CCS|CSE|CIS1072- Introduction to Software Engineering

June 15, 2026

SOFTWARE DEVELOPMENT PROCESS

Romenia Silva
romaniasilva@sjp.ac.lk

WHY DO WE NEED SOFTWARE DEVELOPMENT PROCESS?

- **Manage Complexity:** Modern software systems require systematic approaches
- **Ensure Quality:** Consistent standards and reliable outcomes
- **Control Risks:** Early identification and mitigation of problems
- **Optimize Resources:** Efficient use of time, budget, and team capabilities
- **Enable Communication:** Common understanding across stakeholders

SOFTWARE DEVELOPMENT PROCESS

A software development process is a structured framework that defines the activities, methods, and practices used to plan, create, test, and deploy software systems.

Traditional Approaches	Modern Approaches
Waterfall Model	Agile
Spiral Model	DevOps

WHAT IS AGILE?

Agile is the ability to create and respond to **change**. It is a way of dealing with, and ultimately succeeding in, an uncertain and unstable environment..

Agile is an **iterative development** methodology, where requirements evolve through **collaboration** between the **customer** and **self-organizing teams** and agile **aligns** development with **customer needs**.

THE AGILE **MANIFESTO**

TOP 4 AGILE VALUES

	INDIVIDUAL INTERACTIONS	OVER	PROCESS AND TOOLS 
	WORKING SOFTWARE		THOROUGH DOCUMENTATIONS 
	COLLABORATION WITH CUSTOMERS		CONTRACT NEGOTIATIONS 
	RESPOND TO CHANGE		FOLLOWING A PLAN 

THE AGILE **MANIFESTO** CONT.

THE 12 AGILE PRINCIPLES

12 Core Tenets from the Agile Manifesto

1. Satisfy Customer

Deliver valuable software early and continuously.

2. Welcome Change

Embrace changing requirements for customer competitive advantage.

3. Deliver Frequently

Working software in short cycles (weeks to months).

4. Daily Collaboration

Work together daily (Business & Developers)

5. Motivated Individuals

Build projects with motivated people, trust and support.

6. Face-to-Face

Most efficient method for conveying info.

7. Working Software

The primary measure of progress.

8. Sustainable Development

Maintain a constant pace indefinitely (No burnout)

9. Continuous Attention to Technical Excellence

Continuous focus on high-quality code and design enhances agility.

10. Simplicity

Maximize the amount of work *not* done.

11. Self-Organizing Teams

Best architecture, requirements, and designs emerge.

12. Reflect & Adjust

Regular intervals for reflection and adjustment to improve.

1. Which of the following does not apply to agility to a software process?

A. Uses incremental product delivery strategy.
B. Only essential work products are produced
C. Eliminate the use of project planning and testing
D. All the mentioned

Answer: C

2. Collaboration requires that the team must take joint responsibility for their work. In order for this to effectively take place, what must the team members build?

