tutorialspoint

SIMPLYEASYLEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia https://twitter.com/tutorialspoint

About the Tutorial

Agile is a software development methodology to build a software incrementally using short iterations of 1 to 4 weeks so that the development is aligned with the changing business needs. This simple tutorial uses appropriate examples to help you understand agile development in a general and quick way.

Audience

This tutorial has been prepared for beginners to help them understand the basics of Agile principles and its implementation. After completing this tutorial, you will find yourself at a moderate level of expertise, from where you can advance further.

Prerequisites

Before proceeding with this tutorial, you need a basic knowledge of software development concepts such as software requirements, coding, testing, etc.

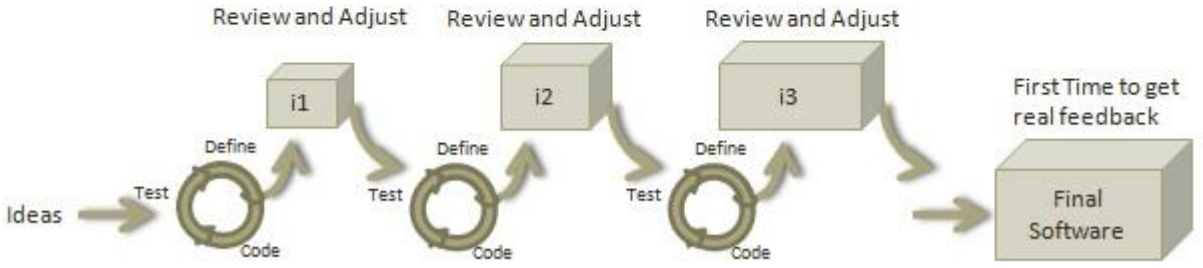
Copyright & Disclaimer

Table of Contents

1. AGILE – PRIMER

Agile is a software development methodology to build a software incrementally using short iterations of 1 to 4 weeks so that the development process is aligned with the changing business needs. Instead of a single-pass development of 6 to 18 months where all the requirements and risks are predicted upfront, Agile adopts a process of frequent feedback where a workable product is delivered after 1 to 4 week iteration.

Traditional Method



Agile Method

Roles in Agile

Scrum Master

A Scrum Master is a team leader and facilitator who helps the team members to follow agile practices so that they can meet their commitments. The responsibilities of a scrum master are as follows:

- To enable close co-operation between all roles and functions.
- To remove any blocks.
- To shield the team from any disturbances.
- To work with the organization to track the progress and processes of the company.
- To ensure that Agile Inspect & Adapt processes are leveraged properly which includes

- Daily stand-ups,
- Planned meetings,
- Demo,
- Review,
- Retrospective Meetings, and
- To facilitate team meetings and decision-making process.

