Agile Project Management Overview

PMP® exam prep by edzest

Hello Project Leaders,

Understanding Agile way of Project Management can be a bit tricky, especially if you don’t have experience of working in an Agile setting or if the projects in your organizations are not being managed as per the Agile Mindset.

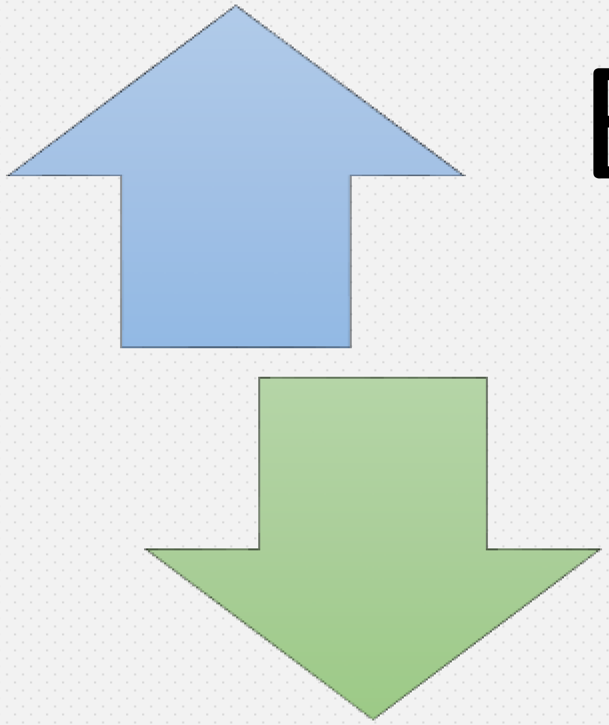
To ensure that you have the right grasp on Agile approach of Project Management, we have created this document that talks about Agile ways of working.

Note that Agile is not a set of processes to be followed, but it’s a mindset- which means you will not see processes and sequence of activities to be followed, as you would in Traditional approach. Rather, we will discuss what and how you can ensure that you deliver value to the customer while leading the team towards success.

All the best for your studies.

(Please reach out to us for any queries or discussions.)

Content



Being Agile

Doing Agile

Agile is a Mindset, hence instead of focusing on following a framework or an approach, agile practitioners should work on developing the mindset that will make them Agile.

Project Chartering

edzest

Chartering in Agile

- Chartering in a project sets the direction, aligns the vision, and empowers the team with shared understanding rather than rigid control.

Components of an Agile Charter

Component	Explanation
Vision Statement	A clear and inspiring description of the desired future state or value the product aims to deliver.
Problem Statement	A brief explanation of the key challenge or unmet need the product is intended to address.
Objectives & Success Criteria	Specific, measurable goals that define what success looks like for the project or product.
Product Scope (initial)	A high-level outline of the key features, functions, or capabilities to be included in the initial version of the product.
Key Stakeholders	Individuals or groups who are affected by the project or can influence its success, including users, sponsors, and team members.
Team Norms	Agreed-upon working agreements and behavioral expectations that guide how the team collaborates.
Constraints	Known limitations or boundaries the team must work within, such as time, budget, technology, or compliance rules.
Risks & Assumptions	Identified uncertainties that could impact the project (risks) and unverified conditions expected to hold true (assumptions).

Visualizing Scope

edzest

Scope Management in Agile

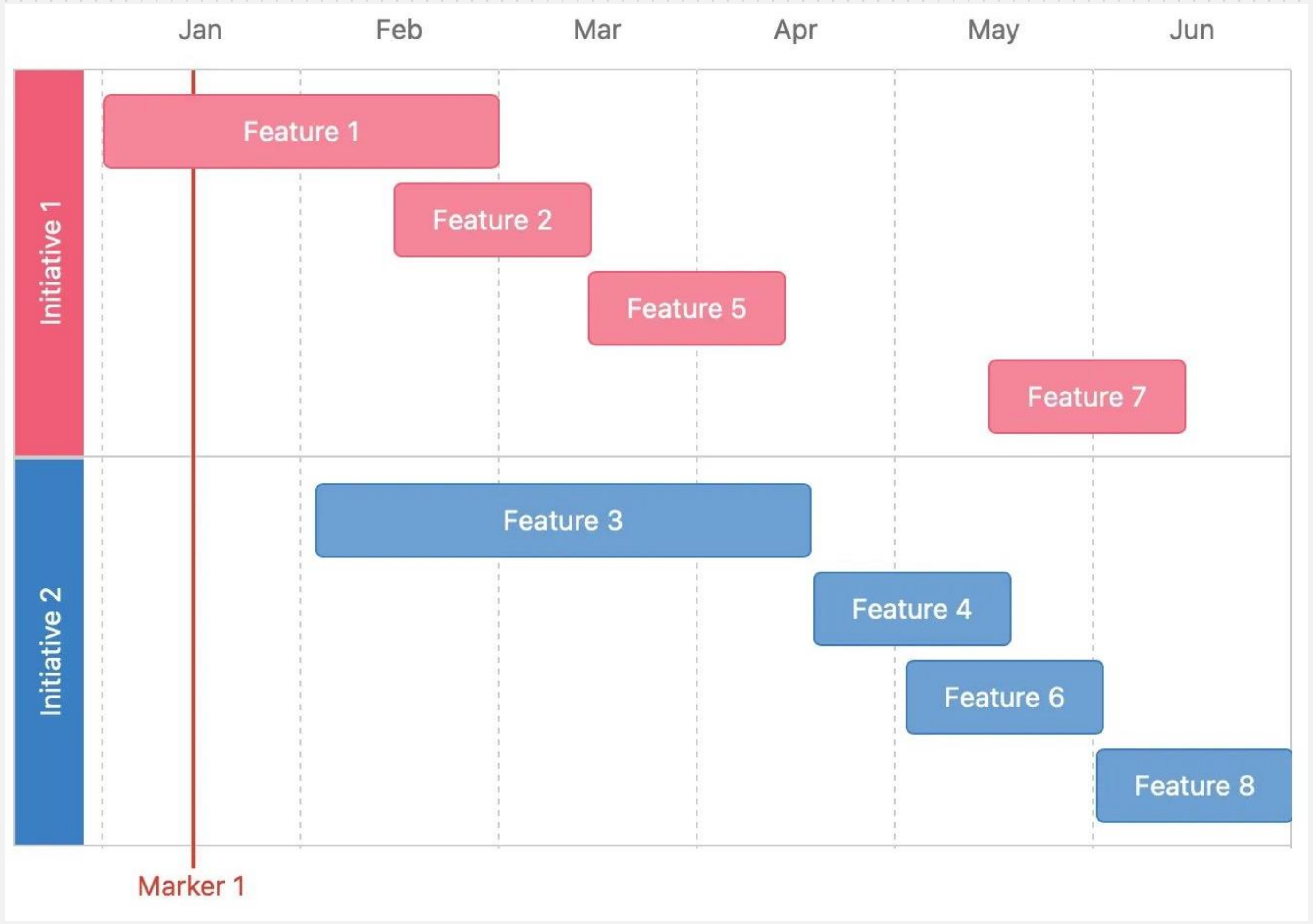
Agile approach is selected in a project to take care of the uncertainties in the requirements, which is expected to grow as the project proceeds.

Since, the requirements are expected to change, it is futile to try to fix and finalize the scope in the beginning. The team might also not know exactly what might be needed in the product. Hence, The team collaborates, carries out multiple workshops, discussion session regularly throughout the project to visualize the scope and refine their understanding as the project needs evolve.

In the following slides we will see the most common ways of visualizing the product- Product roadmap and Product Backlog. We will also discuss what goes into a product backlog- User stories, and how do we estimate the efforts required to complete a User story.