A. Definition of done
B. Continuous flow
C. Trust
D. An information radiator

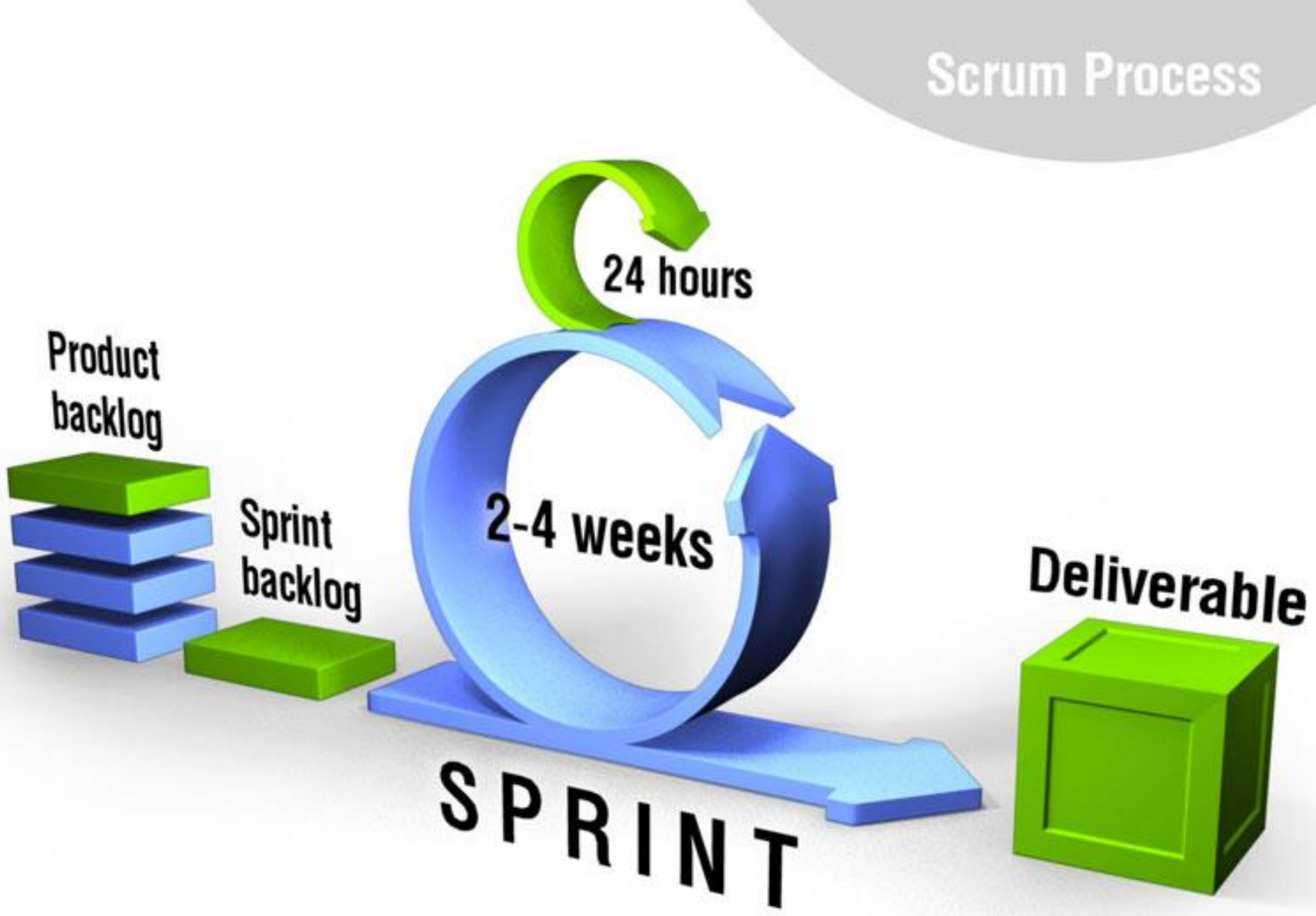
Answer: C

Scrum

"Scrum is an **agile way to manage a project**, usually software development. Agile software development with Scrum is often perceived as a **methodology**; but rather than viewing Scrum as methodology, think of it as a **framework** for managing a process."

Why is it called Scrum?

The software development term **scrum** was first used in a 1986 paper titled "The New New Product Development Game". The term is borrowed from rugby, where a **scrum** is a formation of players. The term **scrum** was chosen by the paper's authors because it emphasizes teamwork.