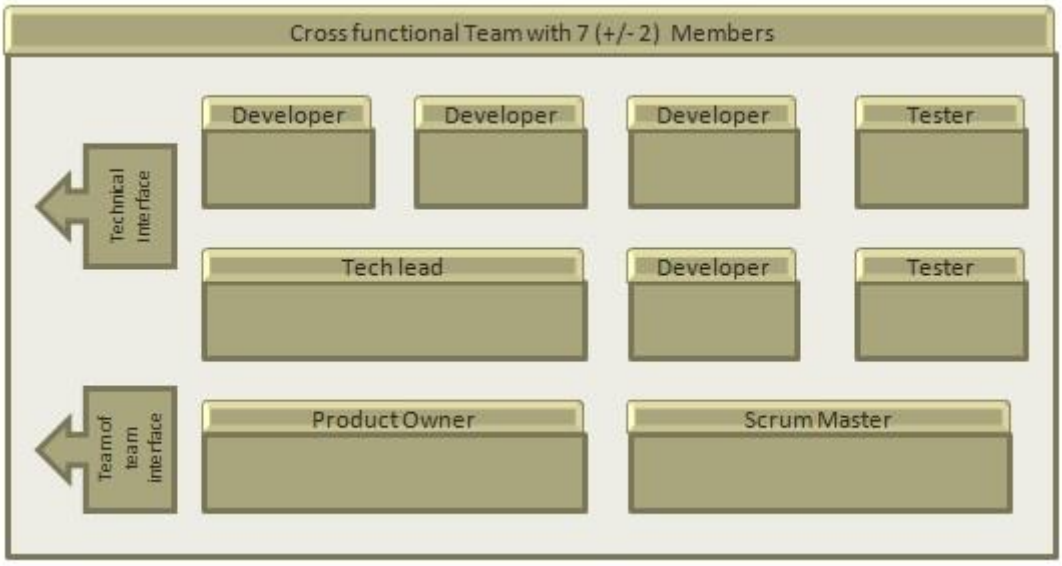
Product Owner

A Product Owner is the one who drives the product from business perspective. The responsibilities or a Product Owner are as follows:

- To define the requirements and prioritize their values.
- To determine the release date and contents.
- To take an active role in iteration planning and release planning meetings.
- To ensure that team is working on the most valued requirement.
- To represent the voice of the customer.
- To accept the user stories that meet the definition of done and defined acceptance criteria.

Cross-functional Team

Every agile team should be a self-sufficient team with 5 to 9 team members and an average experience ranging from of 6 to 10 years. Typically, an agile team comprises of 3 to 4 developers, 1 tester, 1 technical lead, 1 product owner and 1 scrum master.



Product Owner and Scrum master are considered to be a part of Team Interface, whereas other members are part of Technical Interface.

How an Agile Team Plans its Work?

An Agile team works in iterations to deliver user stories where each iteration is of 10 to 15 days. Each user story is planned based on its backlog prioritization and size. The team uses its capacity – how many hours are available with team to work on tasks – to decide how much scope they have to plan.

Point

A Point defines how much a team can commit. A point usually refers to 8 hours. Each story is estimated in points.

Capacity

Capacity defines how much an individual can commit. Capacity is estimated in hours.

What is a User Story?

A user story is a requirement which defines what is required by the user as functionality. A user story can be in two forms:

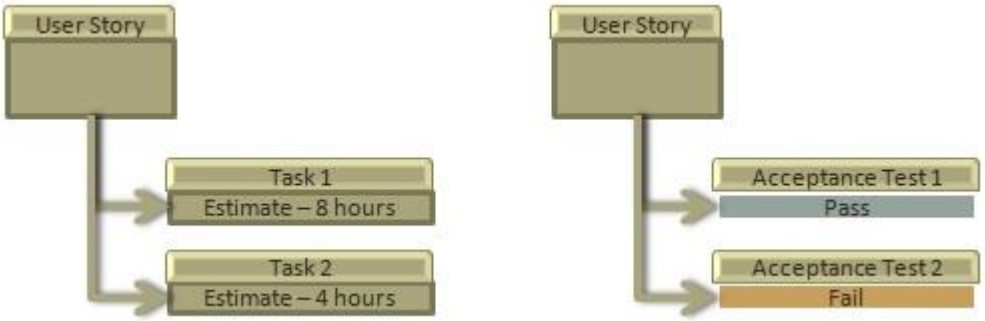
- As a <User Role> I want <Functionality> so that <Business Value>
- In order to <Business value> as a <User Role> I want <Functionality>

During release planning, a rough estimate is given to a user story using relative scale as points. During iteration planning, the story is broken down into tasks.

Relationship of User Stories and Tasks

- User story talks about what is to be done. It defines what a user needs.
- Task talks about how it is to be done. It defines how a functionality is to be implemented.
- Stories are implemented by tasks. Each story is a collection of tasks.
- User story is divided into tasks when it is planned in current iteration.
- Tasks are estimated in hours, typically from 2 to 12 hours.

- Stories are validated using acceptance tests.



