your certification, you're required to complete 20 Scrum Education Units® (SEUs), along with \$100 renewal fee every two years.** It's easy to earn SEUs and by doing so, you'll stay relevant and competitive. **Learn more about SEUs here:** [scrumalliance.org/get-certified/scrum-education-units](https://scrumalliance.org/get-certified/scrum-education-units)

## Keep Going >>>

Meaningful training helps you stay current with best practices. Plan the next step in your Agile journey at [scrumalliance.org/get-certified](https://scrumalliance.org/get-certified).

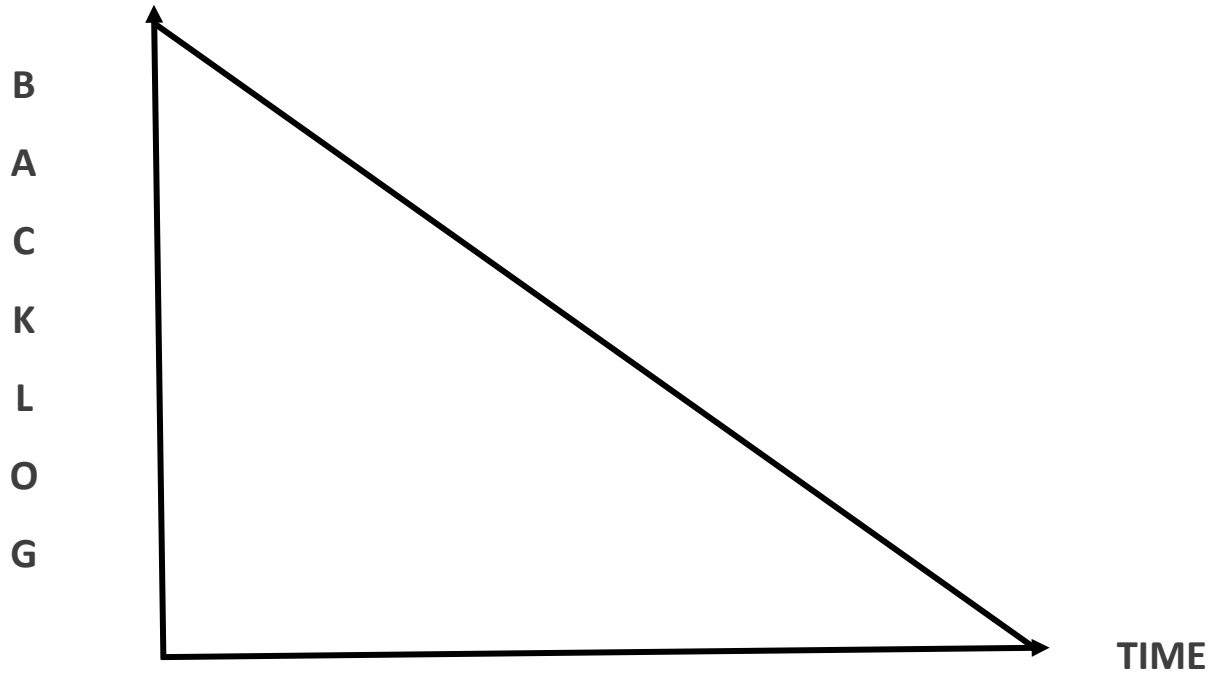
Again, congratulations and welcome to Scrum Alliance, the world's largest association of Agile practitioners and Scrum certification body. Our nonprofit organization will be here to support you along your Agile journey with an engaged community of practitioners, avenues to further your knowledge, and ongoing encouragement. Thank you for helping transform the world of work.

Version 1/2019

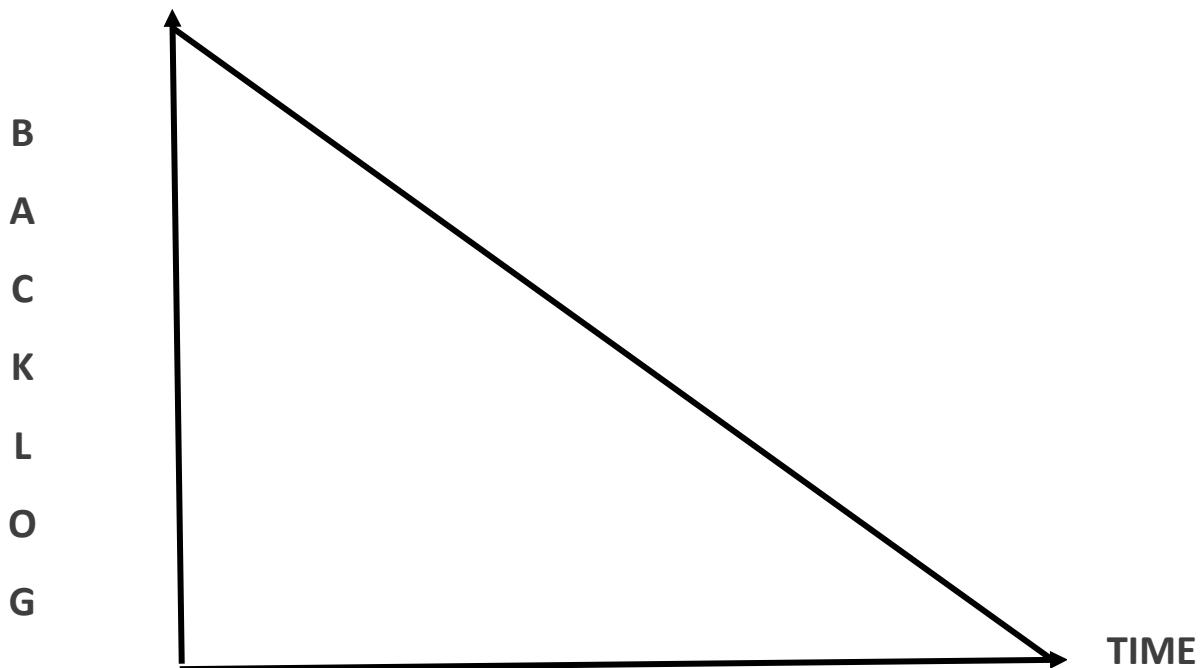
[scrumalliance.org](https://scrumalliance.org)



### BURNDOWN CHART – DAY 1



### BURNDOWN CHART – DAY 2



# The Agile Manifesto

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

\_\_\_\_\_ over \_\_\_\_\_

\_\_\_\_\_ over \_\_\_\_\_

\_\_\_\_\_ over \_\_\_\_\_

\_\_\_\_\_ over \_\_\_\_\_

That is, while there is value in the items on the right, we value the items on the left more.

---

Using the choices below, fill in the blanks above:

- Customer collaboration
- processes and tools
- following a plan
- Working software
- comprehensive documentation
- Responding to change
- Individuals and interactions
- contract negotiation

# Principles behind the Agile Manifesto

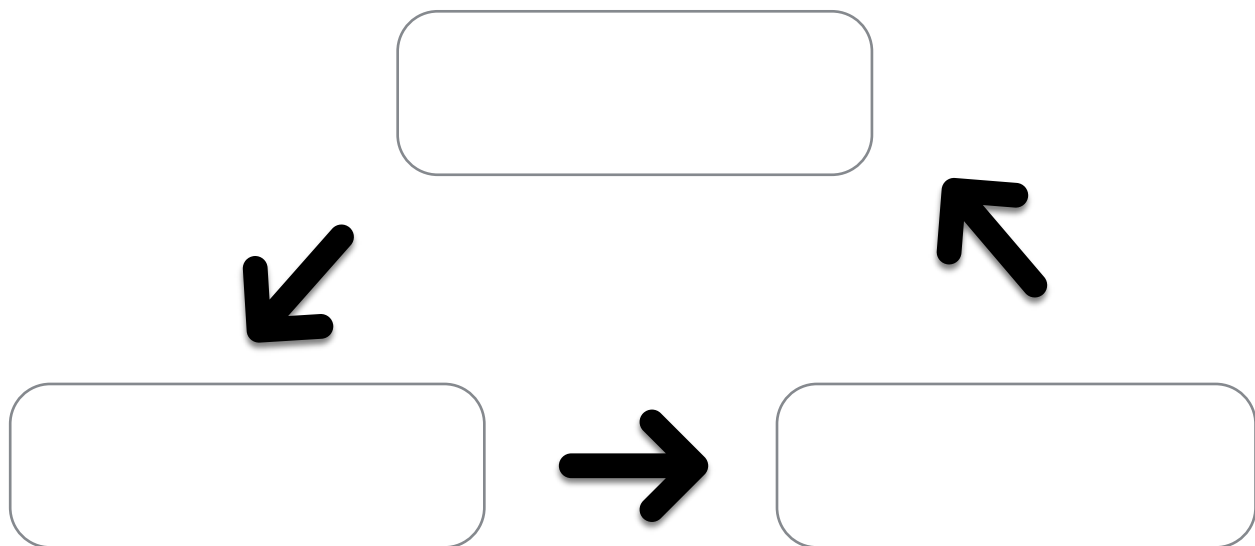
We follow these principles:

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.
5. Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.
6. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
7. Working software is the primary measure of progress.
8. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
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.