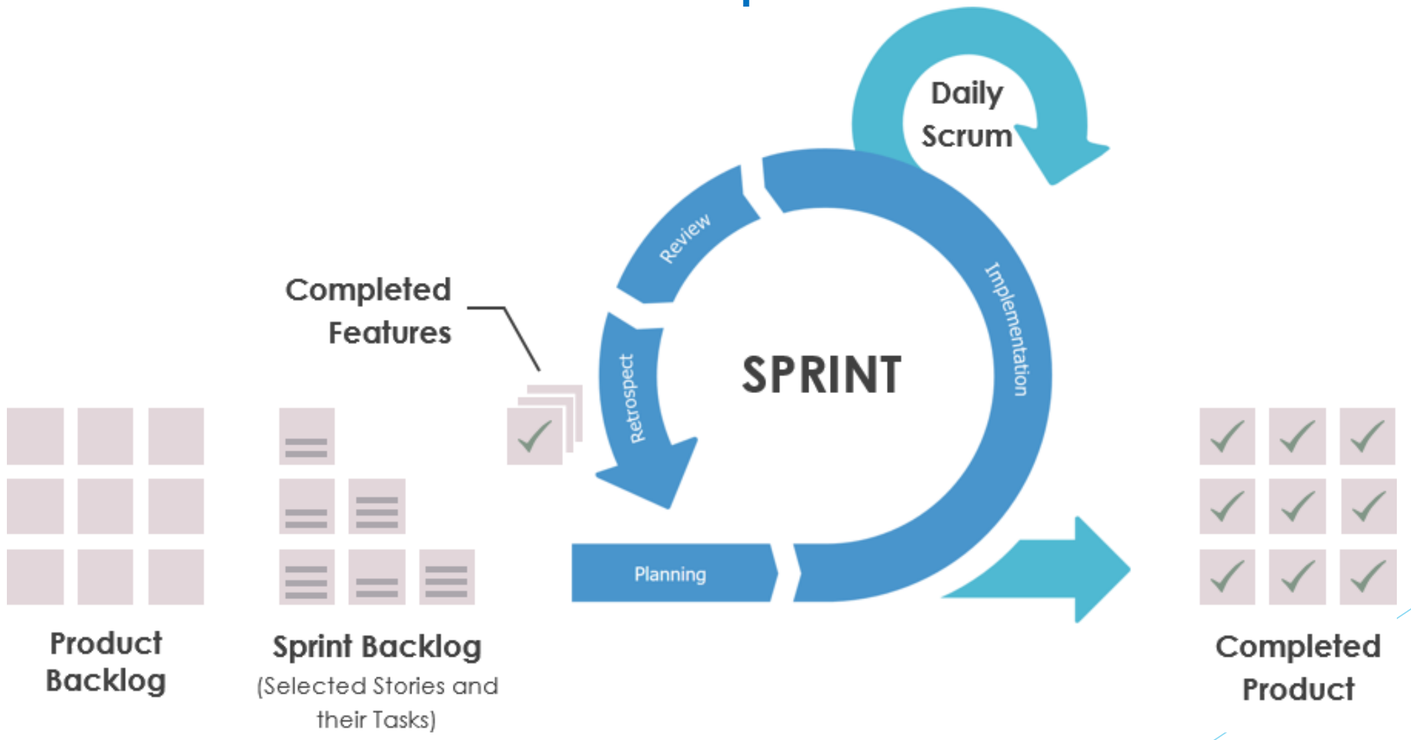


Scrum Process

The Main Artifacts

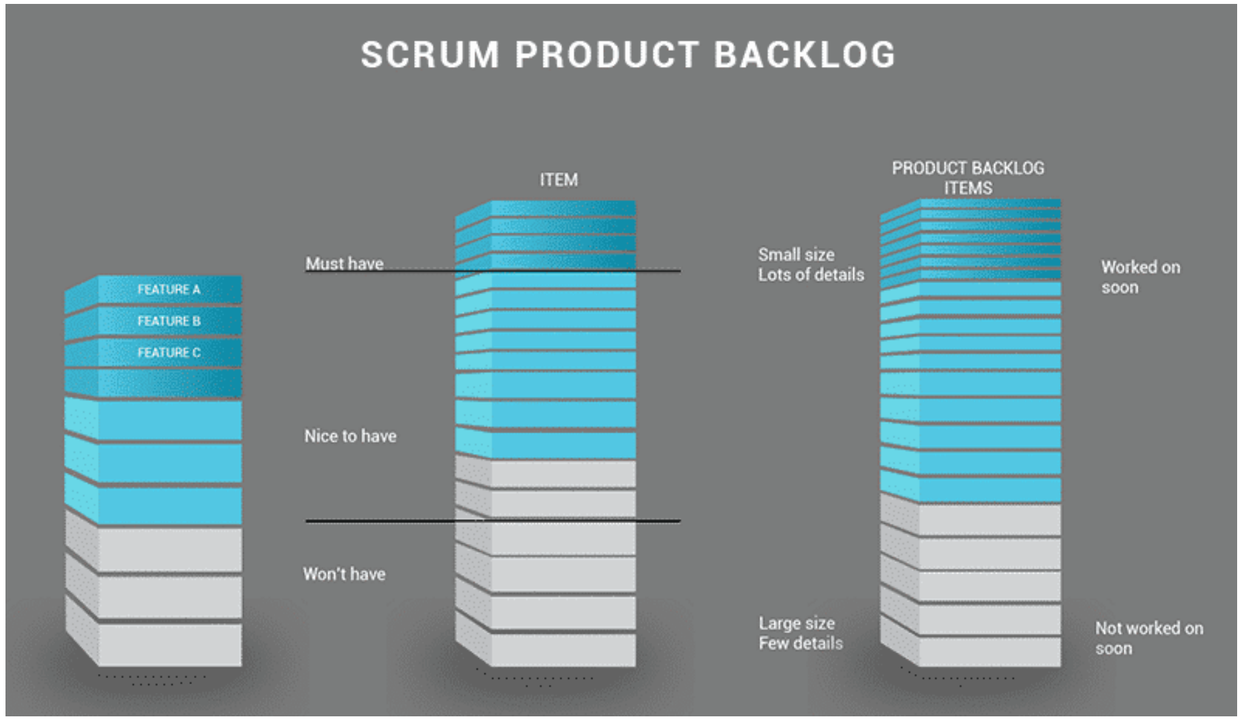
Scrum Process

- The primary artifact in Scrum development is, of course, the product itself. The Scrum model expects the team to bring the product or system to a potentially shippable state at the end of each Scrum sprint.



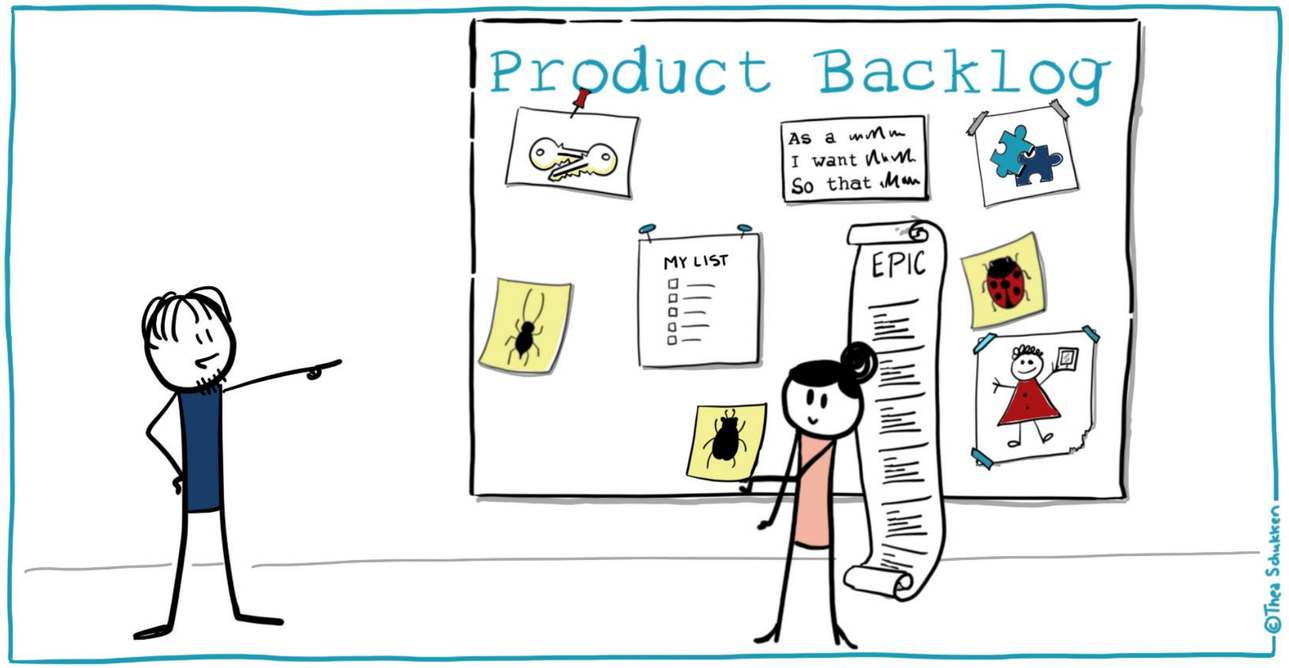
Scrum Process Cont.

- The product backlog is another artifact of Scrum. **This is the complete list of the functionality** that remains to be added to the product. The product owner prioritizes the backlog so the team always works on the most valuable features first.