When a Story is Done

The team decides what **done** means. The criteria may be:

- All tasks (development, testing) are completed.
- All acceptance tests are running and are passed.
- No defect is open.
- Product owner has accepted the story.
- Deliverable to the end-user.

What is Acceptance Criteria?

Criteria defines the functionality, behavior, and performance required by a feature so that it can be accepted by the product owner. It defines what is to be done so that the developer knows when a user story is complete.

How the Requirements are Defined?

Requirements are defined as

- A User Story,
- With Acceptance Criteria, and
- Tasks to implement the story.

2. AGILE – MANIFESTO

In February 2001, at the Snowbird resort in Utah, 17 software developers met to discuss lightweight development methods. The outcome of their meeting was the following Agile Manifesto for software development:

> We are uncovering better ways of developing software by doing it and helping others do it. Through this work, we have come to value:
>
> - Individuals and interactions over Processes and tools
> - Working software over Comprehensive documentation
> - Customer collaboration over Contract negotiation
> - Responding to change over Following a plan
>
> That is, while there is value in the items on the right, we value the items on the left more.

Twelve Principles of Agile Manifesto

1. **Customer Satisfaction** - Highest priority is given to satisfy the requirements of customers through early and continuous delivery of valuable software.

2. **Welcome Change** - Changes are inevitable during software development. Ever-changing requirements should be welcome, even late in the development phase. Agile processes should work to increase customers' competitive advantage.

3. **Deliver a Working Software** - Deliver a working software frequently, ranging from a few weeks to a few months, considering shorter time-scale.

4. **Collaboration** - Business people and developers must work together during the entire life of a project.

5. **Motivation** - Projects should be built around motivated individuals. Provide an environment to support individual team members and trust them so as to make them feel responsible to get the job done.

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

7. **Measure the Progress as per the Working Software** - Working software is the key and it should be the primary measure of progress.

8. **Maintain Constant Pace** - Agile processes aim towards sustainable development. The business, the developers, and the users should be able to maintain a constant pace with the project.

9. **Monitoring** - Pay regular attention to technical excellence and good design to enhance agility.

10. **Simplicity** - Keep things simple and use simple terms to measure the work that is not completed.

11. **Self-organized Teams** - An agile team should be self-organized and should not depend heavily on other teams because the best architectures, requirements, and designs emerge from self-organized teams.

12. **Review the Work Regularly** - Review the work done at regular intervals so that the team can reflect on how to become more effective and adjust its behavior accordingly.

3. AGILE – CHARACTERISTICS

Iterative/incremental and Ready to Evolve

Most of the agile development methods break a problem into smaller tasks. There is no direct long-term planning for any requirement. Normally, iterations are planned which are of vary short period of time, for example, 1 to 4 weeks. A cross-functional team is created for each iteration that works in all functions of software development like planning, requirements analysis, design, coding, unit testing, and acceptance testing. The result at the end of the iteration is a working product and it is demonstrated to the stakeholders at the end of an iteration.

After demo, review comments are taken and are planned to be incorporated in the working software as required.

Face-to-face Communication

Each agile team should have a customer representative such as a product owner in scrum methodology. This representative is authorized to act on behalf of the stakeholders and he can answer the queries of the developers in between iterations.

An information radiator (physical display) is normally located prominently in an office, where passers-by can see the progress of the agile team. This information radiator shows an up-to-date summary of the status of a project.

Feedback Loop

Daily stand-up is a common culture of any agile development; it is also known as **daily scrum**. It is a kind of a brief session where each team member reports to each other regarding the status of what they have done, what to do next, and any issues they are facing.

4. AGILE – DAILY STAND-UP

Daily stand-up, as the name suggests, is a daily status meeting among all the members of an agile team. It not only provides a forum for regular updates but also brings the problems of team members into focus so that it can be quickly addressed. Daily stand-up is a must-do practice, no matter how an agile team is established regardless of its office location.

