DevOps Fundamentals

Plan & Track

Agenda

- Scrum
- Kanban
- Best Practices
- GitHub Issues & Projects

moOngy.

Continuous Planning

Practice that requires planners, architects and agile teams to define their plans working as a whole

On-going plan, based on feedback, integrating new ideas and requirements

Main tasks: Definition, refinement, decomposition, prioritization

Rely on Agile Methodologies to better fit on plan & track requirements

Teams need to be autonomous as possible to take their decisions with real impact

Teams need to be aligned with organization strategy to embrace DevOps Culture

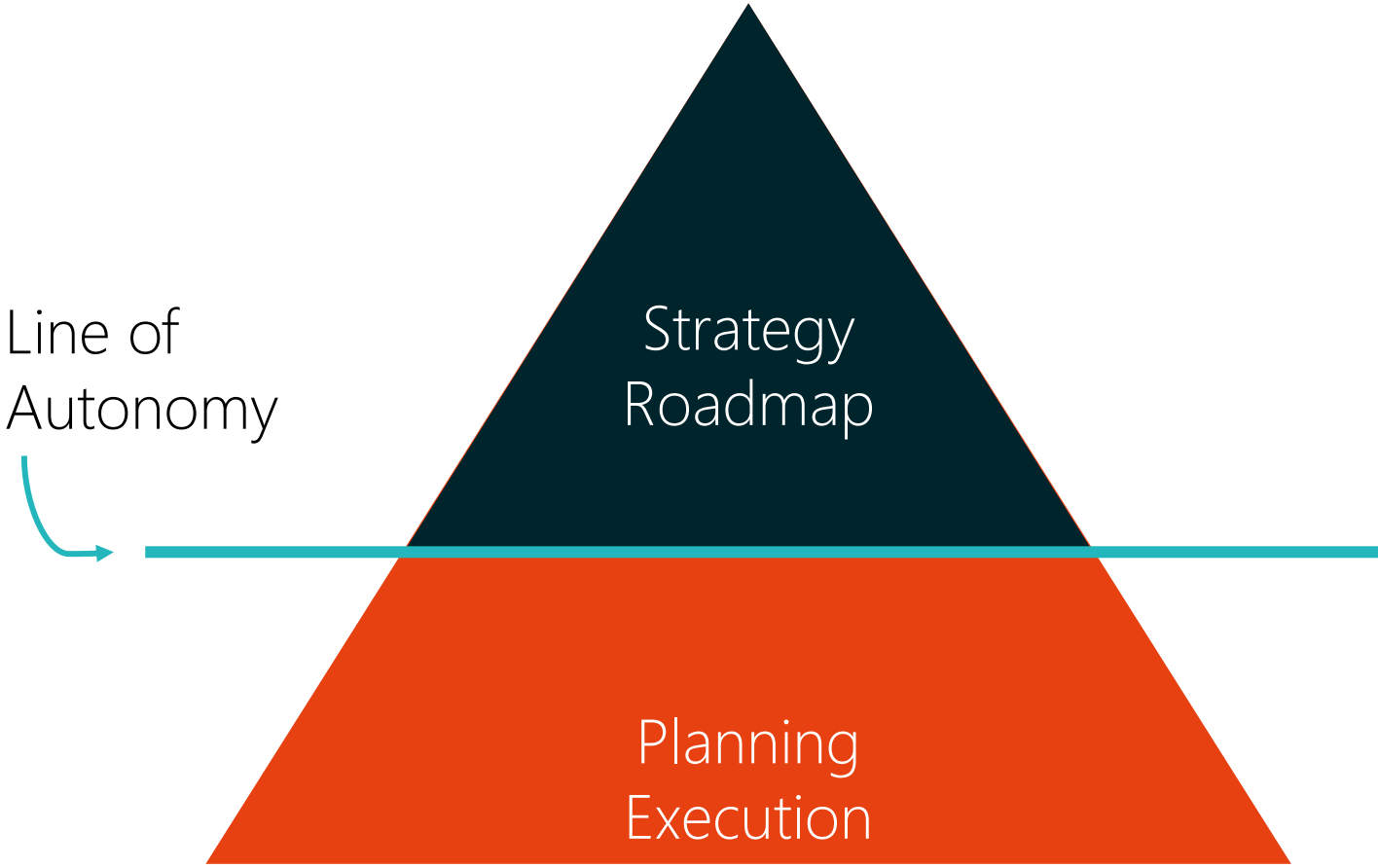
Continuous Planning

Autonomy

- Plan
- Practices

Alignment

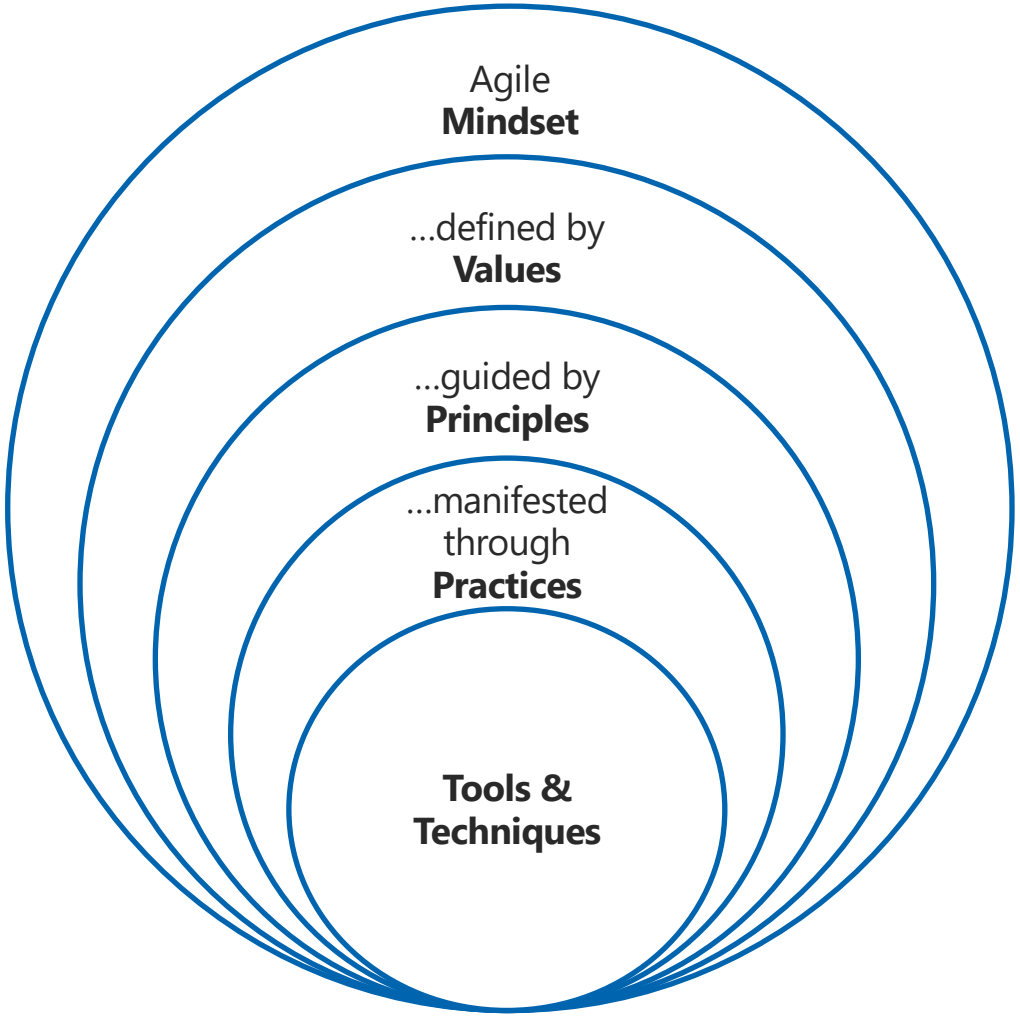
- Organization
- Roles
- Teams
- Cadence
- Taxonomy



Where to put Line of Autonomy?