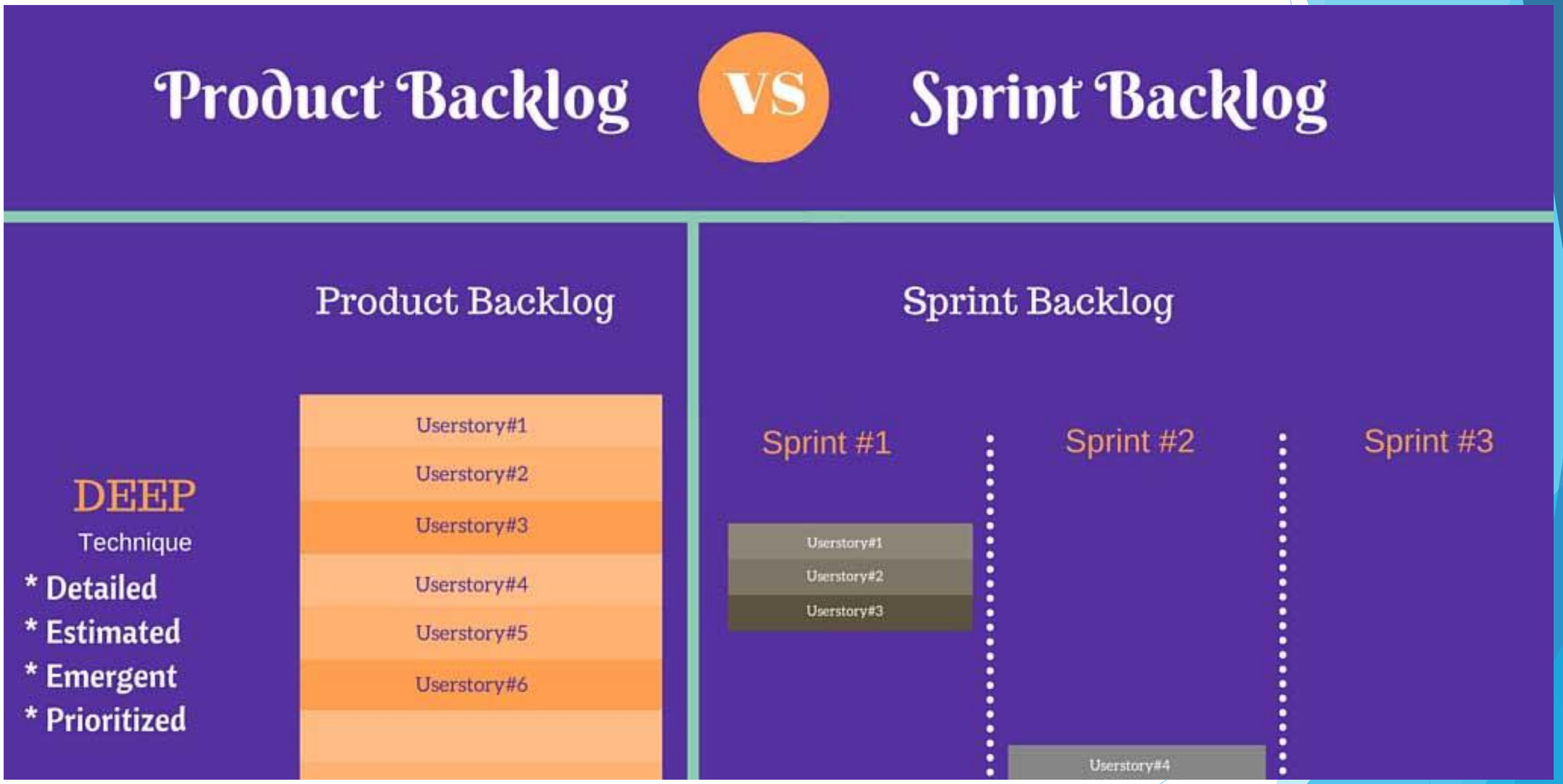
Scrum Process Cont.

- The most popular and successful way to create a product backlog using Scrum methodology is to populate it with **user stories**, which are short descriptions of functionality described from the perspective of a user or customer.



Scrum Process Cont.

- In Scrum project management, on the first day of a sprint and during the planning meeting, team members create the sprint backlog. The sprint backlog can be thought of as the team's to-do list for the sprint, whereas a product backlog is a list of features to be built (written in the form of user stories).


Product Backlog
VS
Sprint Backlog
Product Backlog
DEEP
Technique
* Detailed
* Estimated
* Emergent
* Prioritized
Userstory#1
Userstory#2
Userstory#3
Userstory#4
Userstory#5
Userstory#6
Sprint Backlog
Sprint #1
Sprint #2
Sprint #3
Userstory#1
Userstory#2
Userstory#3
Userstory#4

Product Backlog vs. Sprint Backlog Difference

Product Backlog

- The product backlog is a priority list of user requirements, use cases to be done in order to create, maintain and sustain a product.
- **Product Owner** owns the product backlog,(s)he is the one who prioritize it based on the customers feedback or business value.

Sprint backlog

- Sprint backlog is the subset of the product backlog.
- Each sprint, **scrum team** picks the user stories from product backlog on top of its stack, the number of user story picked by scrum team for a time box sprint is based on the average velocity of a scrum team.
- Product Owner set the sprint's goal for the team, scrum team pick the user stories from product backlog fulfilling those goals.
- Those user stories which moved to sprint is owned by scrum team, as the team is committed with the sprint backlog items during a sprint which is in time box.

Scrum Process

- Additional artifacts resulting from the Scrum agile methodology is the sprint burndown chart and release burndown chart
- Burndown charts show the amount of work remaining either in a sprint or a release and are an effective tool in Scrum software development to determine whether a sprint or release is on schedule to have all planned work finished by the desired date.

The Scrum Team

The Scrum Team

Scrum teams are typically composed of 7 ± 2 members and have no team leader to delegate tasks or decide how a problem is solved.

The team as a unit decides how to address issues and solve problems.

Each member of the Scrum team is an integral part of the solution and is expected to carry a product from inception to completion.

There are three key roles in a Scrum team

Product Owner

Scrum Master

Development Team

The Product Owner

- The Product Owner is the project's key stakeholder - usually an internal or external customer, or a spokesperson for the customer.
- There is only one Product Owner who conveys the overall mission and vision of the product which the team is building.
- The Product Owner is ultimately accountable for managing the product backlog and accepting completed increments of work.