What is Daily Stand-up?

- A daily stand-up is a daily status meeting among all team members and it is held roughly for 15 minutes.
- Every member has to answer three important questions:
  - What I did yesterday?
  - What I'll do today?
  - Any impediment I am facing.../ I am blocked due to...
- Daily stand-up is for status update, not for any discussion. For discussion, team members should schedule another meeting at a different time.
- Participants usually stand instead of sitting so that the meeting gets over quickly.

Why Stand-up is Important?

The benefits of having a daily stand-up in agile are as follows:

- The team can evaluate the progress on a daily basis and see if they can deliver as per the iteration plan.
- Each team member informs all about his/ her commitments for the day.
- It provides visibility to the team on any delay or obstacles.

Who Attends a Stand-up?

- The scrum master, the product owner, and the delivery team should attend the stand-up on a daily basis.
- Stakeholders and Customers are encouraged to attend the meeting and they can act as an observer, but they are not supposed to participate in stand-ups.
- It is the scrum master's responsibility to take note of each team member's queries and the problems they are facing.

Geographically Dispersed Teams

Stand-ups can be done in multiple ways, in case the agile team members are operating from different time zones:

- Select a member on a rotational basis, who can attend the stand-up meeting of teams located in different time zones.

- Have a separate stand-up per team, update the status of the stand-up in a tool such as Rally, SharePoint, Wikis, etc.

- Have a wide variety of communication tools ready like conference call, video conferencing, instant messengers, or any other third-party knowledge sharing tools.

5. AGILE – DEFINITION OF DONE

The definition of **done** for User Story, Iteration, and Release is given below.

User Story

A user story is a requirement which is formulated in a few sentences in everyday language of a user and it should be completed within an iteration. A user story is done when

- All the related code have been checked-in.
- All the unit test cases have been passed.
- All the acceptance test cases have been passed.
- Help text is written.
- Product Owner has accepted the story.

Iteration

An iteration is a time boxed collection of user stories / defects to be worked upon and accepted within the release of a product. Iterations are defined during iteration planning meeting and completed with an iteration demo and review meeting. An iteration is also termed as a **sprint**. An iteration is done when

- Product backup is complete.
- Performance has been tested.
- User stories have been accepted or moved to the next iteration.
- Defects have been fixed or postponed to the next iteration.

Release

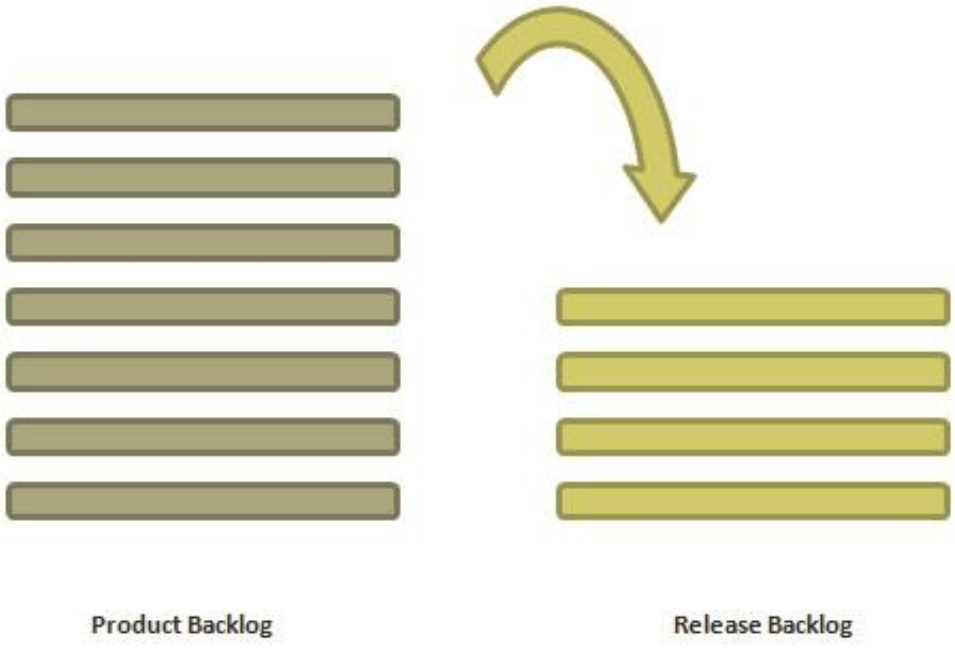
A release is a major milestone that represents an internal or external delivery of working, tested version of the product/system. A release is done when