- Too much alignment, teams are dependent and lack innovation
- Too much autonomy, makes hard to create patterns and culture

Agile Layers (Agile Onion)

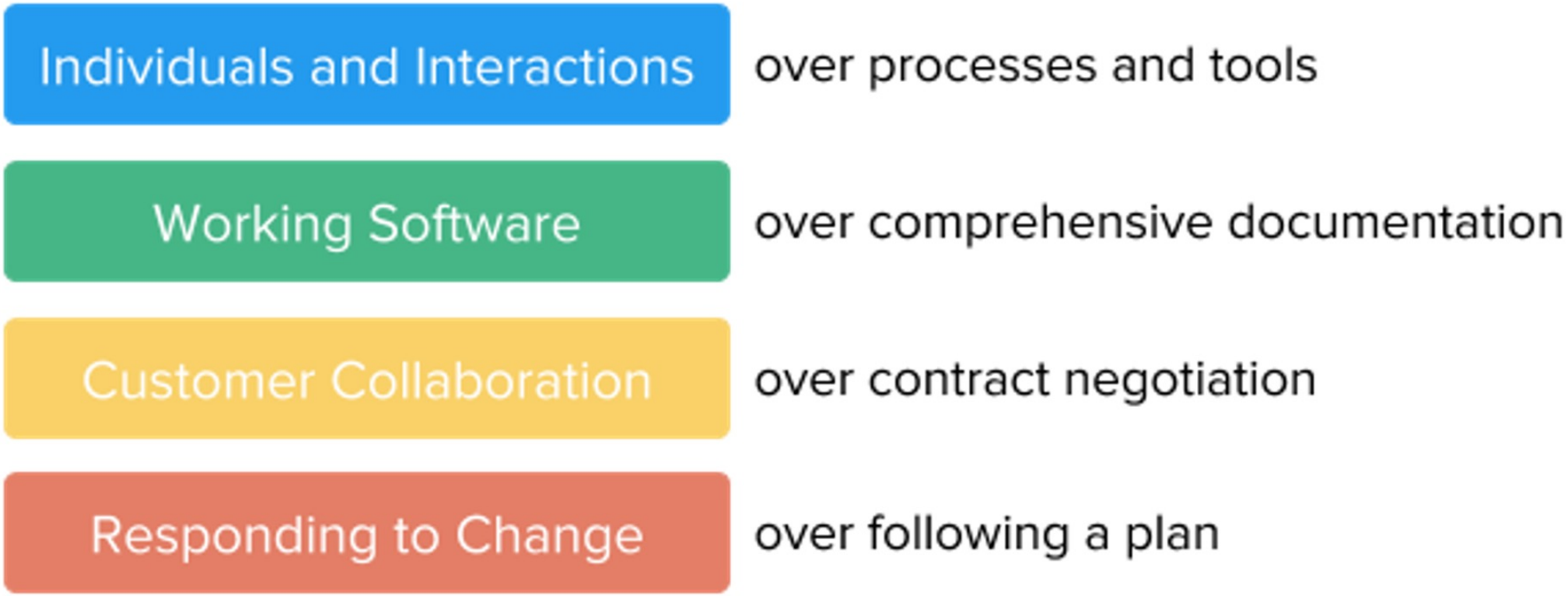


Agile Manifesto

Scrum, SAFe
XP, DSDM, ...

Scrum Board,
Estimation
Techniques, ...

Agile Values



Agile Principles

12 AGILE PRINCIPLES

01 Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	02 Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	03 Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
04 Business people and developers must work together daily throughout the project.	05 Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.	06 Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
07 Working software is the primary measure of progress.	08 The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.	09 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.

[Manifesto for Agile Software Development (agilemanifesto.org)]

moOngy.

Agile Principles

moOngy.