The Scrum Master

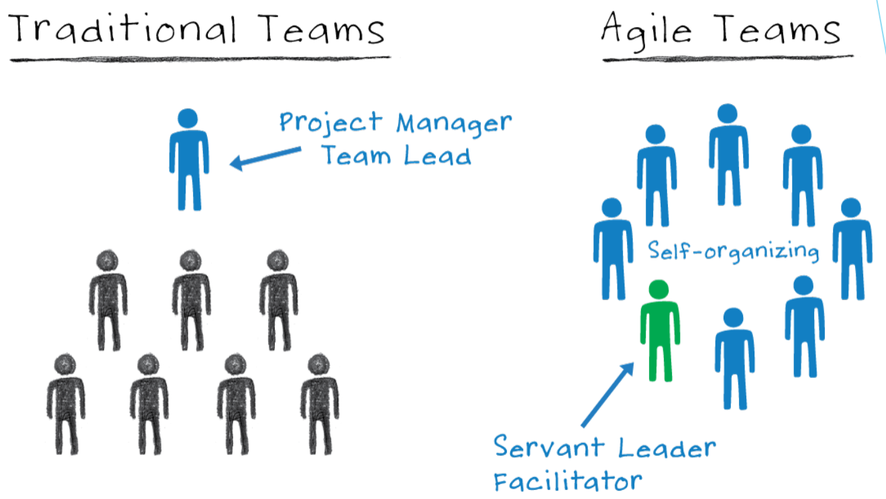
- The Scrum Master is the **servant leader** to the Product Owner, Development Team and Organization.
- With no hierarchical authority over the team but rather more of a facilitator, the ScrumMaster ensures that the team adheres to Scrum theory, practices, and rules.
- The ScrumMaster protects the team by doing anything possible to help the team perform at the highest level.
- This may include removing impediments, facilitating meetings, and helping the Product Owner groom the backlog.

The Development Team

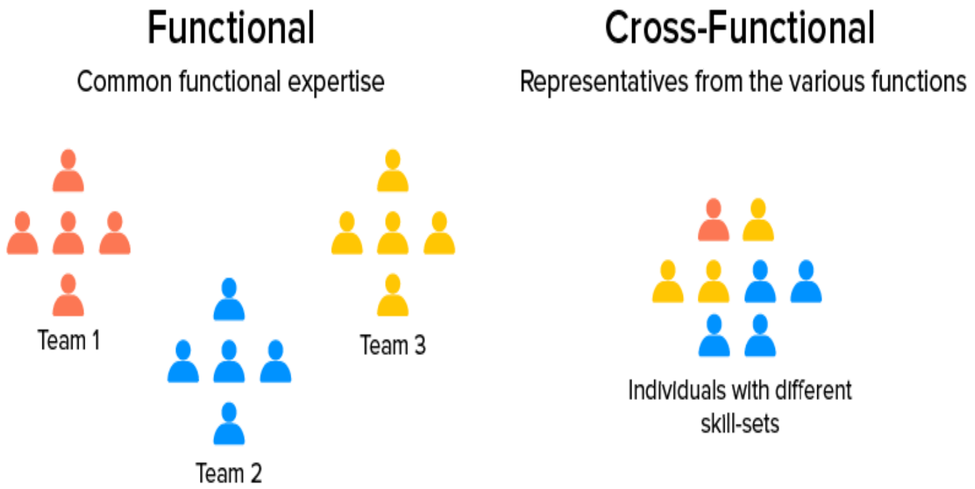
- The Development Team is a **self-organizing, cross-functional** group armed with all of the skills to deliver shippable increments at the completion of each sprint.
- Scrum broadens the definition of the term "developer" beyond programmers to include anyone who participates in the creation of the delivered increment.
- There are no titles in the Development Team and no one, including the ScrumMaster, tells the Development Team how to turn product backlog items into potentially shippable increments