# 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 founded on empirical process control theory, or empiricism. Empiricism asserts that knowledge comes from experience and making decisions based on what is known.**



**Risk**



**Probability of  
Success**



# Scrum Overview

## 3 Pillars of Scrum are:

1. \_\_\_\_\_
2. \_\_\_\_\_
3. \_\_\_\_\_

## 3 Roles are:

1. \_\_\_\_\_
2. \_\_\_\_\_
3. \_\_\_\_\_

## 5 Inspect and Adapt Events are:

1. \_\_\_\_\_
2. \_\_\_\_\_
3. \_\_\_\_\_
4. \_\_\_\_\_
5. \_\_\_\_\_

## 5 Scrum Values are:

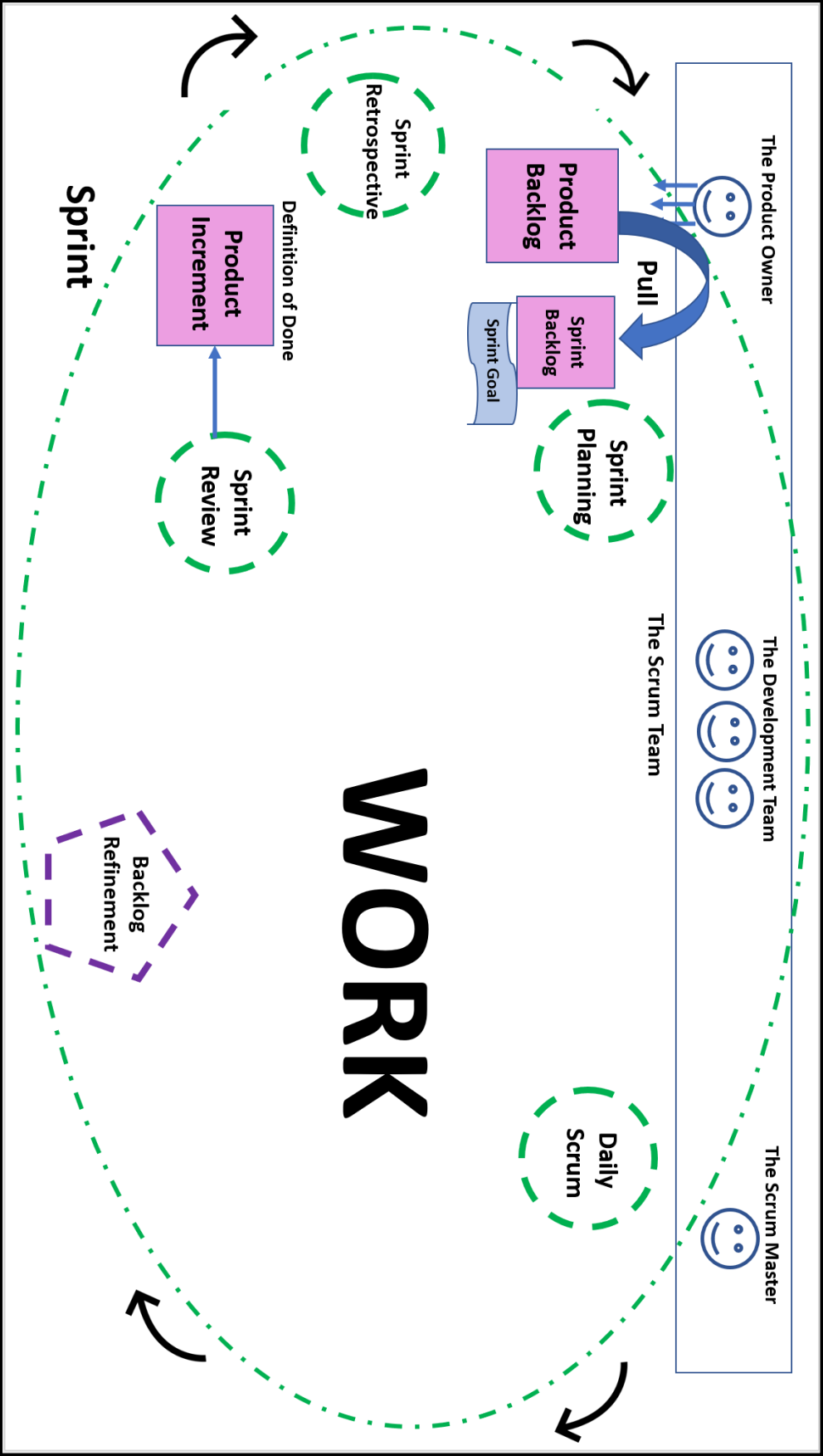
1. \_\_\_\_\_
2. \_\_\_\_\_
3. \_\_\_\_\_
4. \_\_\_\_\_
5. \_\_\_\_\_

## 3 Artifacts are:

1. \_\_\_\_\_
2. \_\_\_\_\_
3. \_\_\_\_\_

## 1 Inspect and Adapt Activity is:

\_\_\_\_\_





# Scrum Values

**Courage**

**Focus**

**Commitment**

**Respect**

**Openness**

## In Summary... *THE VALUES OF SCRUM*

- \* The values of Scrum enable the three pillars of Scrum to come to life.
- \* Through teamwork and continuous improvement, Scrum both supports these values and relies on them.
- \* The elements of Scrum - the parts of the framework, the roles, the artifacts and the events - are the way we put the Scrum framework into practice.
- \* The values of Scrum are a diagnostic tool that helps you improve your team and your application of Scrum.
- \* Only add practices, policies and procedures that support the values of Scrum.
- \* The Agile Manifesto describes the values and principles that form a common basis for Scrum and other methods. Agile values and principles apply directly to Scrum.

## Scrum Overview Wrap Up

**I learned:**

**Earlier I thought.....now I know:**

**I want to know more about:**

# Product Backlog

- Ordered list of ideas, outcomes, work to do, supports Product Vision
- Items collaboratively created by PO and Development Team
- Product backlogs are often too long and detailed, and therefore difficult to use. Part of the solution is to ensure that your product backlog is **DEEP**:

**D** \_\_\_\_\_ appropriately (more granular near top)

**E** \_\_\_\_\_ (items may be added, removed, reordered and refined over time)

**E** \_\_\_\_\_ by \_\_\_\_\_

Ordered by \_\_\_\_\_

**P** \_\_\_\_\_ prior to Sprint Planning

## Fill in the blanks with these selections:

The Development Team

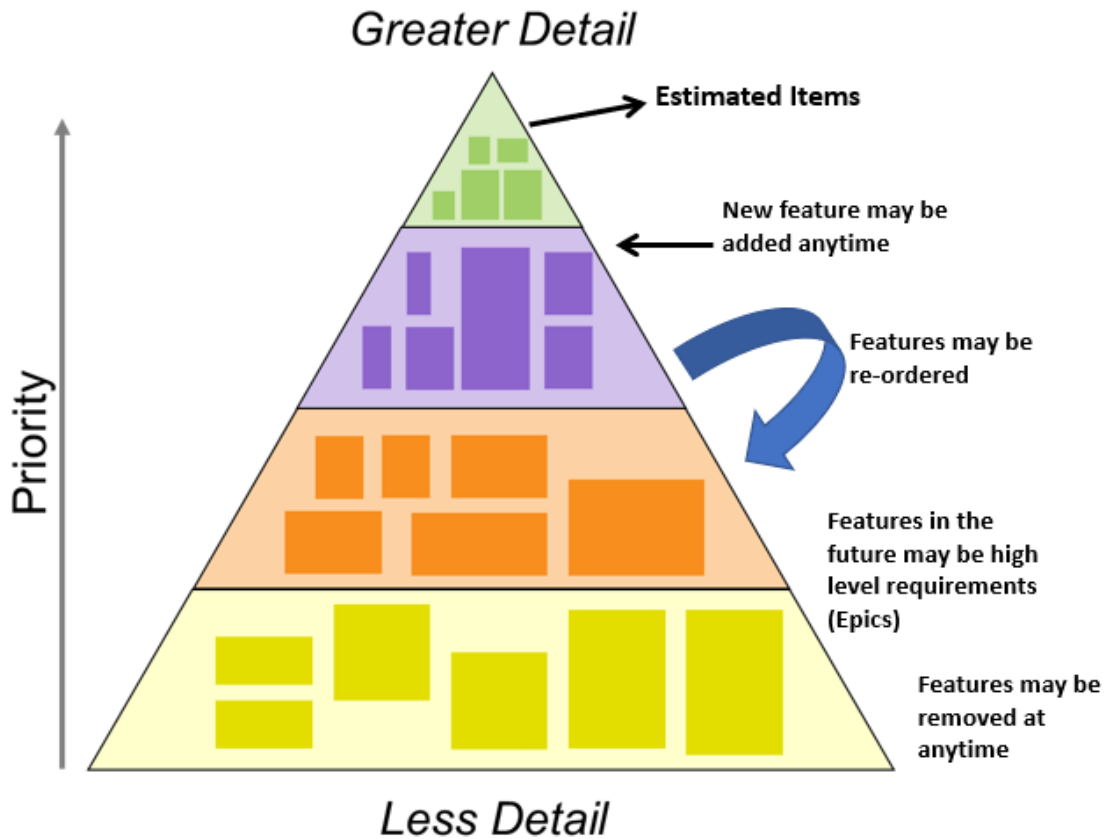
Product Owner

Prioritized

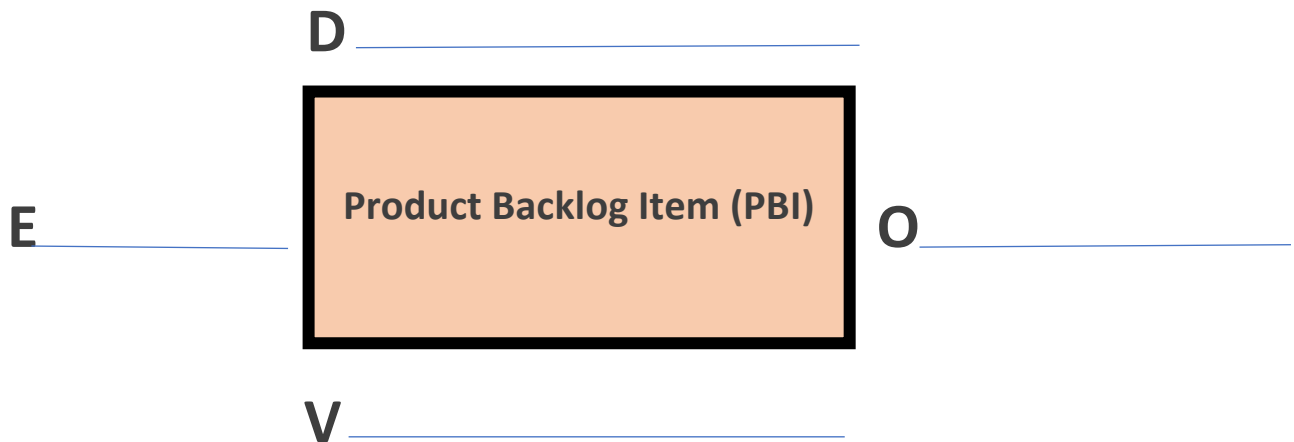
Emergent

Detailed

Estimated



### Attributes of a Product Backlog Item - DOVE:



## Prioritization Technique:

**M**

**Must have:** Non-negotiable product needs that are mandatory for the team.

**S**

**Should have:** Important initiatives that are not vital, but add significant value.

**C**

**Could have:** Nice to have initiatives that will have a small impact if left out.

**W**

**Will not have:** Initiatives that are not a priority for this specific time frame.

## In Summary.....The Product Backlog

- \* The Product Backlog is a prioritized list of the ideas, features, outcomes and deliverables that the business wants the Team to complete and any other work like fixing bugs or providing documentation. The Product Backlog is a list of the functional and non-functional requirements for the product. It represents the scope.
- \* The Product Owner is responsible for regularly prioritizing the items in the Product Backlog according to business value. No two items of the Product Backlog are allowed to have the same priority.
- \* The owner of the Product Backlog is the Product Owner. Many people can give the Product Owner advice about the items in the Product Backlog and their priority, but the Product Owner has the final decision about what items go into the Product Backlog and the priority of each item.
- \* The Product Backlog is a dynamic and essential Scrum artifact that changes with time. The items in the Product Backlog are progressively refined just-in-time.
- \* Each Product Backlog item has the attributes of description, order, estimate, and value (to the business). The description usually includes the user, the feature or outcome, and the acceptance criteria. The Product Owner identifies the description, order, and value while the Development Team provides the estimate.
- \* All the Product Backlog items are written in the language of the business and provide business value. Each Product Backlog item defines a complete, vertical slice that contains all the horizontal layers common in software.
- \* The creation and refinement of the Product Backlog is the result of a collaboration between the Product Owner and the Development Team. Both roles need to reserve time each Sprint to define, understand, discuss and decompose Product Backlog items for the current, and future, Sprints.
- \* When the Development Team, or the Product Owner, do not know how to define, decompose or estimate a Product Backlog item, they can schedule a spike. A spike is a time-boxed investigation, or exploration, of a specific question.

# Engineering Practices- Technical Debt

**Instructions: Read the article below and discuss within your group.**

You have a piece of functionality that you need to add to your system. You see two ways to do it, one is quick to do but is messy - you are sure that it will make further changes harder in the future. The other results in a cleaner design, but will take longer to put in place.

Technical Debt is a wonderful metaphor developed by Ward Cunningham to help us think about this problem. In this metaphor, doing things the quick and dirty way sets us up with a technical debt, which is similar to a financial debt. Like a financial debt, the technical debt incurs interest payments, which come in the form of the extra effort that we have to do in future development because of the quick and dirty design choice. We can choose to continue paying the interest, or we can pay down the principal by refactoring the quick and dirty design into the better design. Although it costs to pay down the principal, we gain by reduced interest payments in the future.

The metaphor also explains why it may be sensible to do the quick and dirty approach. Just as a business incurs some debt to take advantage of a market opportunity developers may incur technical debt to hit an important deadline. The all too common problem is that development organizations let their debt get out of control and spend most of their future development effort paying crippling interest payments.