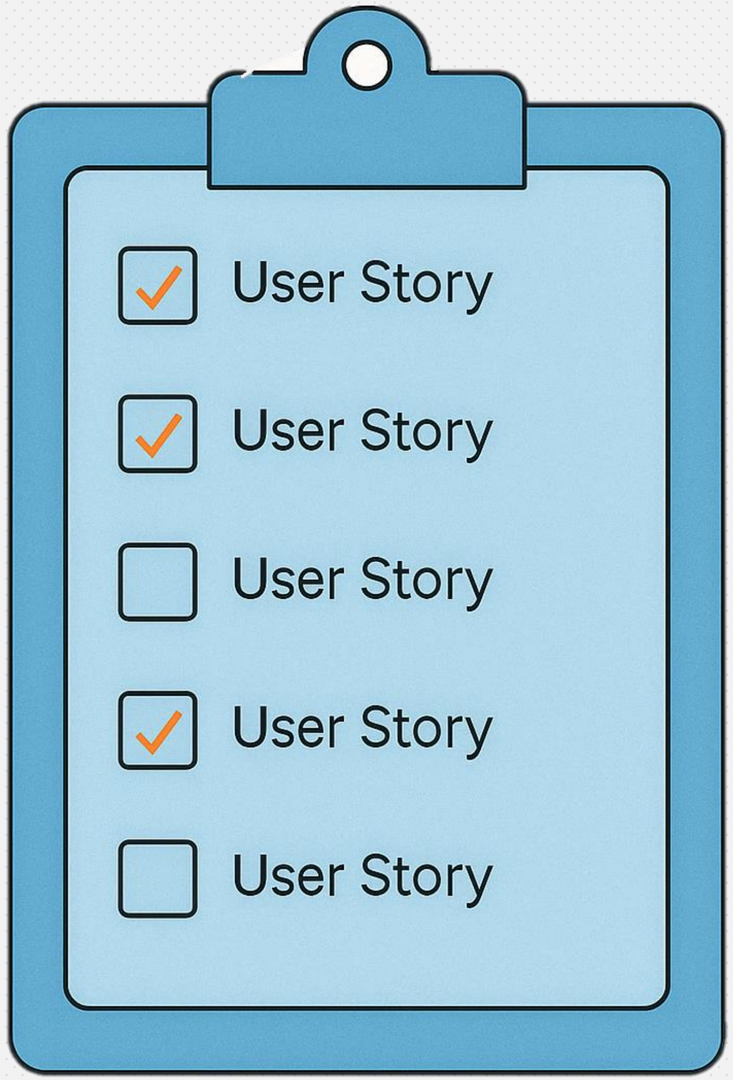
Product Roadmap

A **product roadmap** is a strategic visual document that outlines the vision, direction, priorities, and progress of a product over time, guiding stakeholders on what will be delivered and when.



Product Backlog

A product backlog is an ordered list of all desired features, enhancements, fixes, and technical work required for a product, serving as the single source of work for the Agile team.



edzest

User Stories

Backlog items in Agile are usually written as **user stories**, often following the format:

“As a [user role], I want to [do something] so that [benefit].”

Examples:

- “As a learner, I want to see my percentile compared to others so that I know where I stand.”
- “As an admin, I want to tag questions by difficulty so that we can generate adaptive tests later.”

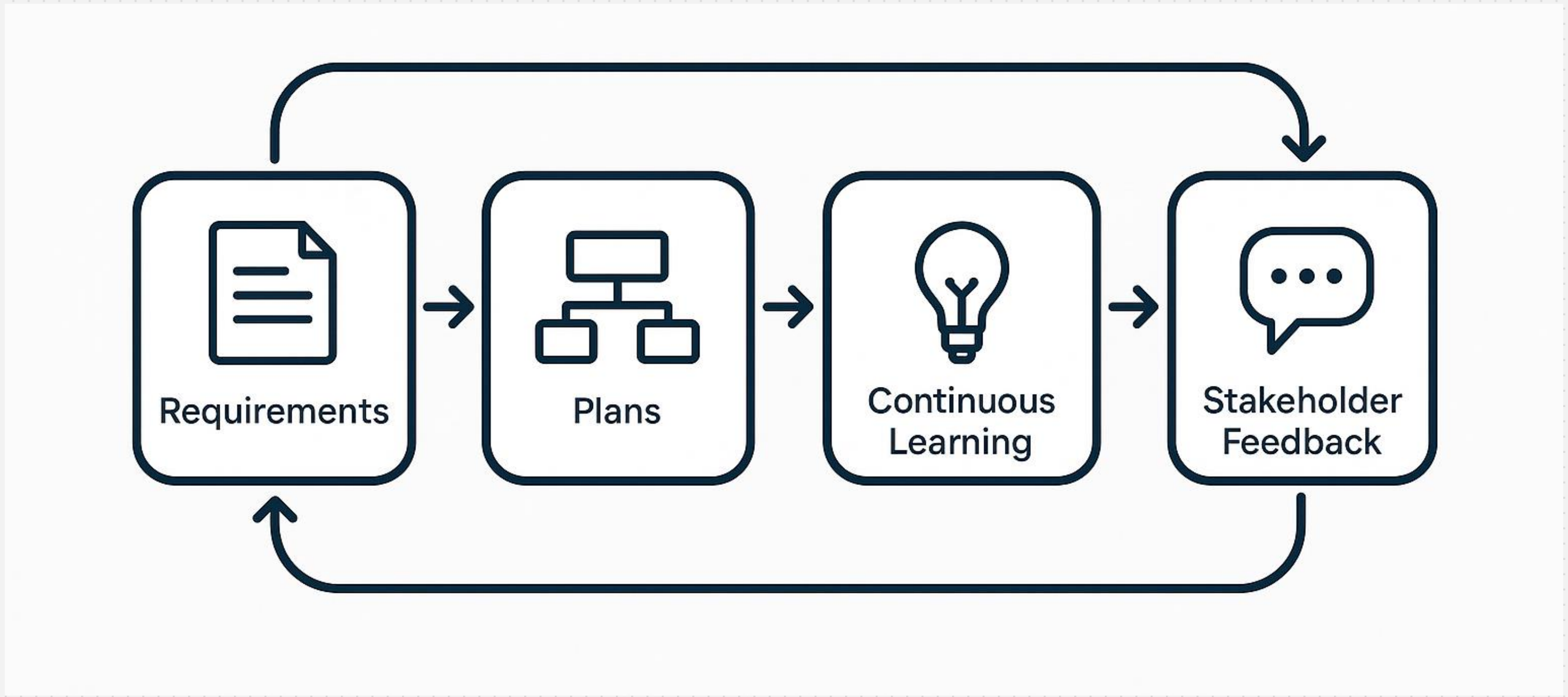
Estimating efforts

Collaborative	Relative	Iterative
Collective effort of user stories are estimated. Estimation should be done with the team members collaborating within themselves and with PO/Team Lead	The collective efforts of User Stories are estimated by **comparing** the effort of an unknown user story to a known user story.	The estimation is done at multiple stages in the project- during initiation, while building a roadmap, during product roamap, and before starting a sprint.

edzest

Progressive Elaboration

The ongoing process of refining and detailing requirements and plans iteratively based on continuous learning, stakeholder feedback, and evolving understanding of the product.



Coarse vs Refined estimating

Aspect	Coarse Estimation	Refined Estimation
Purpose	Early-stage planning, roadmap development	Sprint planning, story readiness
Granularity	High-level (epics/features)	Detailed user stories/tasks
Tools	T-shirt sizing, affinity mapping	Planning poker, story points
Who uses it?	Product Owner + team (release planners)	Entire team (developers, testers, designers)
When?	During roadmap or initial backlog creation	During grooming and sprint planning
Accuracy	Low – directional	Higher – used for commitment

Estimating Techniques

Unit of the collective efforts- Story Points

Unit of measurement used to estimate the effort required to complete a user story

Story points are a relative measure that helps teams assess the complexity, size, and effort of a particular work item compared to others.

Why Story Points?

Story points are used instead of estimating in time because estimating in time, especially for uncertain work has some problems that team members face.

Problems:

People are not good at making absolute estimates

There is a lack of details of work to be done

While estimating, team also doesn’t know who will be carrying out the tasks, hence estimating time might not be accurate.

Story points provides the solution to the problems mentioned.

Solution:

People are better at making comparative estimates

Relative unit such as story helps with useful estimates at different stages of project

It also allows the team to focus on work rather than “this would take me less time” kind of discussions

Sample Product Backlog

Backlog Item (User Story)	Priority	Estimate	Sprint Target
As a user, I want to create an account and log in so that I can access mock exams	High	5 Story Points	Sprint 1
As a user, I want to take a full-length PMP mock test with timer and navigation	High	8 SP	Sprint 2
As a user, I want to view my test score and see correct/incorrect answers	High	5 SP	Sprint 3
As a user, I want to pause and resume an incomplete test	Medium	5 SP	Sprint 4
As a user, I want to see section-wise analytics for each mock	Medium	8 SP	Sprint 4
As an admin, I want to upload questions in bulk using a spreadsheet	High	5 SP	Sprint 3
As a trainer, I want to view learners' performance data in a dashboard	Low	8 SP	Sprint 5
As a user, I want to give feedback after a test is completed	Low	3 SP	Sprint 5

Product Backlog Refinement

Product Backlog Refinement (also known as backlog grooming) is the ongoing process of reviewing, clarifying, and updating product backlog items to ensure they are well-understood, appropriately sized, and prioritized for upcoming sprints.

It typically includes:

- Breaking down epics into smaller user stories
- Clarifying acceptance criteria for every user story
- Estimating effort (e.g., with story points)
- Re-prioritizing the backlog based on business value, risk, or urgency
- Removing outdated or irrelevant items

Ways of Working

edzest

Ways of working

In Agile approach, teams work for a specified period of time with an understanding to release a certain set of features to the production or end user.

This specified time period is known as Release.

Within every release, The team may choose to work in iterative based agile or flow based. Whatever the team decided for the first release, is expected to continue for the remaining duration of the project. This consistency helps in measuring the progress of the project.