Scrum Team

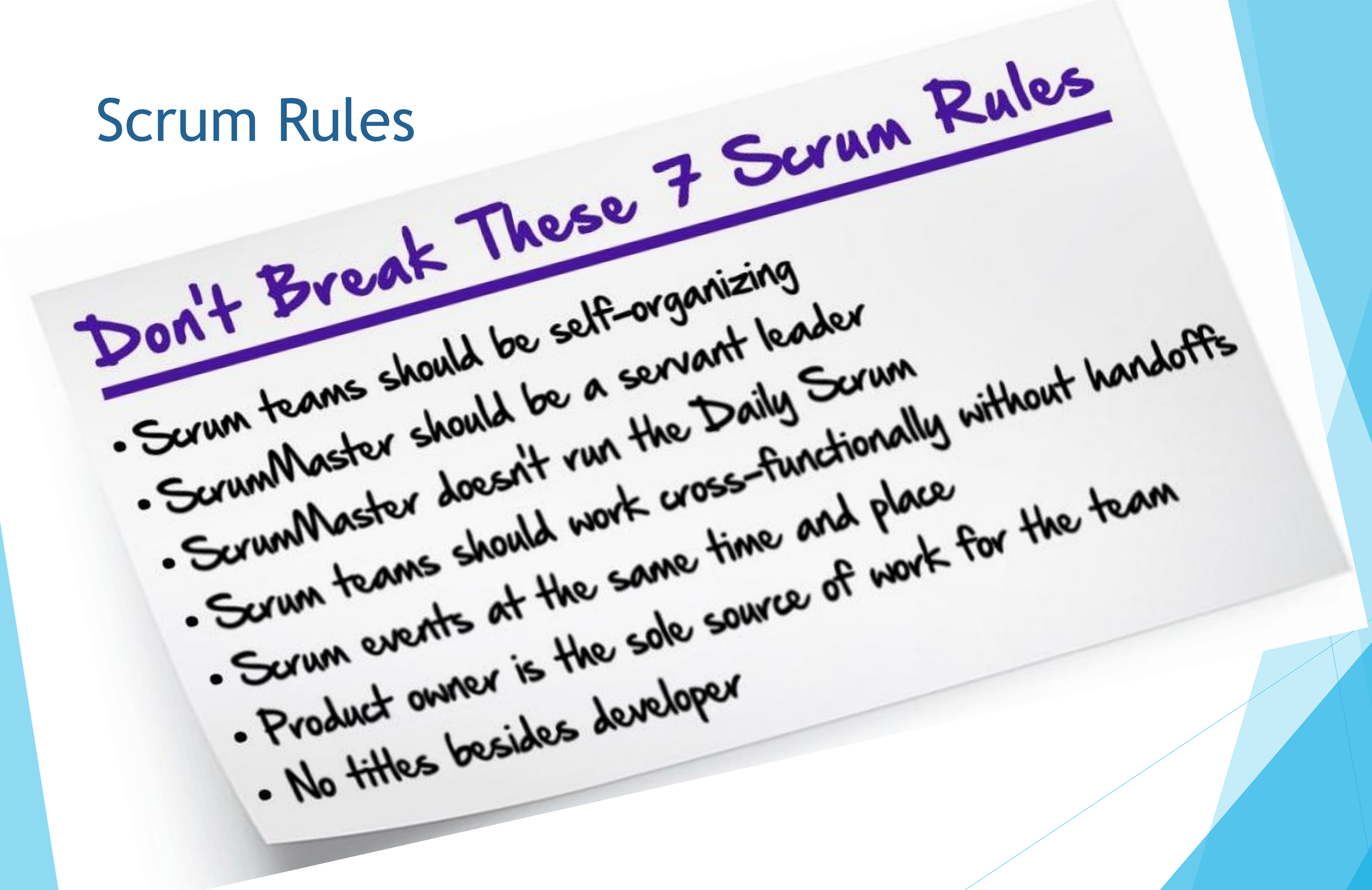
- Scrum Team is Self-Organizing



- Scrum Team is Cross-Functional



Scrum Rules


Don't Break These 7 Scrum Rules
• Scrum teams should be self-organizing
• ScrumMaster should be a servant leader
• ScrumMaster doesn't run the Daily Scrum
• Scrum teams should work cross-functionally without handoffs
• Scrum events at the same time and place
• Product owner is the sole source of work for the team
• No titles besides developer

Benefits

- Higher productivity
- Better-quality products
- Reduced time to market
- Improved stakeholder satisfaction
- Better team dynamics
- Happier employees

1. In Scrum, when is a Sprint Over?

A. When all the Sprint Backlog Items are completed

B. When the Product Owner suggests

C. When all the Sprint Backlog tasks are completed

D. When the final testing is completed

E. When the time box expires

Answer: E

2. Which of the following characteristics is MOST important for an Agile team?

A. They should be able to create and manage their own work schedule
B. They should be able to plan the project
C. They should be flexible and adaptable
D. They should be able to work on multiple tasks at a time

Answer: C

3. Which of the following is delivered at the end of the Sprint?

A. A document containing test cases for the current sprint
B. An architectural design of the solution
C. Wireframes designs for User Interface
D. An increment of Done software

.

Answer: D

4. Why should the team be interested in pursuing an incremental approach to delivering features?

A. Incremental delivery provides early value to the customer and reduces the risk of delivering something the customer does not want.
B. Incremental delivery allows you to bunch features, so that customer feels he has got something substantial.
C. Incremental delivery eliminates the possibility of change requests
D. Incremental delivery simplifies work for the team