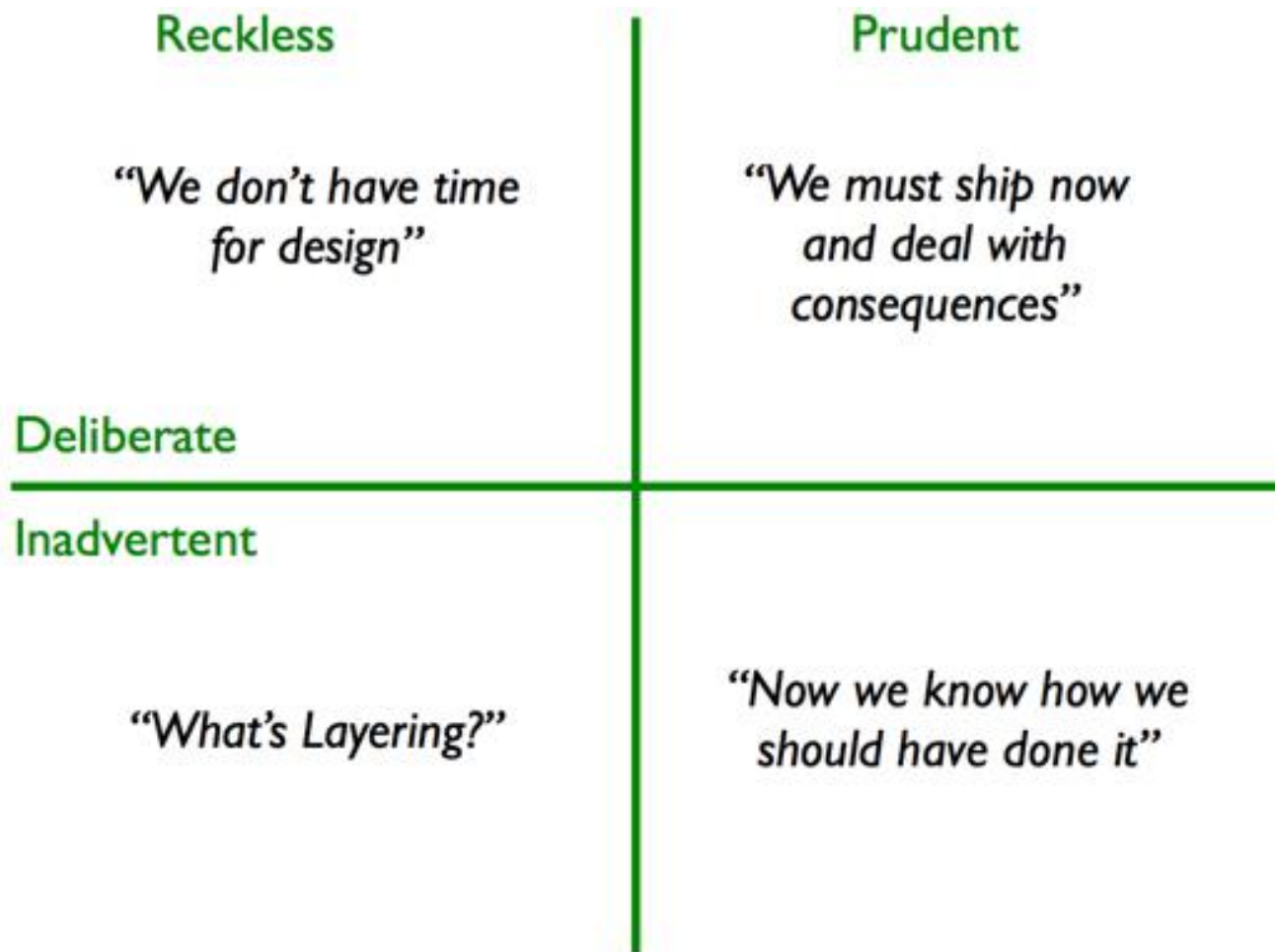


The tricky thing about technical debt, of course, is that unlike money it's impossible to measure effectively. The interest payments hurt a team's productivity, but since we cannot measure productivity, we can't really see the true effect of our technical debt.

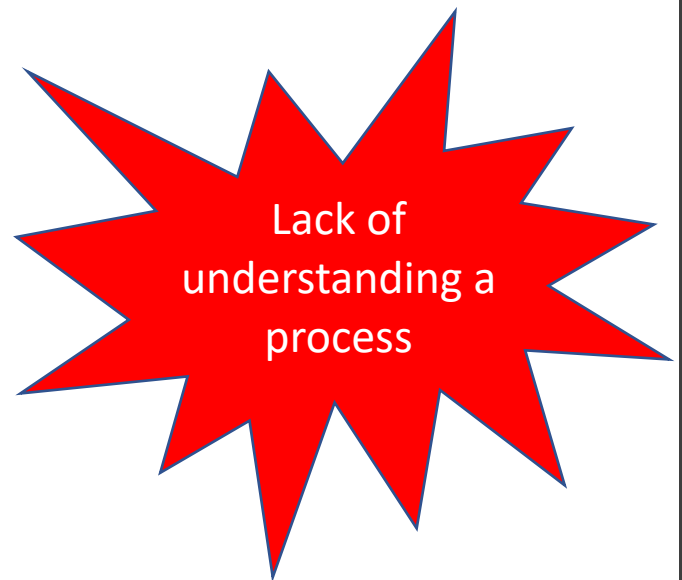
One thing that is easily missed is that you only make money on your loan by delivering. The biggest cost of technical debt is the fact that it slows your ability to deliver future features, thus handing you an opportunity cost for lost revenue. Furthermore, following the Design Stamina Hypothesis, you need to deliver before you reach the design payoff line to give you any chance of making a gain on your debt. Often taking on technical debt ends up slowing you down so much you end up delivering later.

Technical Debt comes from various sources, some of which can be good and some bad – **Martin Fowler** uses the below Technical Debt Quadrant to describe them.

(As far as I can tell, Ward first introduced this concept in an experience report for OOPSLA 1992. It has also been discussed on the wiki.)