Release Planning

Release planning is the process of determining which product features or user stories will be delivered in an upcoming release, aligning business goals, and customer needs within a defined timeframe.

Release planning activities:

1. Establish the business outcome or user value that the release aims to achieve.
2. Identify the highest-priority backlog items that align with the release goal.
3. Discus with team and assess how many user stories can be delivered within the release window.
4. Highlight any technical, content, or team-based dependencies that could affect delivery.
5. Communicate the release plan, expectations, and potential trade-offs for alignment.
6. Establish what must be true for the release to be considered complete and deliverable.
7. Determine how feedback will be collected post-release and how success will be measured.
8. Adjust the roadmap based on release outcomes, learnings, or changes in priority.

Scheduling in Agile

Iterative Scheduling	Progressive elaboration techniques to develop and schedule activities in a **specified time window** called Iterations/sprints (e.g. Scrum)
On-demand/ Flow-based Scheduling	Pull the work from the queue **as their availability allows** and manage the flow (e.g. Kanban)

How to Choose: Scrum or Kanban?

Criteria	Scrum (Iteration)	Kanban (Flow)
Work Type	Feature-heavy, goal-driven, project-like	Support, maintenance, continuous delivery
Cadence	Fixed timeboxes (e.g., 2-week sprints)	Continuous flow
Predictability	Good for planning MVP or releases	Flexible, adapts to demand
Team Maturity	Requires stable, cross-functional teams	Can work with flexible or shared teams
Feedback Loops	Sprint reviews and retrospectives	On-demand, frequent delivery checkpoints

Iteration based- Scrum

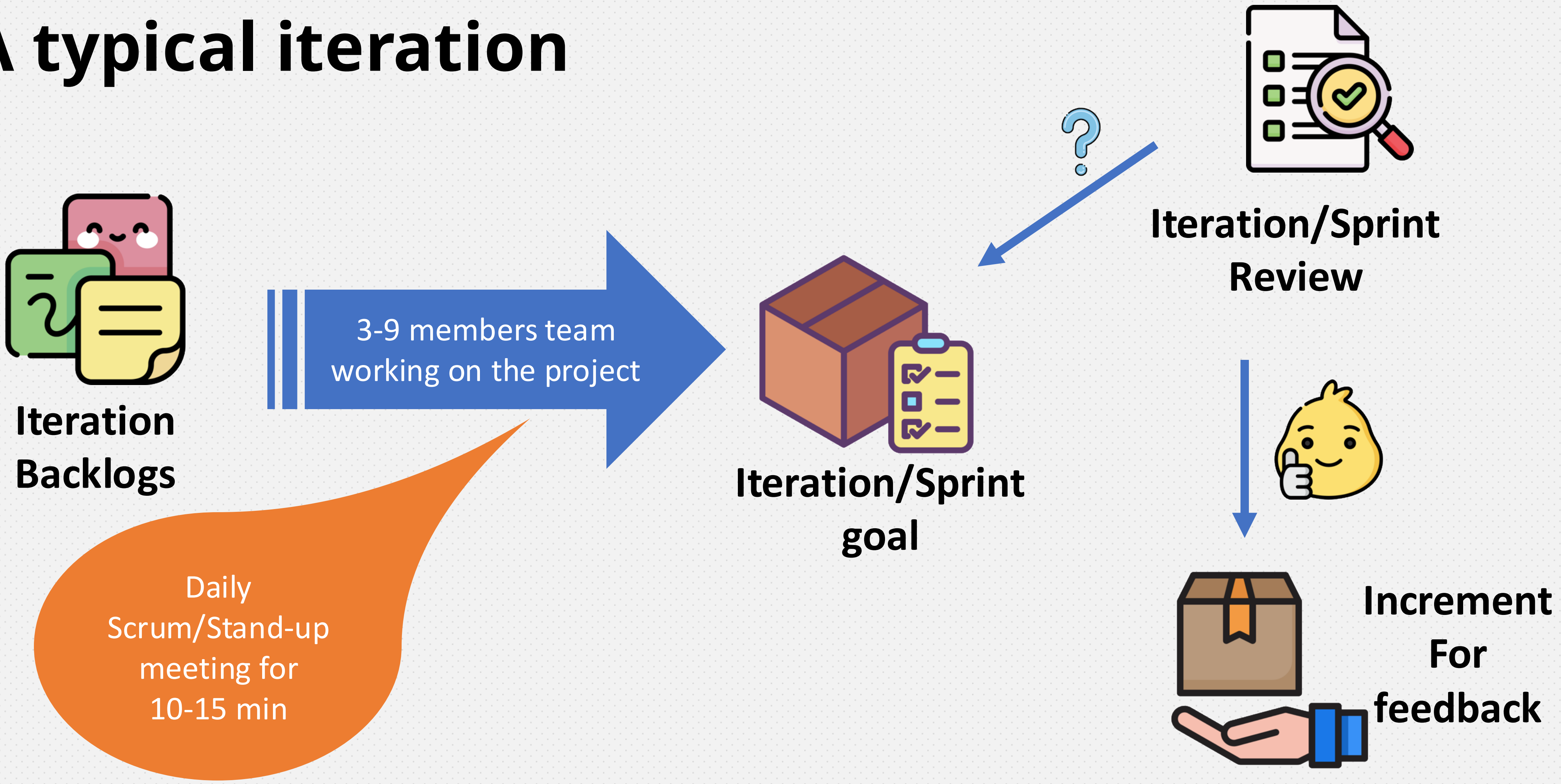
Progressive elaboration: Iterative

January	February	March	April	May	June	July	August	September	October	November	December
Product Vision/Roadmap											
Release 1			**Release 2**		**Release 3**			**Release 4**			
Iterations 1	Iterations 2	Iterations 3	Iterations 4	Iterations 5	Iterations 6	Iterations 7	Iterations 8	Iterations 9	Iterations 10	Iterations 11	Iterations 12
Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings

Typical 12 months calendar with varying release duration and 1 month (4-weeks) sprint

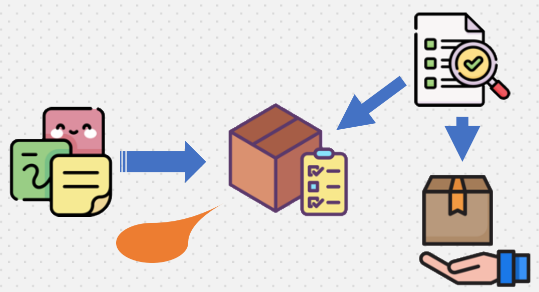
edzest

A typical iteration

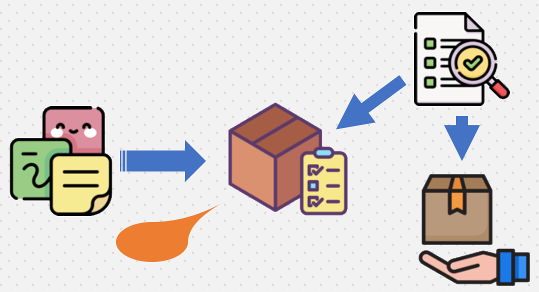


edzest

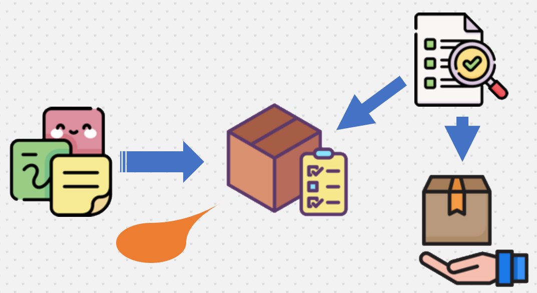
Multiple iterations in a release



Iteration 1



Iteration 2



Iteration 3…

A typical 1-week iteration/sprint

Previous Week

- Plan for the upcoming iteration/sprint
- Ser goals & Commit to deliverables

Monday

- Daily stand-up for sprint kick-off
- Clarify any queries about the work items
- Begin development

Tuesday to Thursday

- Daily stand-up
- Development
- Integration & testing
- Review progress
- Adjustments as needed
- internal reviews

Friday

- Product review/demo
- Retrospective
- Planning for next Sprint

edzest

A typical ~~1-week~~ iteration/sprint

Previous ~~Week~~ iteration

- Plan for the upcoming iteration/sprint
- Ser goals & Commit to deliverables

~~Monday~~ First Day

- Daily stand-up for sprint kick-off
- Clarify any queries about the work items
- Begin development

~~Tuesday to Thursday~~ 2nd to 2nd last day

- Daily stand-up
- Development
- Integration & testing
- Review progress
- Adjustments as needed
- internal reviews

~~Friday~~ Last Day

- Product review/demo
- Retrospective
- Planning for next Sprint

edzest

Key elements of Scrum (Iteration)