Answer: A

5. In a Scrum team, who is responsible for ensuring compliance with the Scrum practices?

A. The team
B. The Scrum Master
C. The Manager
D. The Agile coach

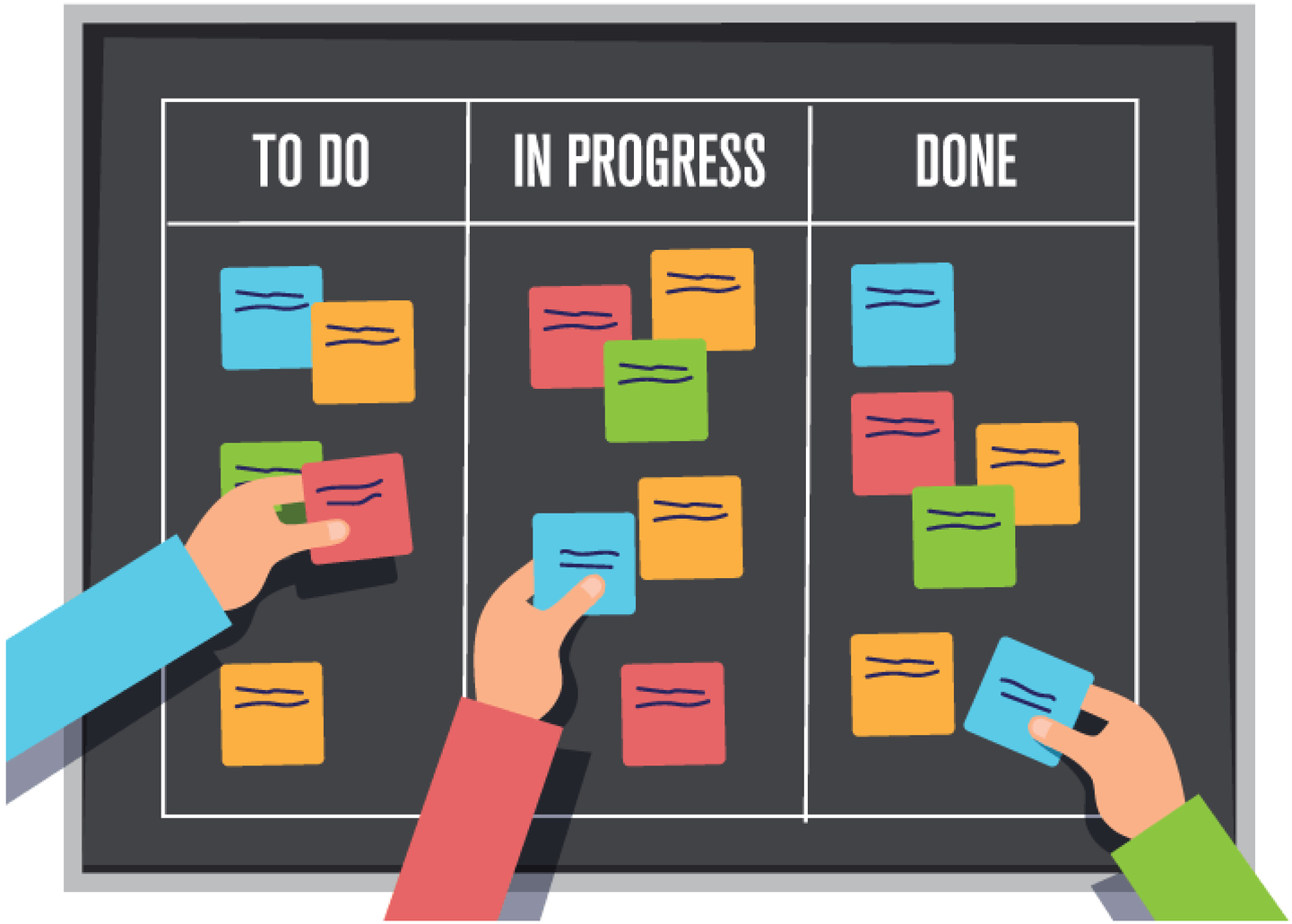
Answer: B

6. What do we mean by a cross-functional Development Team?

A. The developer should be able to create test cases and execute them
B. The Development Team should collaborate with the other Development Teams
C. The Development Team consist of Developers and Testers
D. The Development Team should have all the skills necessary to deliver the Done Increment

Answer: D

Kanban



What Is Kanban

Visualize the workflow :
Split the work into pieces, write each item on a card and stick on the wall
Use named columns to illustrate where each item is in the workflow

Limit Work In Progress (WIP) :
Assign explicit limits to how many items could be in progress at each workflow state.

Measure the lead time :
Average time to complete one item, sometimes called as ‘cycle time’, optimize the process to make lead time as small and predictable as possible

Scrum vs Kanban

	Scrum	Kanban
Cadence	Regular fixed length sprints (ex, 2 weeks)	Continuous flow
Release methodology	At the end of each sprint	Continuous delivery
Roles	Product owner, scrum master, development team	No required roles
Key metrics	Velocity	Lead time, cycle time, WIP
Change philosophy	Teams should not make changes during the sprint.	Change can happen at any time

1. How does Kanban prevent work over capacity?

A. By using Work In Progress (WIP) Limit.
B. By setting a robust Kanban workflow.
C. By having daily meetings about work in progress.
D. By defining explicit policies.

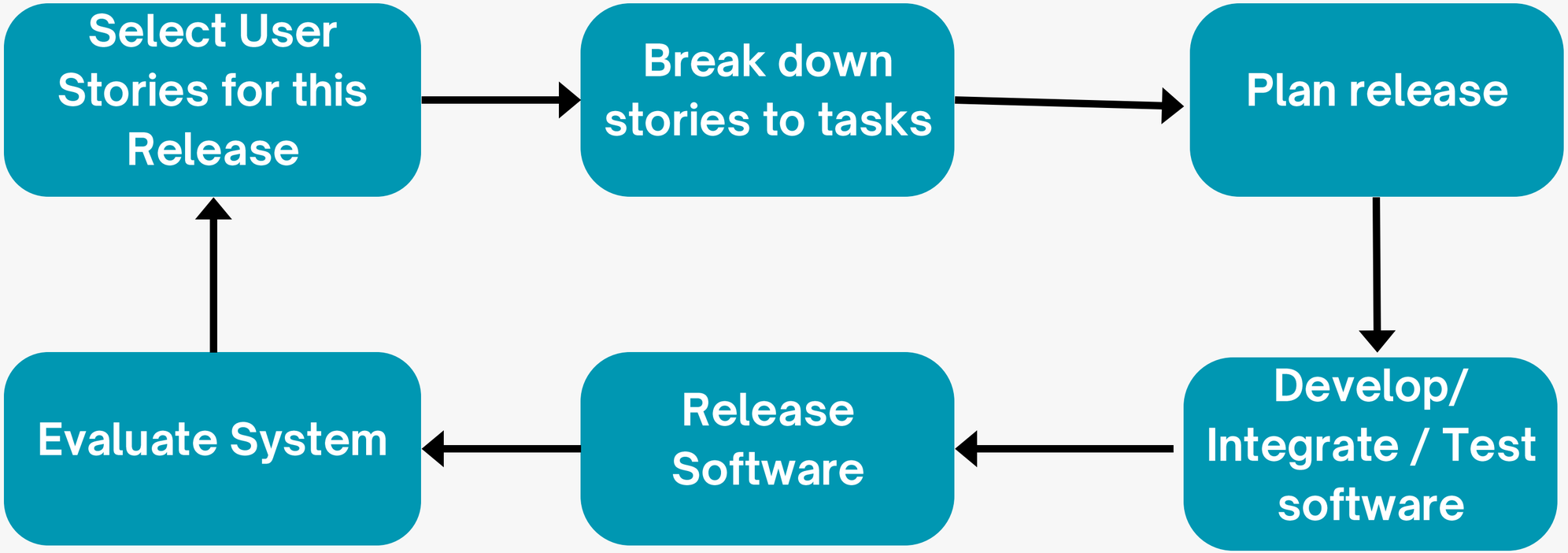
Answer: A

Extreme Programming (XP)