- System is stress tested.
- Performance is tuned.
- Security validations are carried out.
- Disaster recovery plan is tested.

6. AGILE – RELEASE PLANNING

Release Planning

The purpose of release planning is to create a plan to deliver an increment to the product. It is done after every 2 to 3 months.



Who is Involved?

- **Scrum Master** - The scrum master acts as a facilitator for the agile delivery team.
- **Product Owner** - The product owner represents the general view of the product backlog.
- **Agile Team** - Agile delivery team provides insights on the technical feasibilities or any dependencies.
- **Stakeholders** - Stakeholders like customers, program managers, subject matter experts act as advisers as decisions are made around the release planning.

Prerequisites of Planning

The prerequisites of release planning are as follows:

- A ranked product backlog, managed by the Product Owner. Generally five to ten features are taken which the product owner feels that can be included in a release
- Team's input about capabilities, known velocity or about any technical challenge
- High-level vision
- Market and Business objective
- Acknowledgement whether new product backlog items are needed

Materials Required

The list of materials required for release planning is as follows:

- Posted agenda, purpose
- Flip charts, whiteboards, markers
- Projector, way to share computers having data/tools required during planning meeting
- Planning data

Planning Data

The list of data required to do release planning is as follows:

- Previous iterations or release planning results
- Feedback from various stakeholders on product, market conditions, and deadlines
- Action plans of previous releases / iterations
- Features or defects to be considered
- Velocity from previous releases/ estimates.
- Organizational and personal calendars
- Inputs from other teams and subject matter experts to manage any dependencies

Output

The output of a release planning can be the following:

- Release plan
- Commitment
- Issues, concerns, dependencies, and assumptions which are to be monitored
- Suggestions to improve future release plannings

Agenda

The agenda of a release planning can be:

1. **Opening ceremony** - Welcome message, review purpose and agenda, organizing tools and introduction to business sponsors.

2. **Product Vision, Roadmap** - Show the large picture of the product.

3. **Review previous releases** - Discussion on any item which can impact the plan.

4. **Release name / theme** - Inspect the current status of roadmap themes and do the required adjustments, if any.

5. **Velocity** - Present the velocity for the current release and of previous releases.

6. **Release schedule** - Review key milestones and decision on time boxes for release and iterations within release.

7. **Issues and concerns** - Check any concerns or issue and record them.

8. **Review and Update the Definition of Done** - Review the definition of done and make appropriate changes based on technology, skill, or changes in team members since the last iteration / release.

9. **Stories and items to be considered** - Present the user stories and features from the product backlog to be considered for scheduling in the current release.

10. **Determine sizing values** - If the velocity is unknown, then plan the sizing values to be used in the release planning.

11. **Coarse the size of stories** - The delivery team determines the appropriate size of the stories under consideration and splits the stories into multiple iterations if a story is too large. The product owner and the subject matter experts clarify the doubts, elaborate the acceptance criteria, and make proper story splits. The scrum master facilitates the collaboration.

12. **Map stories to iterations** - The delivery team and the product owner move the stories/defects in the iterations based on the size and velocity. The scrum master facilitates the collaboration.

13. **New concerns or issues** - Check any new issues based on previous experience and record the same.

14. **Dependencies and assumptions** - Check any dependencies/assumptions planned during the release planning.