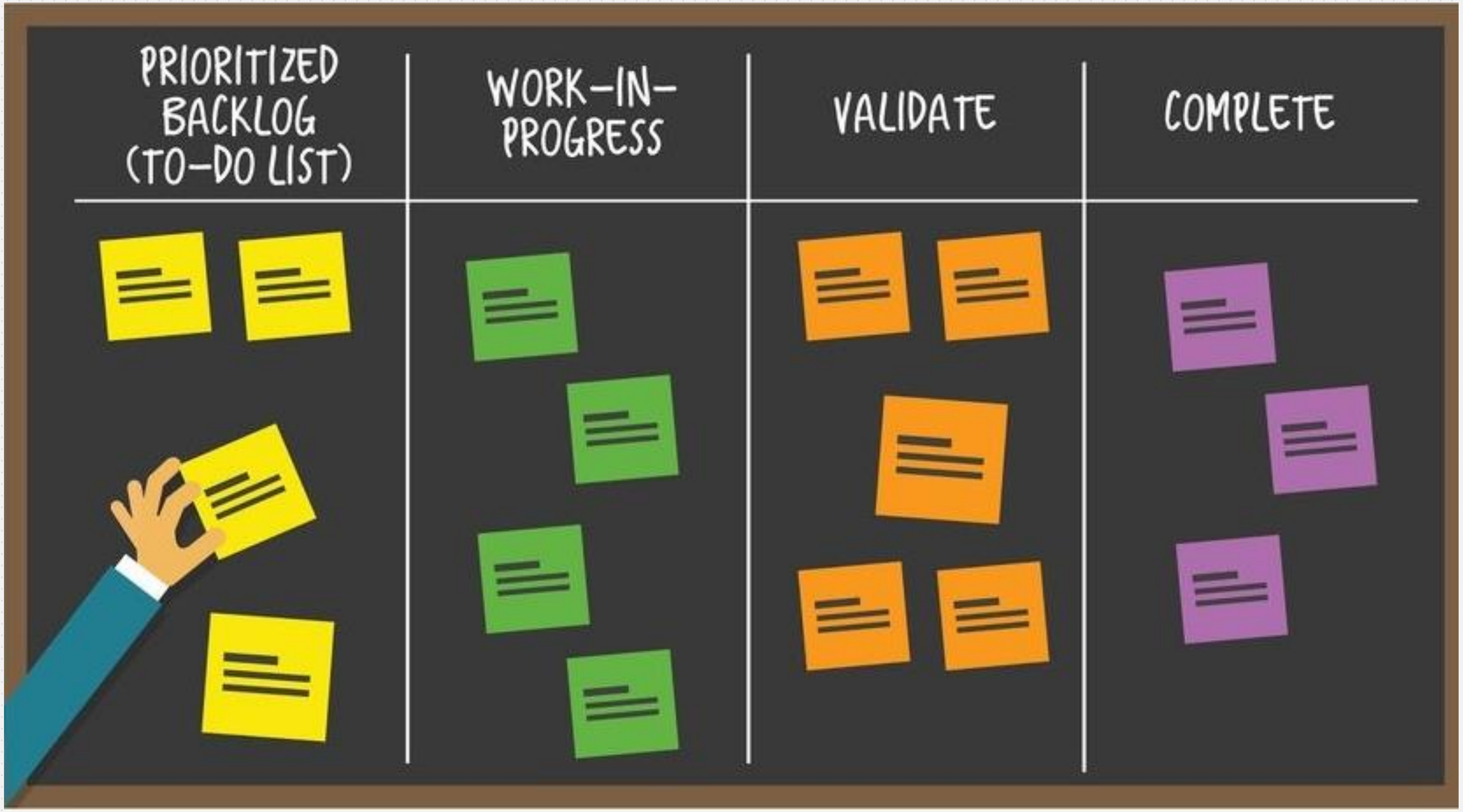
Scrum Element	Description	Example for Edzest
Sprint Planning	Decide what to deliver in a 2-week sprint	Sprint Goal: Build login and user dashboard
Daily Scrum	15-min sync on progress and blockers	Devs meet at 10 AM to discuss progress on test screen
Sprint Review	Demonstrate completed features	Show score report to stakeholder, gather feedback
Sprint Retrospective	Reflect and improve team process	Identify communication gaps or overcommitment
Product Backlog	Ordered list of stories/features	Maintained by Amit (PO)
Sprint Backlog	Stories committed for the sprint	Login, dashboard UI, backend connection
Definition of Done	Agreement on completion criteria	Coded, tested, demo-ready

Flow based- Kanban

Progressive elaboration: Flow based

January	February	March	April	May	June	July	August	September	October	November	December
Product Vision/Roadmap											
Release 1			**Release 2**		**Release 3**			**Release 4**			
Flow using Kanban board			Flow using Kanban board		Flow using Kanban board			Flow using Kanban board			
Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings	Daily meetings

Kanban Board



Key elements of Kanban

Kanban Element	Description	Example for Edzest
Visual Board	Tasks visualized on board (To Do, Doing, Done)	Trello or Jira board for bug fixes, UI updates
Work In Progress (WIP) Limits	Cap on number of concurrent tasks	Max 3 items in ‘In Progress’ column
Pull-based System	Team members pick work as capacity allows	Dev finishes one task before pulling next
Explicit Policies	Rules for moving work across stages	Testing required before ‘Done’
Cycle Time	Time to complete one task	Avg: 3 days from 'To Do' to 'Done'
Throughput	Tasks completed in a time period	e.g., 10 tasks/week
Lead Time	Time from request to delivery	Used to improve service-level expectations

edzest

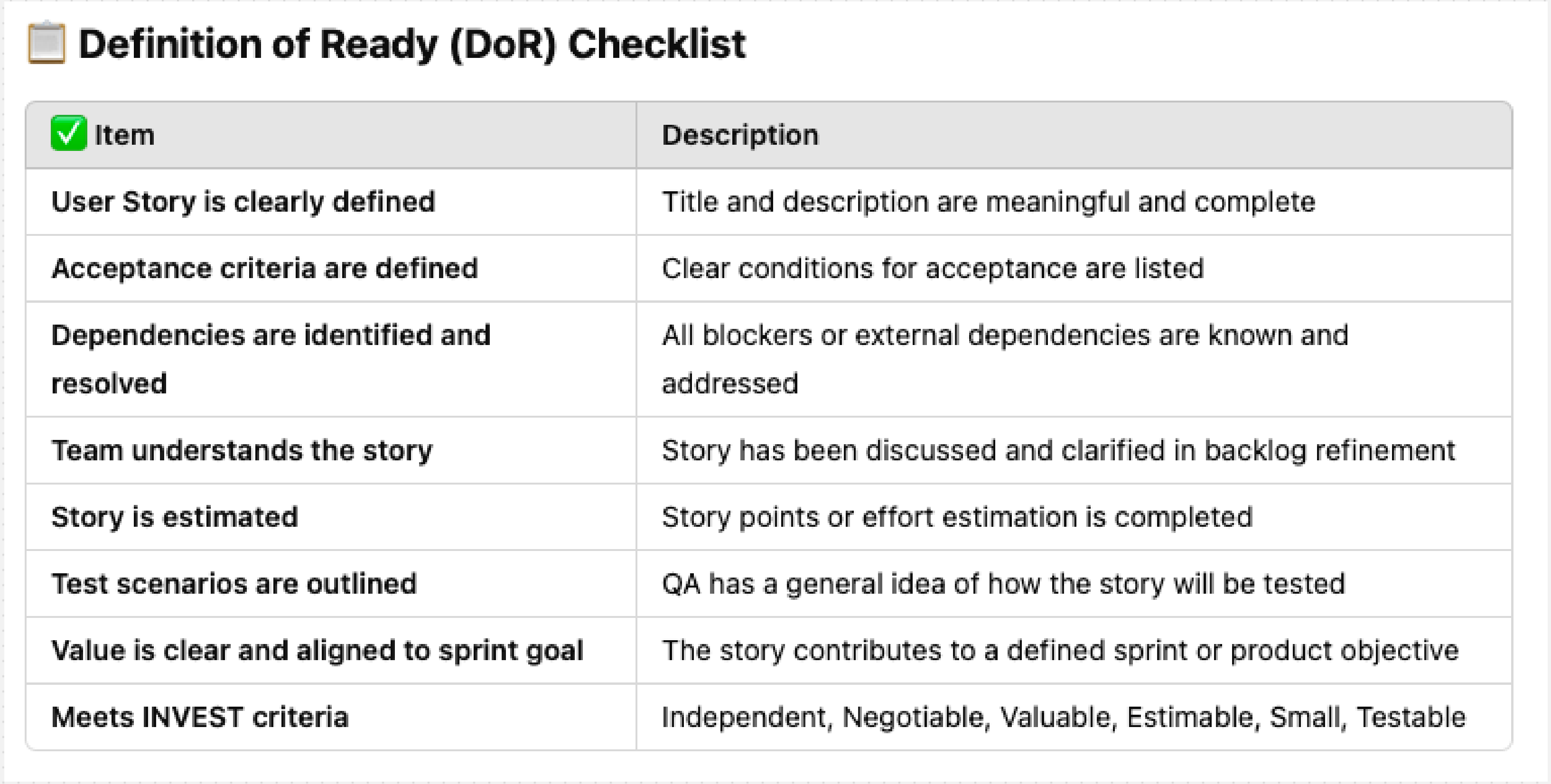
Measurement in Agile

Agile Measurement

- Different from traditional measurement
- Empirical (features finished) > Surrogate (percent done)
- Quantitative + Qualitative Measurement

Definition of ready

A checklist of criteria that a user story must meet before it can be pulled into a sprint for development.