Credit: Martin Fowler

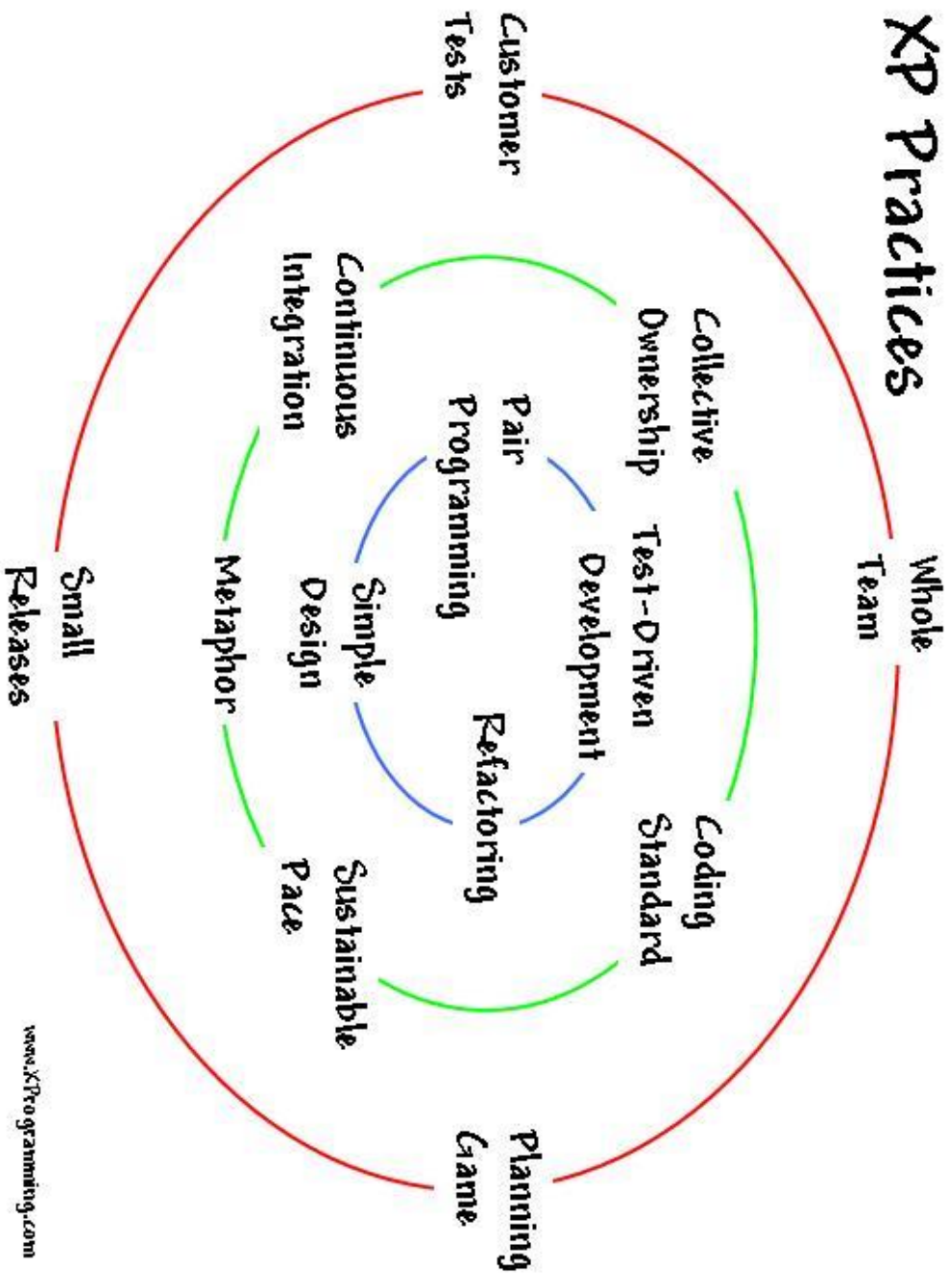


- It's the impediments that get in your way of being more productive.
- Reflects the implied cost of additional rework caused by choosing an easy solution now instead of using a better approach that would take longer.
- It is what slows your progress, something you need to constantly address because "we have to go back and fix that before we can implement this."
- Technical Debt is invisible to most stakeholders, especially those higher up in organizations.
- Executives can easily see the benefits of new features to the product and the detriment of bugs, but your architecture and technical debt are often invisible to the non-technical employees until something breaks and it's too late.

## **Impact of Accumulating Tech Debt:**

- Greater difficulty in writing new code
- Hard to find defects and fix them
- Difficulty in estimating Product Backlog Items
- Splitting larger items into smaller ones not an easy task
- Less flow within the Sprint, leading to a bulge of tasks near the end
- Less reliable planning
- Difficulty in pulling team members with the most knowledge of the code into other teams or tasks.
- Barriers to making more frequent releases.
- Lower team morale
- Slower and less reliable responsiveness to critical problems

# Engineering Practices – XP Practices



Practice	Description
The Planning Game	<p>Business and development cooperate to produce the maximum business value as rapidly as possible. The planning game happens at various scales, but the basic rules are always the same:</p> <ul style="list-style-type: none"> <li>• Business comes up with a list of desired features for the system in user story format.</li> <li>• Development estimates how much effort each story will take, and how much effort the team can produce in a given time interval.</li> <li>• Business then decides which stories to implement in what order.</li> </ul>
Small Releases	Start with the smallest useful feature set. Release early and often, adding a few features each time.
System Metaphor	Each project has an organizing metaphor, which provides an easy to remember naming convention.
Simple Design	<p>At every moment, the design runs all the tests, communicates everything the programmers want to communicate, contains no duplicate code, and has the fewest possible classes and methods.</p> <p>This rule can be summarized as, 'Say everything once and only once.'</p>

Test-Driven Development	<p>Before programmers add a feature, they write a test for it. When the suite runs, the job is done. Tests in XP come in two basic flavors.</p> <ol style="list-style-type: none"> <li>1. <b>Unit Tests</b> are automated tests written by the developers to test functionality as they write it. Each unit test typically tests only a single class, or a small cluster of classes. Unit tests are typically written using a unit testing framework.</li> <li>2. <b>Acceptance Tests</b> are specified by the customer to test that the overall system is functioning as specified. Acceptance tests typically test the entire system, or some large chunk of it. When all the acceptance tests pass for a given user story, that story is considered complete. Test can be manually run, or ideally automated.</li> </ol>
Refactoring	Refactor out any duplicate code generated in a coding session. You can do this with confidence that you didn't break anything because you have the tests.
Pair Programming	All production code is written by two programmers sitting at one machine. Essentially, all code is reviewed as it is written.
Collective Ownership	Every programmer improves any code anywhere in the system at any time if they see the opportunity

Continuous Integration	New code is integrated with the current system after no more than a few hours. When integrating, the system is built from scratch and all tests must pass or the changes are discarded.
Sustainable Pace	No one can work a second consecutive week of overtime. Even isolated overtime used too frequently is a sign of deeper problems that must be addressed.
Customer Tests	A customer sits with the team full-time and tests.
Coding Standards	Everyone codes to the same standards. Ideally, you shouldn't be able to tell by looking at it who on the team has touched a specific piece of code.
Whole Team	All the contributors to an XP project sit together, members of a whole team. The team shares the project goals and the responsibility for achieving them.



# Scrum Roles

## Quiz Time!

**Instructions:** Complete the multiple-choice quiz below. Compare your answers with someone else in your group. When you're done, check your answers on **page 52**.

**1. Select the most accurate statement.**

- a. Estimation is not part of Scrum.
- b. Only Development Team members estimate effort for product backlog items.
- c. People other than the Development team, like project managers and architects, estimate effort for product backlog items.
- d. The Product Owner estimates product backlog items.
- e. The ScrumMaster estimates product backlog items.
- f. None of the above

**2. How are Stakeholders like functional managers, financial sponsors and executives involved with the Scrum Team?**

- a. Stakeholders are involved in the day-to-day activities of the Scrum Team.
- b. Once the Product Backlog is defined, the Stakeholders do not have to pay attention to the activities of the Scrum Team.
- c. Stakeholders participate in all Scrum events.
- d. It's OK for stakeholders to be involved in the Daily Scrum.
- e. Stakeholders provide vital feedback about the product and what to do next at the Sprint Review.
- f. All the above
- g. None of the above

**3. Who analyzes & decomposes Product Backlog items? Choose the best answer.**

- a. Only the Product Owner
- b. The Product Owner and the ScrumMaster
- c. The Product Owner with the assistance of the Development Team
- d. Always the whole Scrum Team

**4. The ScrumMaster role is...**

- a. a full-time role dedicated to assisting the Development Team, Product Owner and Stakeholders in receiving the maximum benefit of using Scrum.
- b. a part time role that anyone on the Development Team can fill.
- c. just like the project manager role.
- d. Accountable for delivery.

**5. Development team members...**

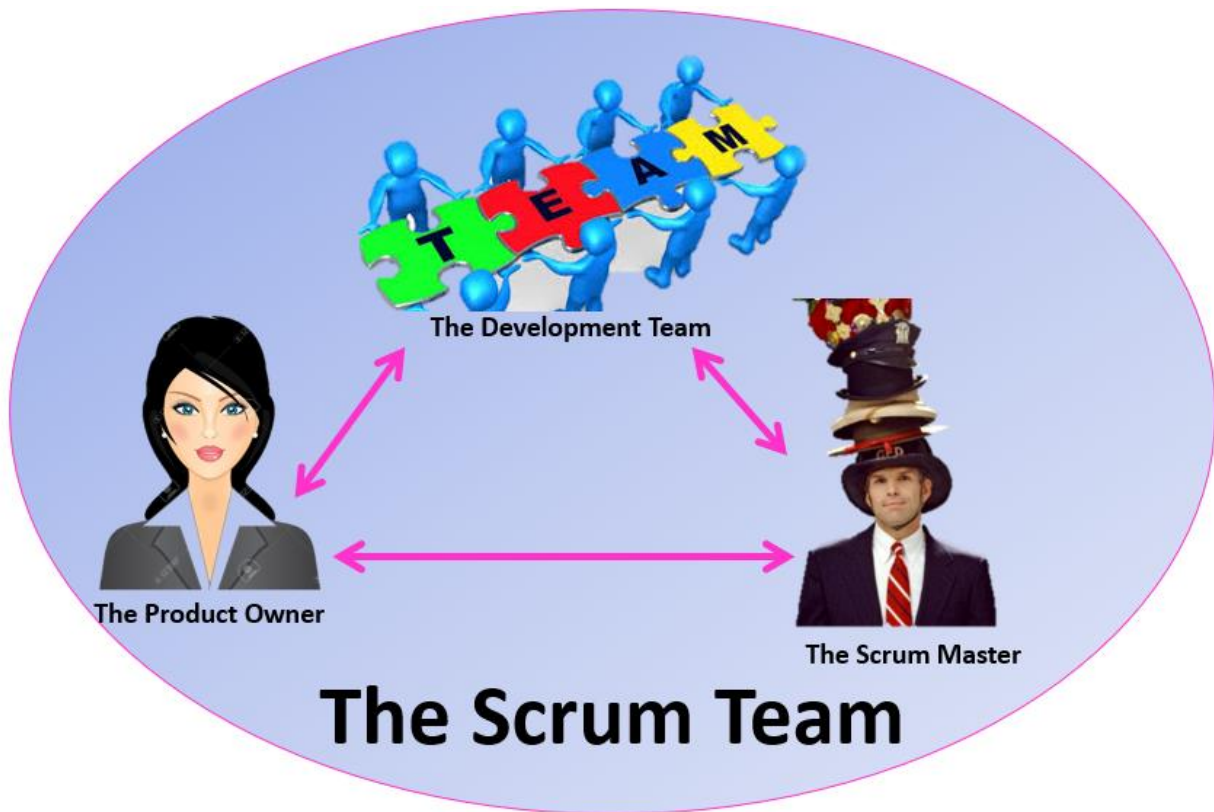
- a. give the Product Owner status updates at the Daily Scrum.
- b. report status to the ScrumMaster at the Daily Scrum.
- c. must follow the three questions at every Daily Scrum.
- d. plan work for the next 24 hours at the Daily Scrum.
- e. can skip the Daily Scrum if they are on track for the Sprint.