15. **Commit** - The scrum master calls for the planning. Delivery team and Product owner signal it as the best plan and then commit to move to the next level of planning, that is, iteration planning.

16. **Communication and logistics planning** - Review/Update the communication and logistics planning for the release.

17. **Parking lot** - Process parking lot means all items should be either resolved or set as action items.

18. **Distribute Action items and action plans** - Distribute the action items among their owners, process the action plan.

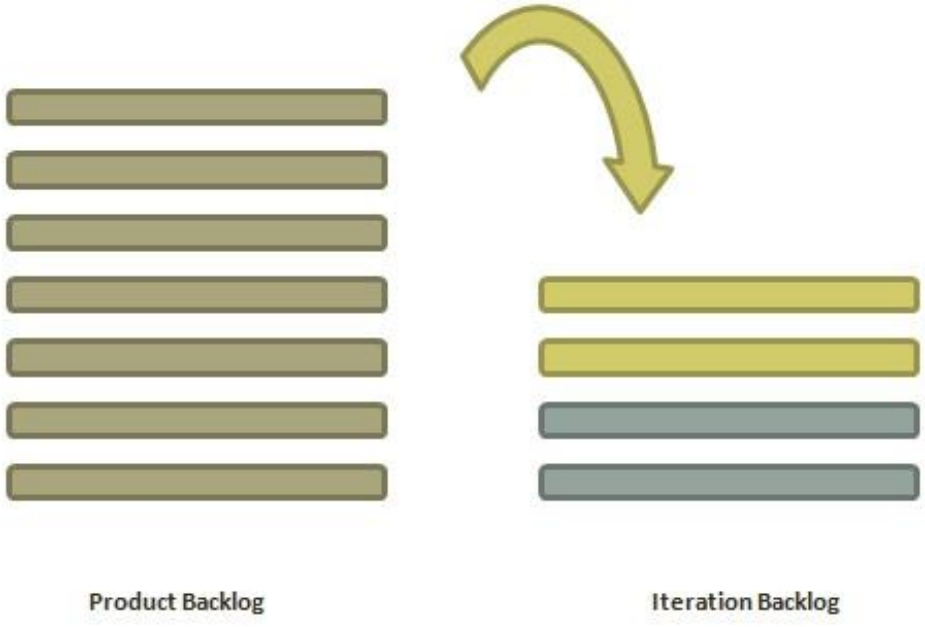
19. **Retrospect** - Solicit feedback from participants to make the meeting successful.

20. **Close** - Celebrate the success.

7. AGILE – ITERATION PLANNING

Iteration Planning

The purpose of iteration planning is for the team to complete the set of top-ranked product backlog items. This commitment is time boxed based on the length of iteration and team velocity.



Who is Involved?

- **Scrum Master** - The scrum master acts as a facilitator for the agile delivery team.
- **Product Owner** - The product owner deals with the detailed view of the product backlog and their acceptance criteria.
- **Agile Team** - Agile delivery defines their tasks and sets the effort estimates required to fulfil the commitment.

Prerequisites of Planning

- Items in product backlog are sized and have a relative story point assigned.
- Ranking has been given to portfolio items by the product owner.
- Acceptance criteria has been clearly stated for each portfolio item.

Planning Process

Following are the steps involved in iteration planning:

- Determine how many stories can fit in an iteration.
- Break these stories into tasks and assign each task to their owners.
- Each task is given estimates in hours.
- These estimates help team members to check how many task hours each member have for the iteration.
- Team members are assigned tasks considering their velocity or capacity so that they are not overburdened.

Velocity Calculation

An agile team calculates velocity based on past iterations. Velocity is an average number of units required to finish user stories in an iteration. For example, if a team took 12, 14, 10 story points in each iteration for the last three iterations, the team can take 12 as velocity for the next iteration.