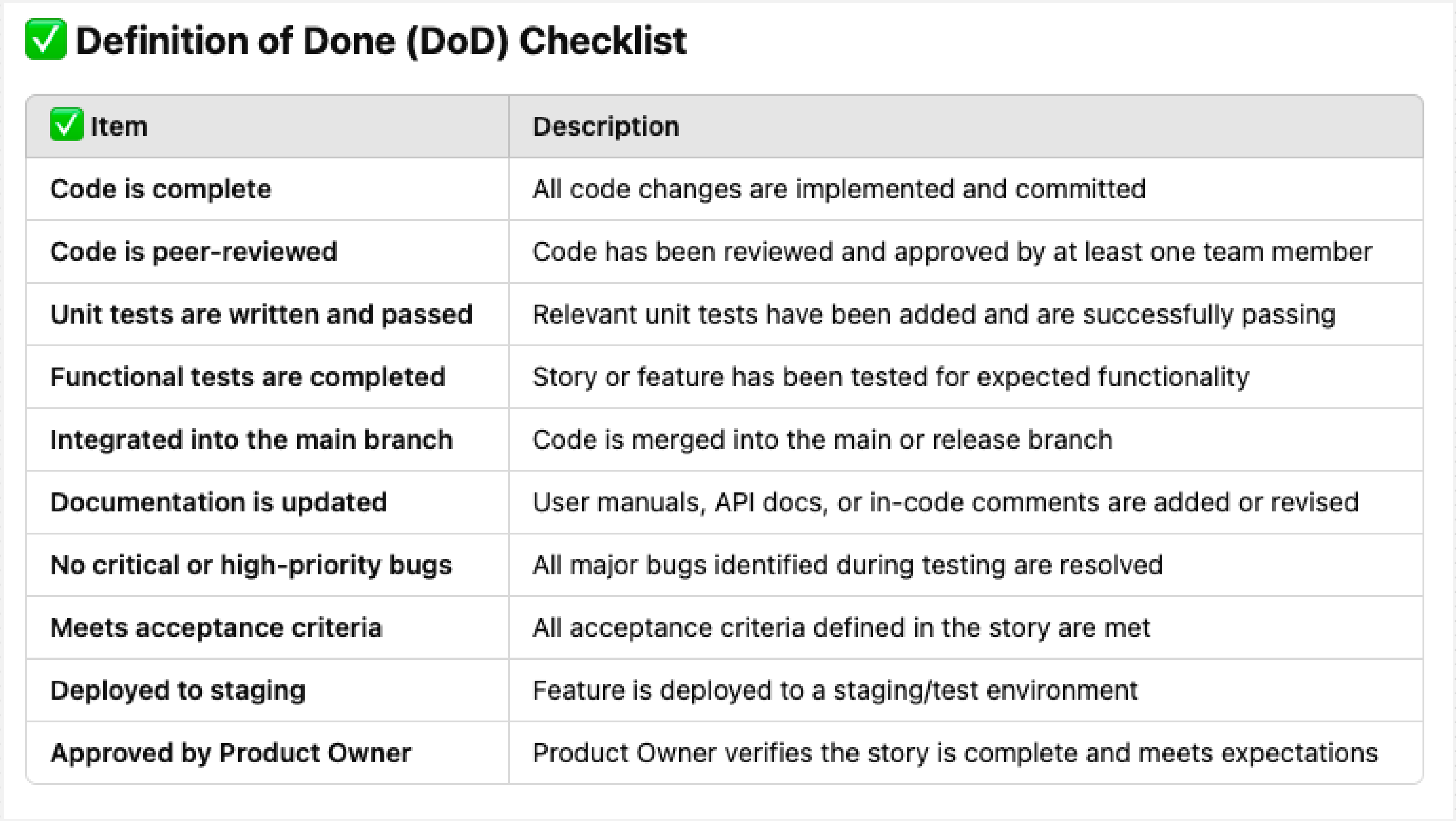


📋 **Definition of Ready (DoR) Checklist**

✅ Item	Description
User Story is clearly defined	Title and description are meaningful and complete
Acceptance criteria are defined	Clear conditions for acceptance are listed
Dependencies are identified and resolved	All blockers or external dependencies are known and addressed
Team understands the story	Story has been discussed and clarified in backlog refinement
Story is estimated	Story points or effort estimation is completed
Test scenarios are outlined	QA has a general idea of how the story will be tested
Value is clear and aligned to sprint goal	The story contributes to a defined sprint or product objective
Meets INVEST criteria	Independent, Negotiable, Valuable, Estimable, Small, Testable

Definition of Done

A shared agreement that specifies the conditions a product increment must satisfy to be considered complete.



✅ Definition of Done (DoD) Checklist

✅ Item	Description
Code is complete	All code changes are implemented and committed
Code is peer-reviewed	Code has been reviewed and approved by at least one team member
Unit tests are written and passed	Relevant unit tests have been added and are successfully passing
Functional tests are completed	Story or feature has been tested for expected functionality
Integrated into the main branch	Code is merged into the main or release branch
Documentation is updated	User manuals, API docs, or in-code comments are added or revised
No critical or high-priority bugs	All major bugs identified during testing are resolved
Meets acceptance criteria	All acceptance criteria defined in the story are met
Deployed to staging	Feature is deployed to a staging/test environment
Approved by Product Owner	Product Owner verifies the story is complete and meets expectations

Iteration review

A meeting held at the end of a sprint where the team demonstrates completed work to stakeholders and gathers feedback.