**6. As a recognized authority of Scrum, the ScrumMaster...**

- a. assigns work to people to maximize business value for the PO.
- b. performs all the tasks normally expected of the project manager.
- c. reports status to management and other stakeholders.
- d. should remove people from team if they are not collaborating or performing.
- e. teaches others how to apply Scrum and empirical product development.
- f. All the above
- g. None of the above

**7. Is combining Scrum roles a good idea?**

- a. Yes
- b. No
- c. Sometimes



- \* There are three roles in Scrum - Development Team Member, Product Owner and ScrumMaster.
- \* The team model in Scrum is designed to optimize flexibility, creativity, and productivity.
- \* Scrum Teams deliver products iteratively and incrementally, maximizing opportunities for feedback.

# The Product Owner



- \* Responsible for maximizing the value of the Product
- \* Provide Vision + Roadmap
- \* Manages Product Backlog
- \* Ensures Product Backlog is visible, transparent, and clear to all
- \* Clearly describes, orders and clarifies Product Backlog Items
- \* Accepts or Rejects potentially releasable items
- \* Available to the team during Sprint
- \* Authority to Cancel the Sprint
- \* 1 Person and Not a Committee
- \* Decides Release of Product Increments
- \* Discusses the objective that the Sprint should achieve
- \* Has the authority to make decisions in the name of the Stakeholders

# The Development Team



- \* Optimal Development Team size: 3 – 9
- \* Cross- Functional
- \* Self-Organized
- \* Owns Estimates
- \* No Titles
- \* Creates the Increment
- \* Definition of Done
- \* Collective Ownership
- \* No sub-teams
- \* Accepts work only from Product Owner
- \* Owns Sprint Backlog
- \* Participates in Meetings
- \* Creates Sprint backlog and Plan to achieve sprint goal

# The Scrum Master



- \* Promotes & Supports Scrum
- \* Shields team from Interruptions
- \* Helps team overcome Impediments
- \* Servant- Leader
- \* Helps increase Transparency of Artifacts
- \* Facilitator
- \* Serves the Development Team
- \* Serves the Product Owner
- \* Assists Development team in meeting Sprint Goal
- \* Openly addresses issues
- \* Encourages experimenting new ideas
- \* Coach
- \* Change Agent

# Servant Leadership

- Desire to serve is a fundamental characteristic of a servant-leader
- It is about helping and identifying the needs of colleagues, customers, and communities
- Creating an environment where people can work together to create desired results
- Being a great empathetic listener
- Encourage the team : Can WE use "" to keep moving ahead?
- Help them see the overall picture - Do WE have enough info to do this?

Top Down Management	vs	Servant Leadership
<ul style="list-style-type: none"><li>• Manager</li><li>• Authority over team</li><li>• Command and Control</li><li>• Communication link between stakeholders and team</li><li>• Delegates impediment removal to leads of the project</li><li>• Responsible for the success of the project</li></ul>		<ul style="list-style-type: none"><li>• Team Member</li><li>• Authority on Framework</li><li>• Empowers Team Members</li><li>• Helps team be self-organized</li><li>• Removes impediments regularly</li><li>• Empowers team to meet the goal of the Sprint</li><li>• Creates safe environment for the Team</li></ul>

## Servant Leader for the Team

- Guiding the Dev Team towards self-organization
- Helping the team make impediments visible, remove and prevent them
- Helping people develop and perform as highly as possible

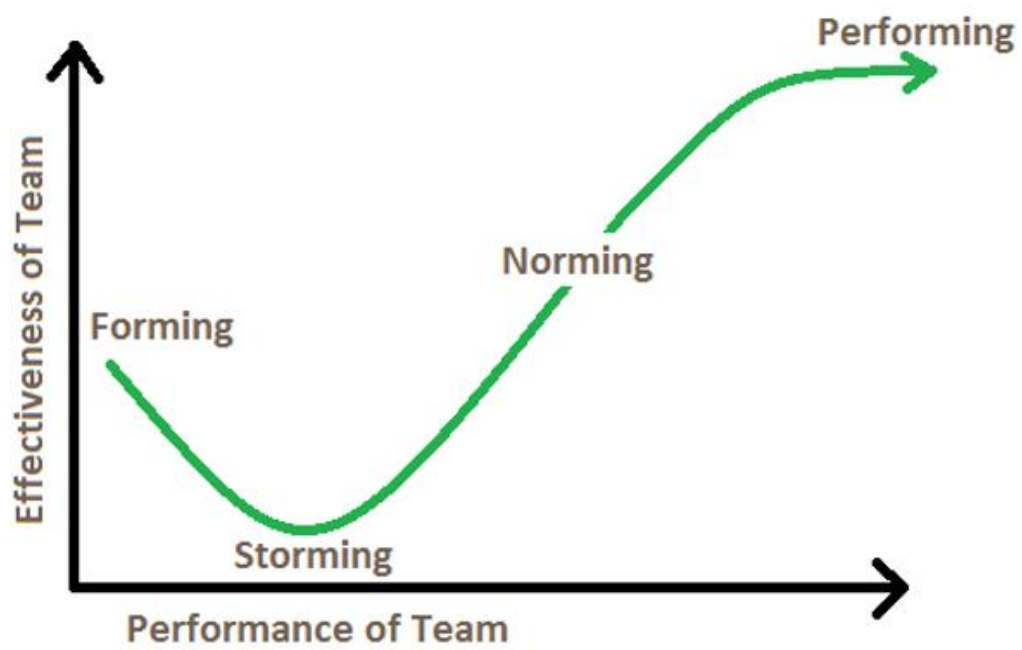
## Servant Leader for the Organization

- Teaching, Coaching & Mentoring the organization in adopting & using Scrum and Being Agile
- Builds communities that works together and learns to serve one another in the process
- Reaches out to the various stakeholders at the organization level and encourages collaboration between different verticals/departments





## Tuckman's Team & Group Development Model



Use the space below to capture the differences between:

**Facilitating**

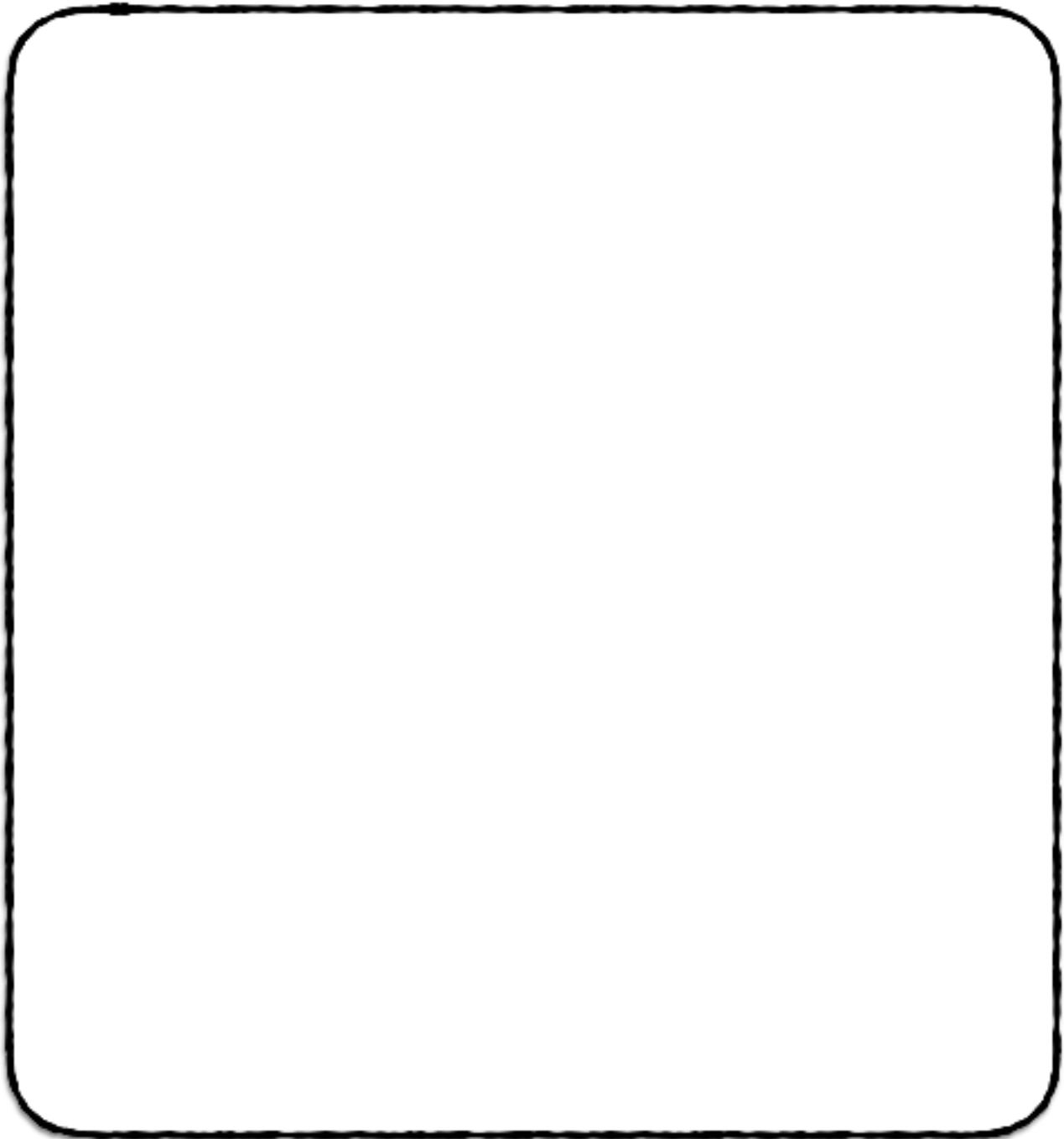
**Mentoring**

**Coaching**

**Teaching**

## Scrum Roles Wrap Up

Use the space below to capture the learning points from the Scrum Roles section.

A large, empty rounded rectangle with a thick black border, intended for capturing learning points. The rectangle is centered on the page and occupies most of the lower half of the document.

# Scrum Events

## **The Sprint – Inspect and Adapt Event**

The heart of Scrum is a Sprint, a 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; and,
- Scope may be clarified and re-negotiated between the Product Owner and Development Team as more is learned.

Each Sprint may be considered a project with no more than a one-month horizon. Like projects, Sprints are used to accomplish something. Each Sprint has a goal of what is to be built, a design and flexible plan that will guide building it, the work, and the resultant product increment.

Sprints are limited to one calendar month.

## **Sprint Planning** – Inspect and Adapt Event

## **Daily Scrum** – Inspect and Adapt Event

## **Sprint Review** – Inspect and Adapt Event

## **Sprint Retrospective** – Inspect and Adapt Event



## **Backlog Refinement** – Inspect and Adapt Activity

# Scrum Artifacts

## Product Backlog

*Owner:*

*Items in it are called:*

*How often can order change:*

*Who contributes to it:*

*When do items get added:*

*Additional Notes:*

# **Sprint Backlog**

*What is it:*

*Owner:*

*Who decides to add/remove  
tasks during a sprint?*

*Additional Notes:*

# **Increment**

*What is it:*

*When is it considered completed:*

*It is a step toward:*

*Who decides to Release it:*

*Additional Notes:*

# Definition of Done

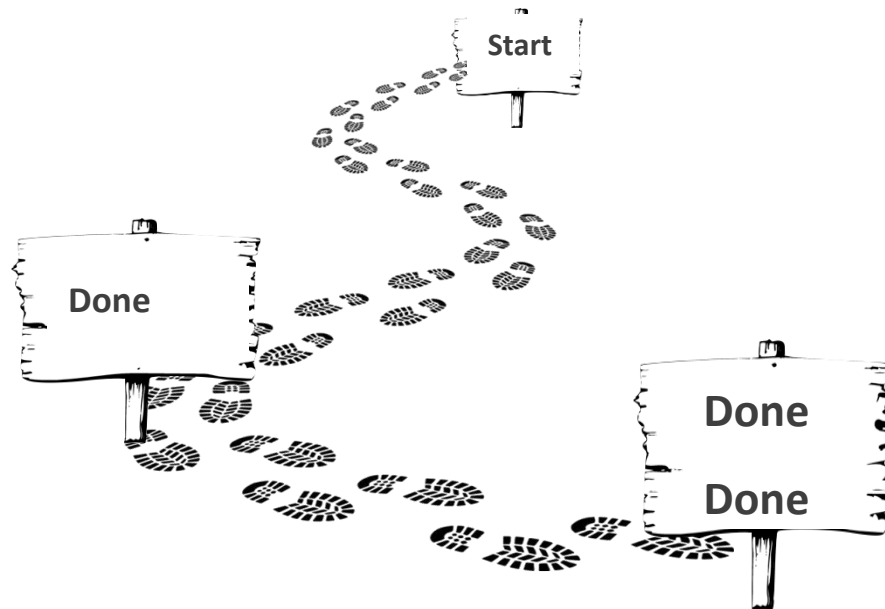
## Synonyms of Done:

- Potentially Releasable
- “Done Done”
- Working Product / Deliverable



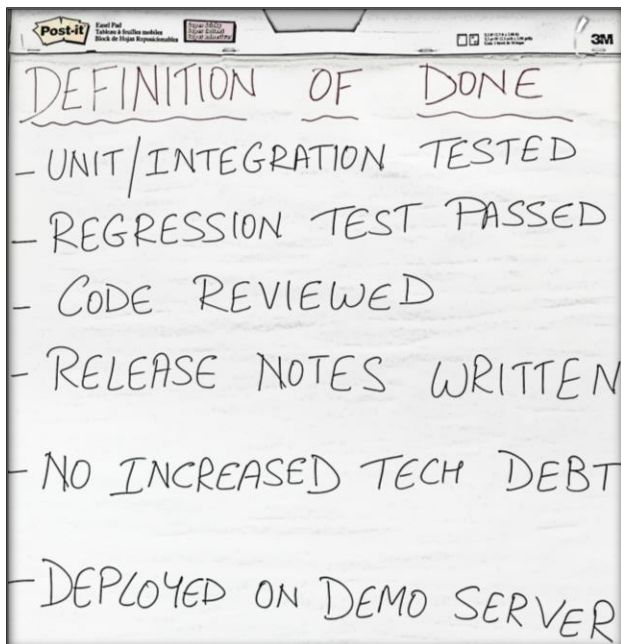
## NOT: A Stretch Goal

- A shared understanding of expectations that Product Increment must live up to in order to be released.
- Definition of Done (DoD) ensures Transparency and Quality of work.
- Guides pre-implementation activities: **Discussion, Estimation, Design.**
- Used to assess when work is complete on the product Increment.
- Limits the cost of rework once a feature has been accepted as "Done".
- Reduced risk of misunderstanding and conflict.
- DoD keeps evolving.