Planned velocity tells the team how many user stories can be completed in the current iteration. If the team quickly finishes the tasks assigned, then more user stories can be pulled in. Otherwise, stories can be moved out too to the next iteration.

Task Capacity

The capacity of a team is derived from the following three facts:

- Number of ideal working hours in a day
- Available days of person in the iteration
- Percentage of time a member is exclusively available for the team.

Suppose a team has 5 members, committed to work full time (8 hours a day) on a project and no one is on leave during an iteration, then the task capacity for a two-week iteration will be:

$$5 \times 8 \times 10 = 400 \text{ hours}$$

Planning Steps

- Product Owner describes the highest ranked item of product backlog.
- Team describes the tasks required to complete the item.
- Team members own the tasks.
- Team members estimate the time to finish each task.
- These steps are repeated for all the items in the iteration.

- If any individual is overloaded with tasks, then his/her task is distributed among other team members.

8. AGILE – PRODUCT BACKLOG

Product Backlog

A product backlog is a list of items to be done. Items are ranked with feature descriptions. In an ideal scenario, items should be broken down into user stories.

Why Product Backlog is Important?

- It is prepared so that estimates can be given to each and every feature.
- It helps in planning the roadmap for the product.
- It helps in re-ranking the features so that more value can be added to the product.
- It helps in determining what to prioritize first. Team ranks the item and then builds value.

Characteristics of Product Backlog

- Each product should have one product backlog which can have a set of large to very large features.
- Multiple teams can work on a single product backlog.
- Ranking of features is done based on business value, technical value, risk management or strategic fitness.
- Highest ranking items are decomposed into smaller stories during release planning so that they can be completed in future iterations.

9. AGILE – USEFUL TERMS

Acceptance Criteria

It is the conditions set by the product owner or the customer in order to accept a feature to be valid and adhering to their requirements.

Backlog Grooming

It is an ongoing process in which the product manager or the customer manages the product backlog by getting feedback from agile teams. This process involves prioritizing the portfolio items, breaking them in smaller items, planning them for future iterations, creating new stories, updating acceptance criteria or elaborating acceptance criteria in details.

Capacity

It is the amount of work a team can take to complete in one iteration.

Feature

An improvement done to a product or capability of value to stakeholder which can be developed in a release.

Iteration

A theme-based work item that can be completed within a time box and accepted within the release of a product. Iteration work is defined during iteration planning and it finishes with demo and review meeting. It is also termed as Sprint.

Increment

An increment is the changing state of a product as it undergoes gradual development. It is normally represented by milestones or number of fixed iterations.

Product Owner

The product owner is a member of the Agile delivery team, responsible to collect and rank business requirements in the product backlog. A product owner communicates what is to be done in a release/iteration. He/she sets the commitments and is responsible to protect team from any change in requirements during an iteration.

Product Backlog

Set of functional and non-functional product requirements.

Product Backlog Items

May be user stories, defects, features which are to be developed by the agile team.

Points

A common unit used to set the relative size of user stories, features, or any other portfolio items.

Release

A time box where work is done to support delivery of testable increment to a software. In scrum, a release consists of multiple iterations.

Requirement

A specification of a software product to satisfy a stated contract or functionality. User stories and portfolio items are types of requirements.

Story Points

A unit used by the agile team to estimate relative sizes of user stories and features.

Sprint

Same as Iteration.

Timebox

A fixed duration of time in which a deliverable is to be developed. Normally, along with fixing start and end date of a timebox, the number of resources is also fixed.

Task

It is a unit of work that contributes towards the completion of a user story within an iteration. User stories are decomposed into multiple tasks and each task can be divided between team members marking them as owner of the tasks. Team members can take responsibility of each task, update estimates, log work done or to-do as desired.

User Story

A listed acceptance criteria to fulfil certain requirements of a user. It is normally written from the perspective of an end-user.

Velocity

A measure to weight the accepted work in an iteration or timebox. Normally it is the sum of story points accepted in an iteration.