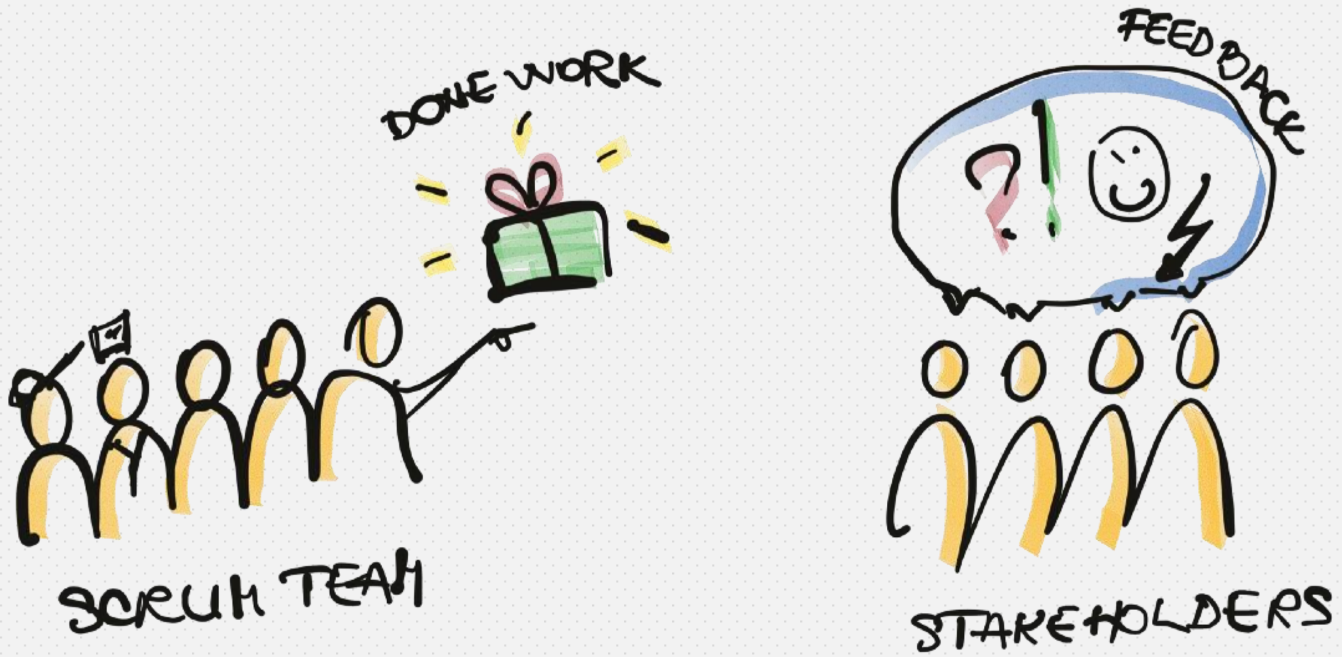
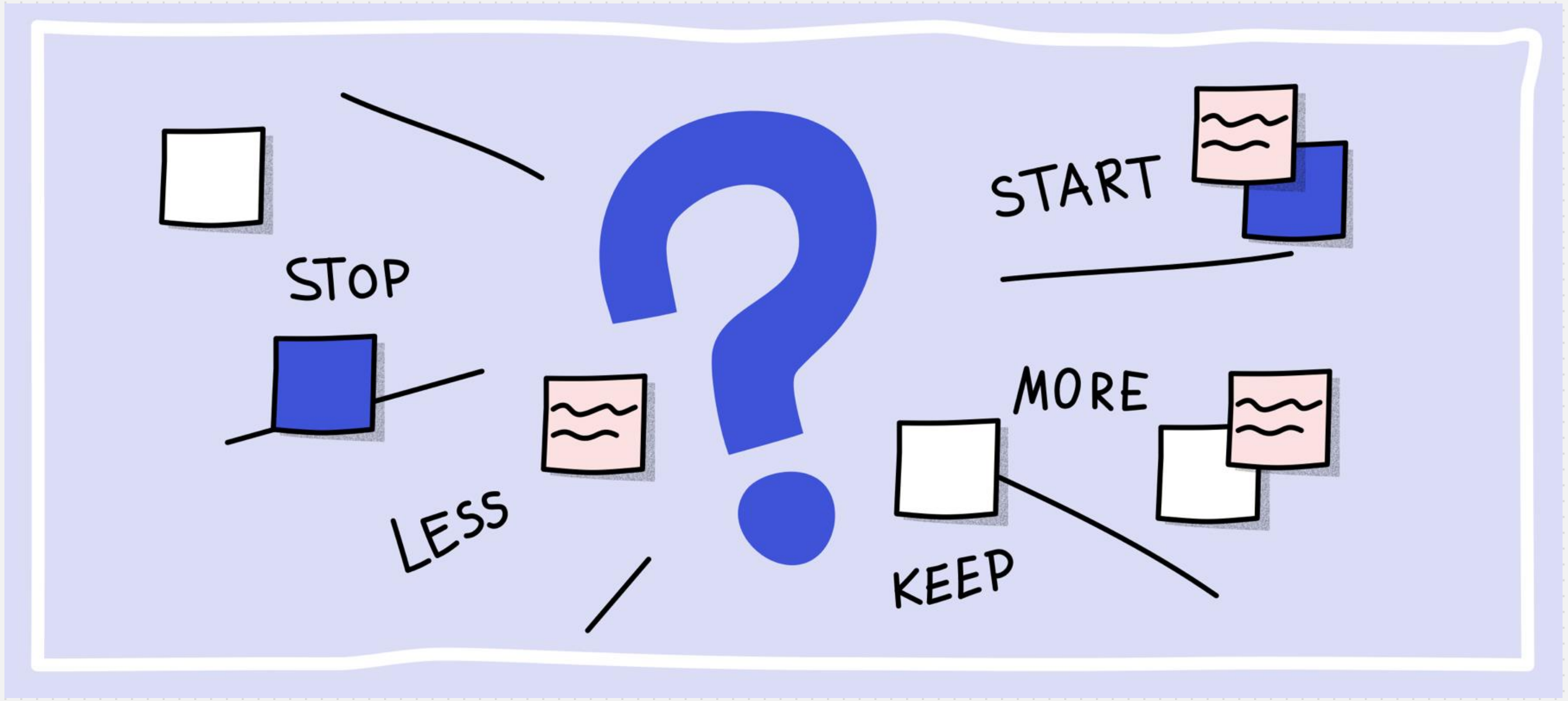


Image source: Anca Onuta on medium

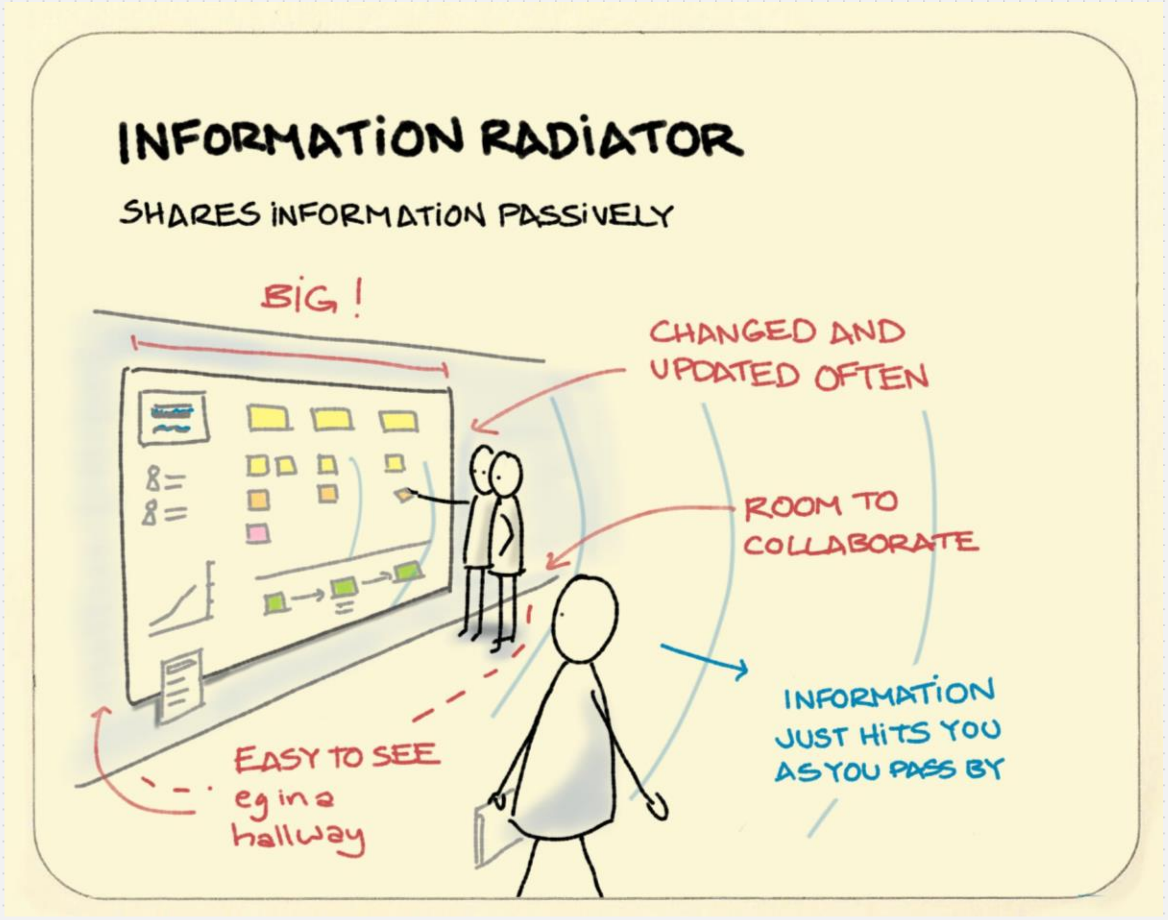
Retrospectives

A team ceremony held after each sprint to reflect on the process, identify what went well, and decide on improvements