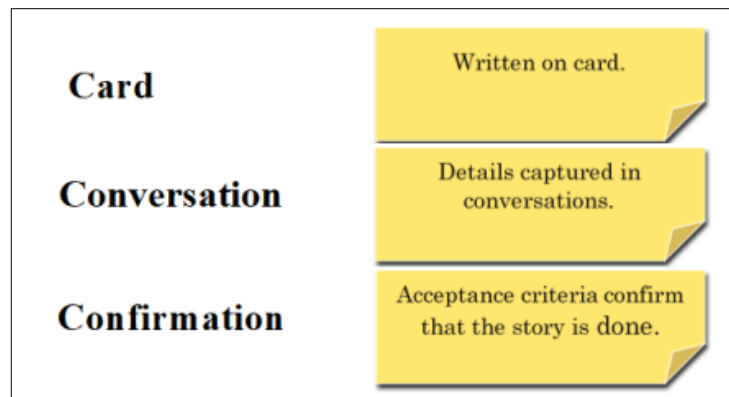
**Activities to complete for Sprint backlog items such that the Product Increment is ready to be shipped/release.**

Sample DOD:



# User Stories

- A description of a feature from an end-**user** perspective
- Describes the type of **USER**, **WHAT** they want and **WHY**
- A simplified description of a requirement
- 3 C's of User Stories



## Front of Card

As a _____
I want _____
So that _____

## Back of Card

I know this story is done when...
○ Acceptance Criteria 1
○ Acceptance Criteria 2
○ Acceptance Criteria 3

## What makes a good User Story

What makes a good User Story	
• <b>Independent</b>	Can be developed, tested (scenarios) & potentially delivered on its own
• <b>Negotiable</b>	Not a contract, but a placeholder for discussion
• <b>Valuable</b>	Must provide value to user, customer, or stakeholder
• <b>Estimable</b>	Must be small enough for rapid implementation, contains clear acceptance criteria
• <b>Small</b>	Sized for a single Sprint
• <b>Testable</b>	Acceptance criteria is verifiable by testers and/or business users

## Examples of User Stories:

- As a teacher, I want to generate assessment report, so I can evaluate student performance.
- As an ATM user, I want to withdraw funds from my bank account, so I can increase my cash on hand.
- As an online grocery shopper, I should be able to save and view my draft order from any of my devices, so that I can complete the order process at my convenience.
- As HR Manager, I want a virtual job openings board, so that I can view job status and manage company personnel needs.
- As a trainer, I want my profile to list my upcoming classes and include a link to a detailed page about each so that prospective attendees can find my courses.
- As someone who wants to hire, I can post a help wanted ad so that I can attract candidates.
- As a participant, I can announce my participation in a course at various times to various social media sites so that I can tell others about what I'm doing.

- Adapted from <https://www.mountangoatsoftware.com/>



# Estimation

- Method of measuring how long it will take to complete a PBI or a task by taking into account:
  - The amount of work to do
  - The complexity of the work
  - Any risk or uncertainty in doing the work
- Story Point is a Relative measure of effort
- We are better at comparing vs. absolute values

Below are some popular Estimation techniques:

## T-shirt Sizing:



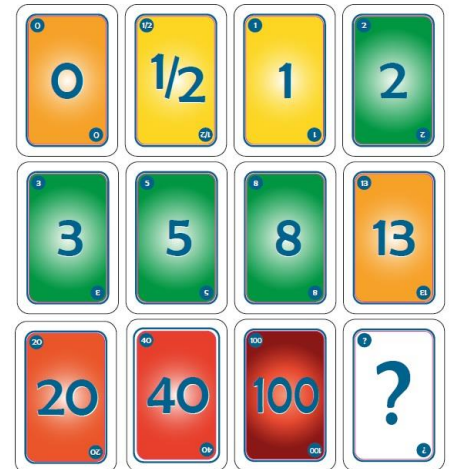
The primary advantage to t-shirt sizes is the ease of getting started. If a team needs to start with t-shirt sizes, that's fine. Later, though, that team will be better off using numbers directly. I might forget that an XL is 33% bigger than an L. I won't forget that a 10 is twice a 5. (If I do, the team has a different problem!)

T-shirt sizes can be a great way of becoming accustomed to relative estimating. So, start with them if your team finds that easier. But minimally put some underlying numbers on them (e.g., Medium=5) and then gradually shift to using the numbers directly.

## Planning Poker:

Planning Poker is an agile estimating and planning technique that is consensus based. To start a poker planning session, the PO reads a User Story or describes a feature to the estimators.

Each estimator is holding a deck of Planning Poker cards with different values, which is the sequence we recommend. The values represent the number of story points, ideal days, or other units in which the team estimates.



The estimators discuss the feature, asking questions to the PO as needed. When the feature has been fully discussed, each estimator privately selects one card to represent his or her estimate. All cards are then revealed at the same time.

If all estimators selected the same value, that becomes the estimate. If not, the estimators discuss their estimates. The high and low estimators should especially share their reasons. After further discussion, each estimator reselects an estimate card, and all cards are again revealed at the same time.

The poker planning process is repeated until consensus is achieved or until the estimators decide that agile estimating and planning of a particular item needs to be deferred until additional information can be acquired.

- Adapted from <https://www.mountangoatsoftware.com/>

## Notes

[illegible]

# Scrum Roles Quiz Answers

1. Select the most accurate statement.
  - a. Estimation is not part of Scrum.
  - b. Only Development Team members estimate effort for product backlog items.**
  - c. People other than the Development team, like project managers and architects, estimate effort for product backlog items.
  - d. The Product Owner estimates product backlog items.
  - e. The ScrumMaster estimates product backlog items.
  - f. None of the above
2. How are Stakeholders like functional managers, financial sponsors and executives involved with the Scrum Team?
  - a. Stakeholders are involved in the day-to-day activities of the Scrum Team.
  - b. Once the Product Backlog is defined, the Stakeholders do not have to pay attention to the activities of the Scrum Team.
  - c. Stakeholders participate in all Scrum events.
  - d. It's OK for stakeholders to be involved in the Daily Scrum.
  - e. Stakeholders provide vital feedback about the product and what to do next at the Sprint Review.**
  - f. All the above
  - g. None of the above
3. Who analyzes and decomposes Product Backlog items? Choose the best answer.
  - a. Only the Product Owner
  - b. The Product Owner and the ScrumMaster
  - c. The Product Owner with the assistance of the Development Team**
  - d. Always the whole Scrum Team

4. The ScrumMaster role is...

- a. is a full time role dedicated to assisting the Development Team, Product Owner and Stakeholders in receiving the maximum benefit of using Scrum.**
- b. a part time role that anyone on the Development Team can fill.
- c. not essential to success with Scrum.
- d. just like the project manager role.
- e. Accountable for delivery.

5. Development team members...

- a. give the Product Owner status updates at the Daily Scrum.
- b. report status to the ScrumMaster at the Daily Scrum.
- c. must follow the three questions at every Daily Scrum.
- d. plan work for the next 24 hours.**
- e. can skip the Daily Scrum if they are on track for the Sprint.

6. As a recognized authority of Scrum, the ScrumMaster...

- a. assigns work to people to maximize business value for the Product Owner.
- b. performs all the tasks normally expected of the project manager.
- c. reports status to management and other stakeholders.
- d. should remove people from the team if they are not collaborating or performing.
- e. teaches others how to apply Scrum and empirical product development.**
- f. All the above
- g. None of the above

7. Is combining Scrum roles a good idea?

- a. Yes
- b. No**
- c. Sometimes

# Recommended Reading