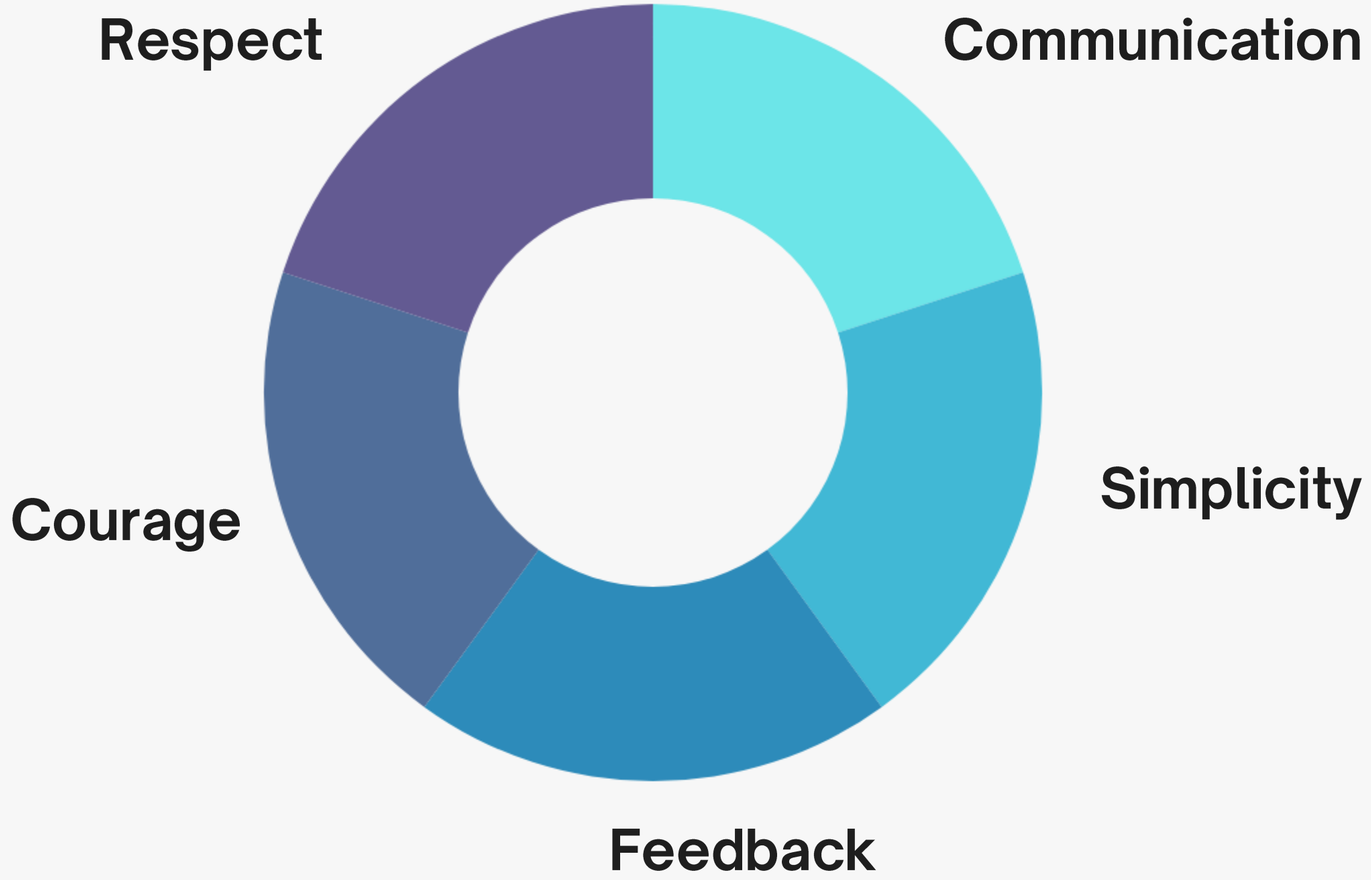
Extreme Programming (XP) is an **Agile software development methodology** that focuses on improving **software quality** and the ability to respond to **changing customer requirements**.

It’s called “Extreme” because it takes **best practices** in software development and pushes them to **extreme levels**.

The extreme programming release cycle



5 Core Values of XP



Key Practices of XP

1. Pair Programming	Developers work in pairs, checking each other's work and providing the support to always do a good job.
2. Test-Driven Development (TDD)	An automated unit test framework is used to write tests for a new piece of functionality before that functionality itself is implemented
3. Continuous Integration	As soon as the work on a task is complete, it is integrated into the whole system. After any such integration, all the unit tests in the system must pass.
4. Refactoring	All developers are expected to refactor the code continuously as soon as possible code improvements are found. This keeps the code simple and maintainable
5. Small Releases	The minimal useful set of functionality that provides business value is developed first. Releases of the system are frequent and incrementally add functionality to the first release.

Key Practices of XP

6. On-Site Customer	A representative of the end-user of the system (the customer) should be available full time for the use of the XP team. In an extreme programming process, the customer is a member of the development team and is responsible for bringing system requirements to the team for implementation
7. Planning Game	Requirements are recorded on story cards and the stories to be included in a release are determined by the time available and their relative priority. The developers break these stories into development ‘Tasks’.
8. Sustainable pace	Large amounts of overtime are not considered acceptable as the net effect is often to reduce code quality and medium term productivity
9. Simple design	Enough design is carried out to meet the current requirements and no more
10. Collective ownership	The pairs of developers work on all areas of the system, so that no islands of expertise develop and all the developers take responsibility for all of the code. Anyone can change anything.

Benefits of Agile Development

For Organizations	For Development Teams	For Customers
• Faster time to market • Better quality products • Increased customer satisfaction • Reduced project risk • Improved team morale	• More autonomy and empowerment • Regular feedback and improvement • Better work-life balance • Continuous learning opportunities	• Early and frequent delivery of value • Greater involvement in development process • Ability to change requirements • Higher quality products

Agile vs. Traditional Development

Aspect	Agile	Traditional (Waterfall)
Approach	Iterative and incremental	Sequential phases
Requirements	Evolving and flexible	Fixed and detailed upfront
Customer Involvement	Continuous throughout	Mainly at beginning and end
Documentation	Just enough	Comprehensive
Testing	Continuous	After development phase
Risk Management	Early identification	Late identification
Change Response	Embrace change	Change is expensive

STUDY QUESTIONS

1. What are the four core values of the Agile Manifesto?
2. Explain the difference between Scrum and Kanban approaches.
3. How does Test-Driven Development support Agile principles?
4. What role does the Product Owner play in Scrum?
5. How do you measure the success of an Agile project?
6. What are the main challenges in transitioning from traditional to Agile development?
7. Explain the concept of "user stories" and their importance in Agile.
8. How does continuous integration support Agile development?