Information Radiator

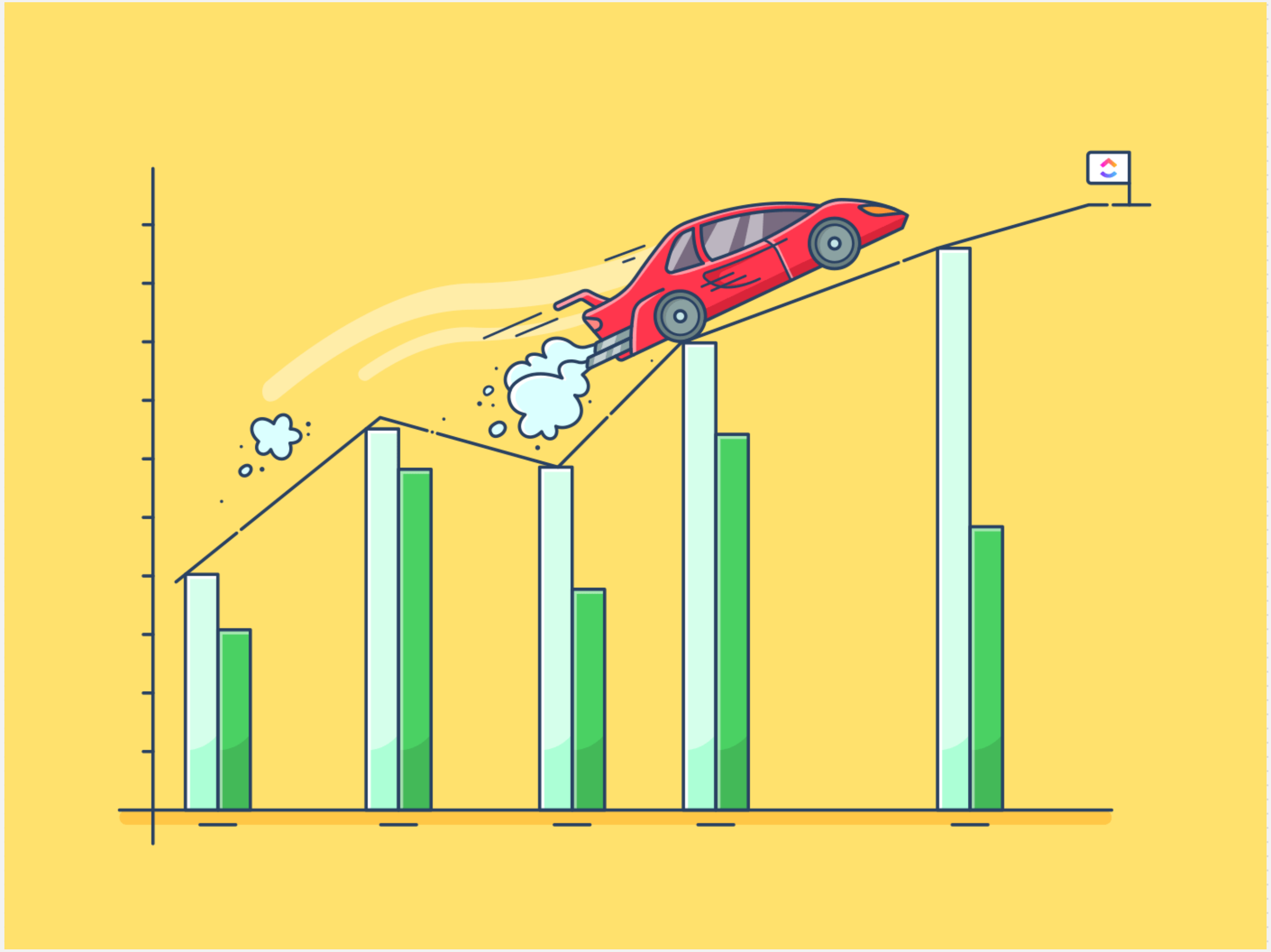
A visible and frequently updated display that shows key project information to promote team transparency and awareness.


INFORMATION RADIATOR
SHARES INFORMATION PASSIVELY
BIG!
CHANGED AND UPDATED OFTEN
ROOM TO COLLABORATE
INFORMATION JUST HITS YOU AS YOU PASS BY
EASY TO SEE eg in a hallway

Velocity

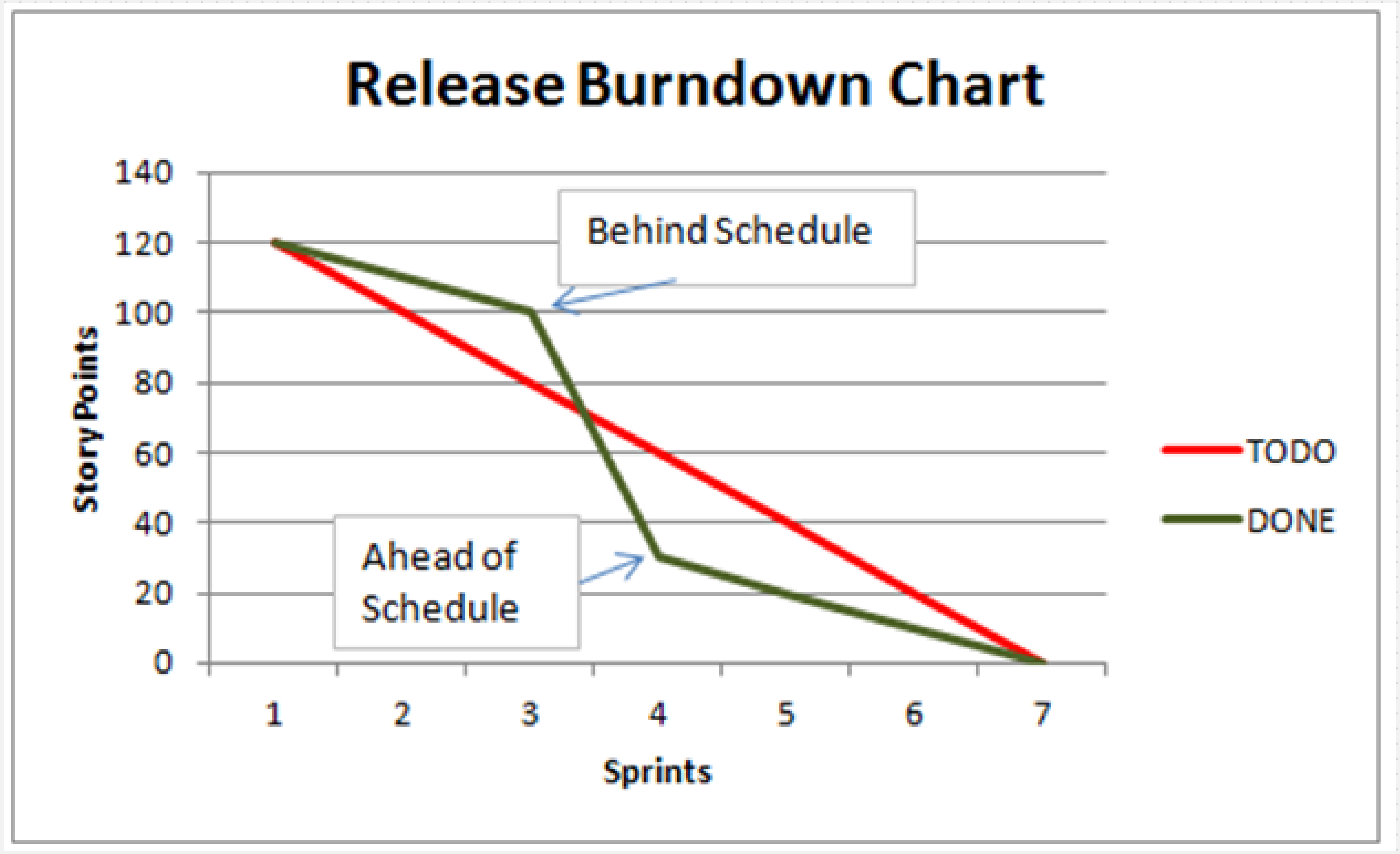
Velocity- Number of Story points completed per iteration

Velocity chart is a graphical representation of the number of story points completed by a team in each sprint, used to forecast future performance.



Burn-down charts

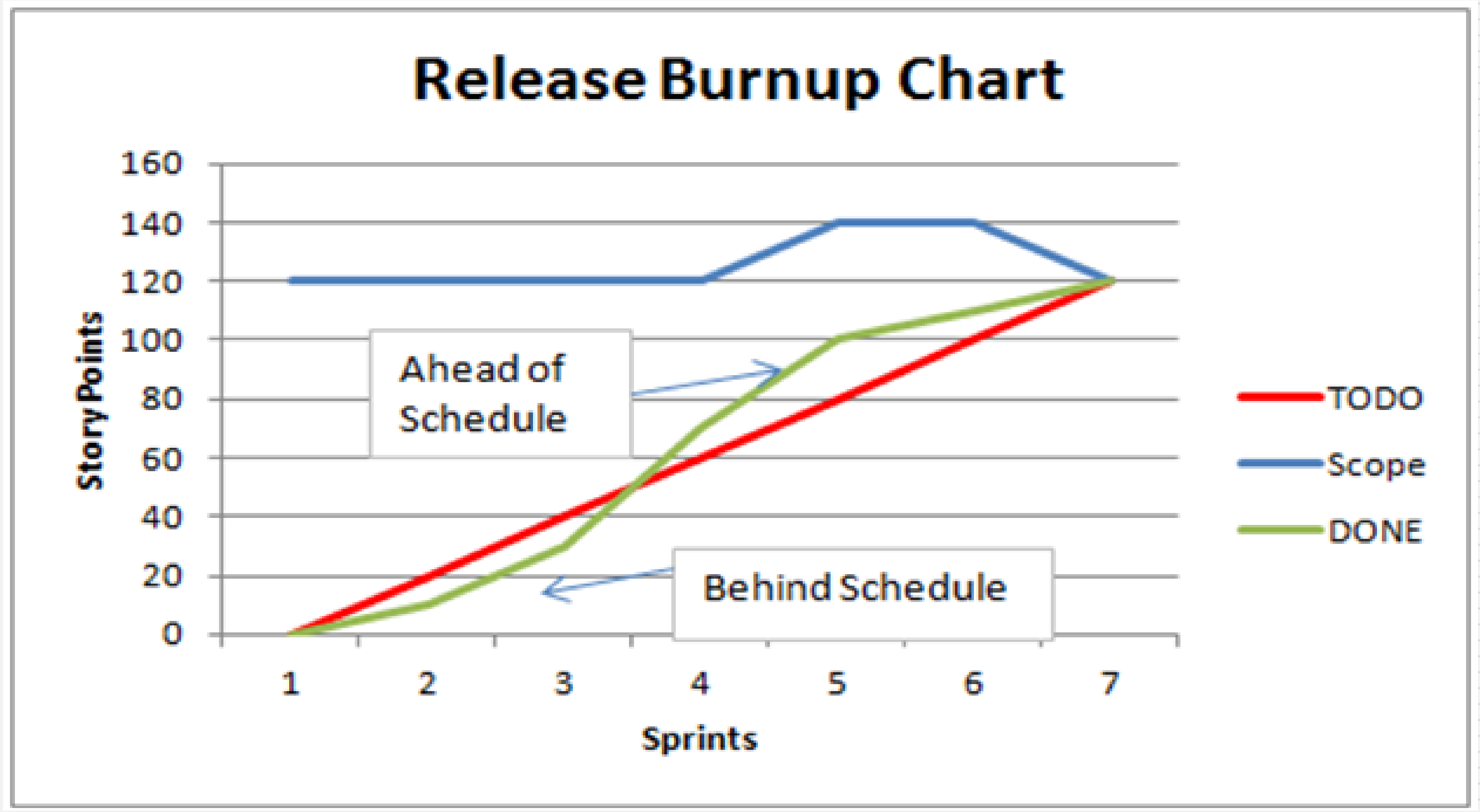
A chart that shows the amount of remaining work versus time to help monitor sprint or release progress.



edzest

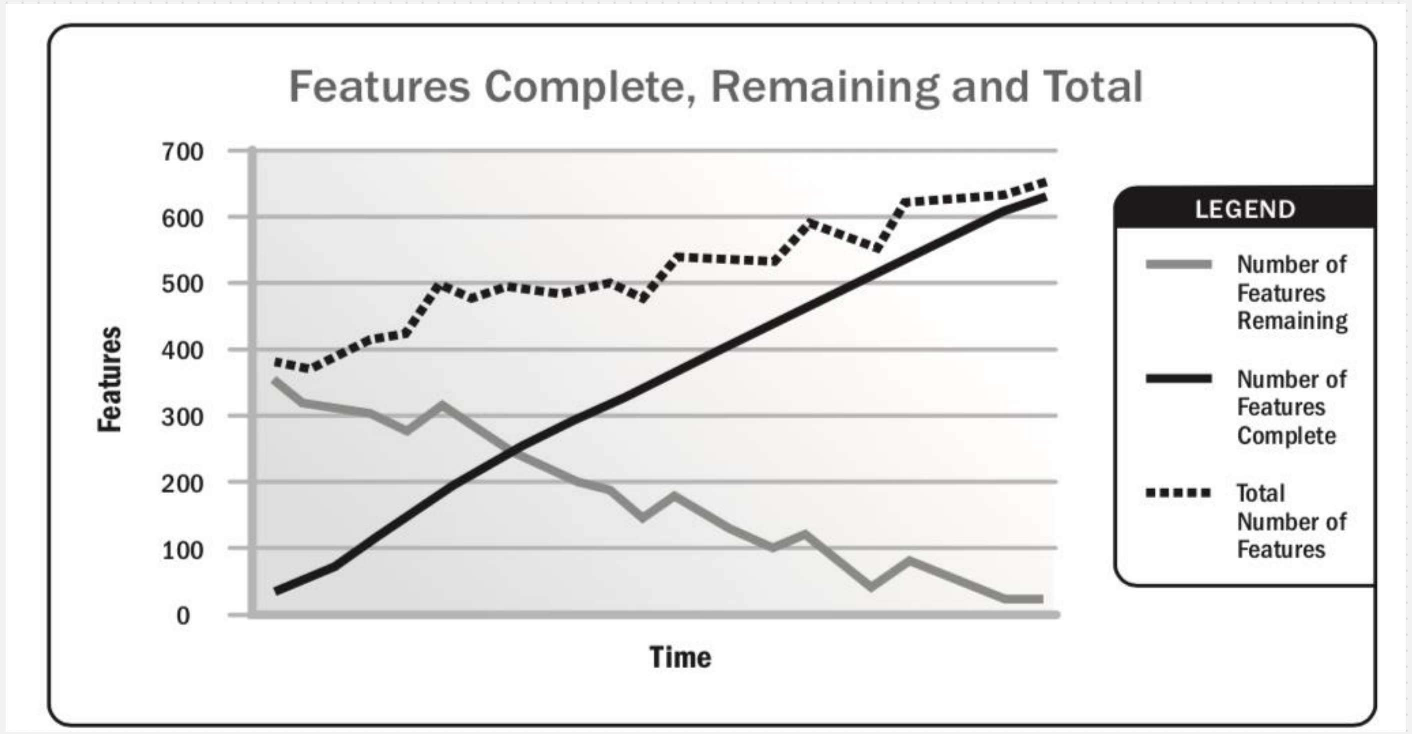
Burn-up charts

A visual tool showing both total scope and completed work over time to track progress toward a release goal.



Feature Charts

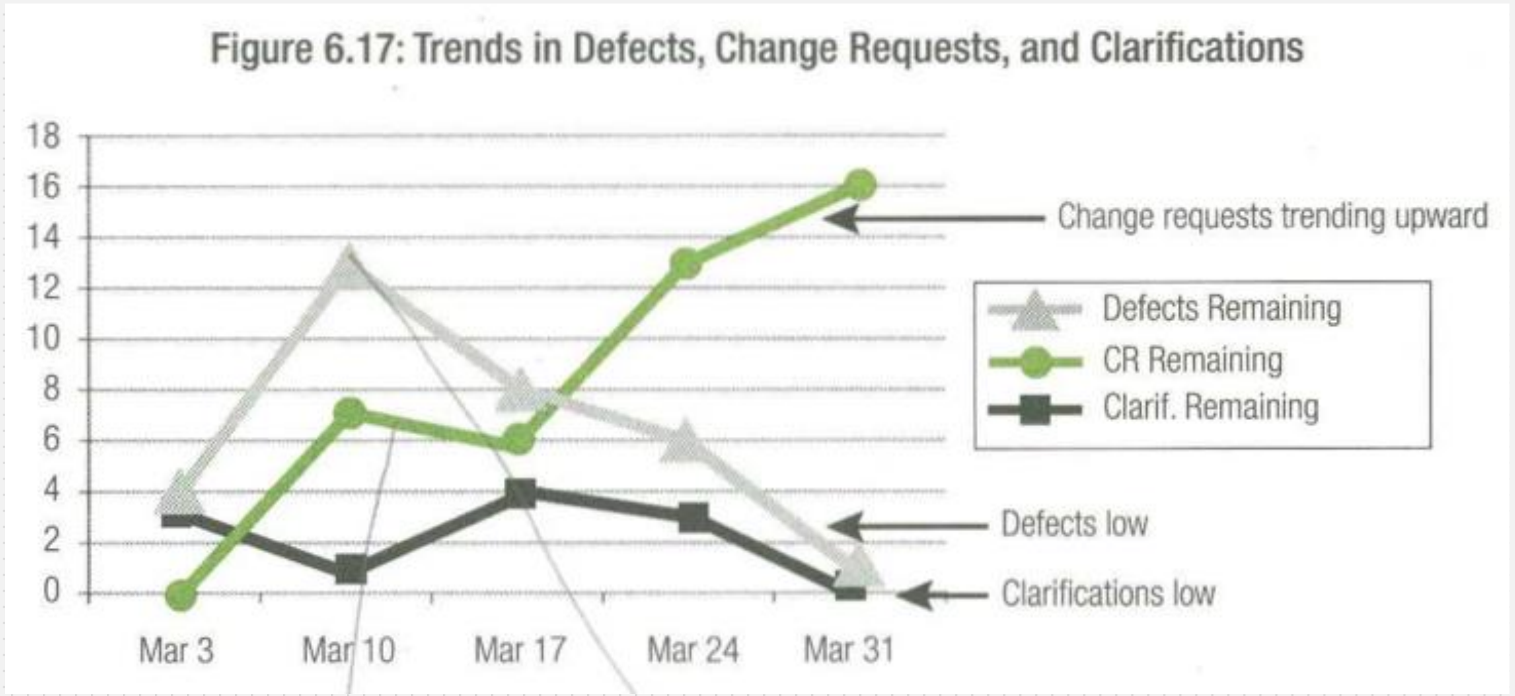
A graph showing how many features have been completed over time, typically used in release or roadmap tracking.



Source: PMBOK 6th edition

Trend Analysis

The practice of examining historical data to identify patterns that help predict future outcomes and team performance.



Source: PMI-ACP exam prep by Mikegriffiths

Thank you

edzest