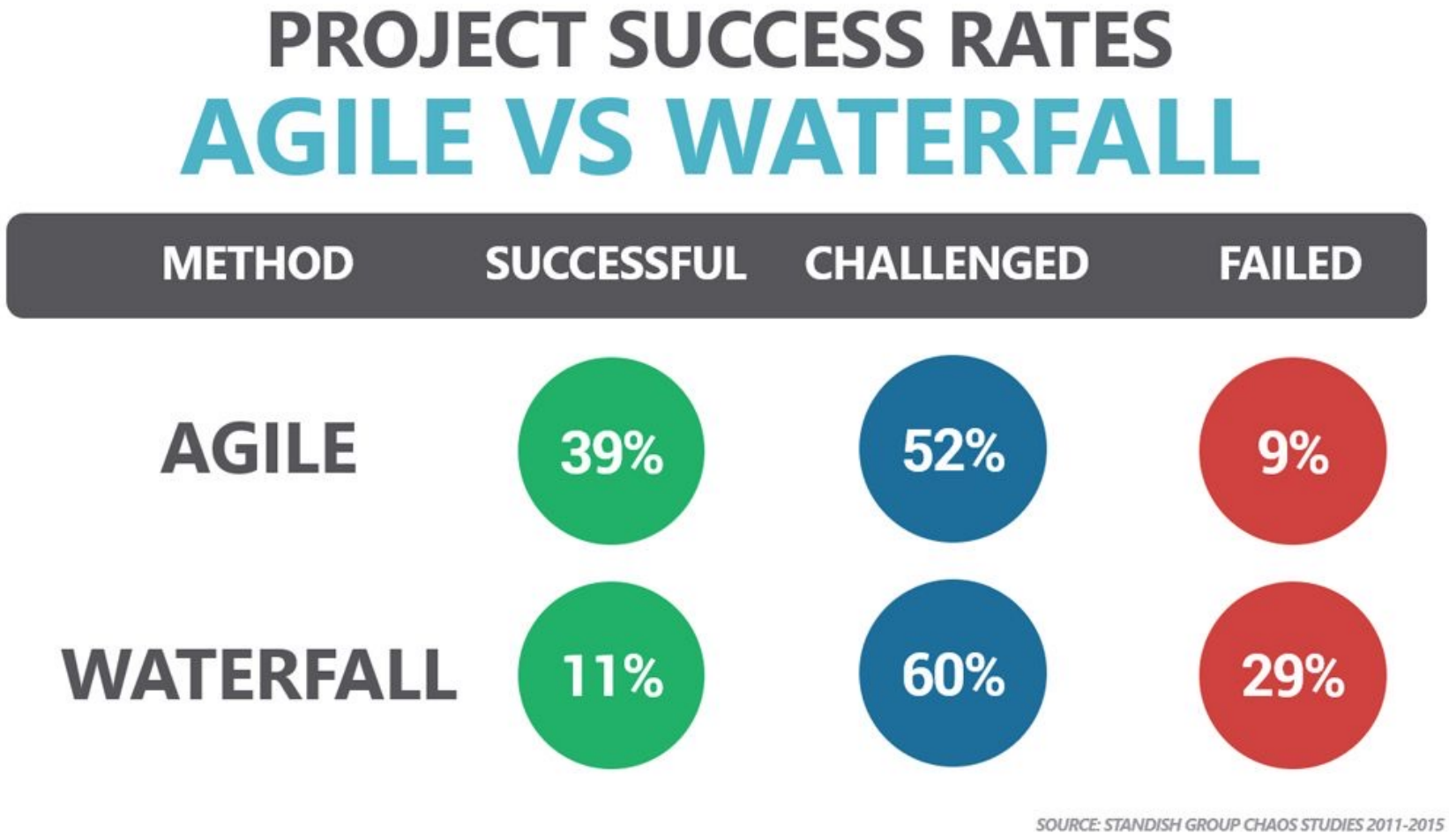
Frameworks

Several frameworks implement these principles defining tools & techniques

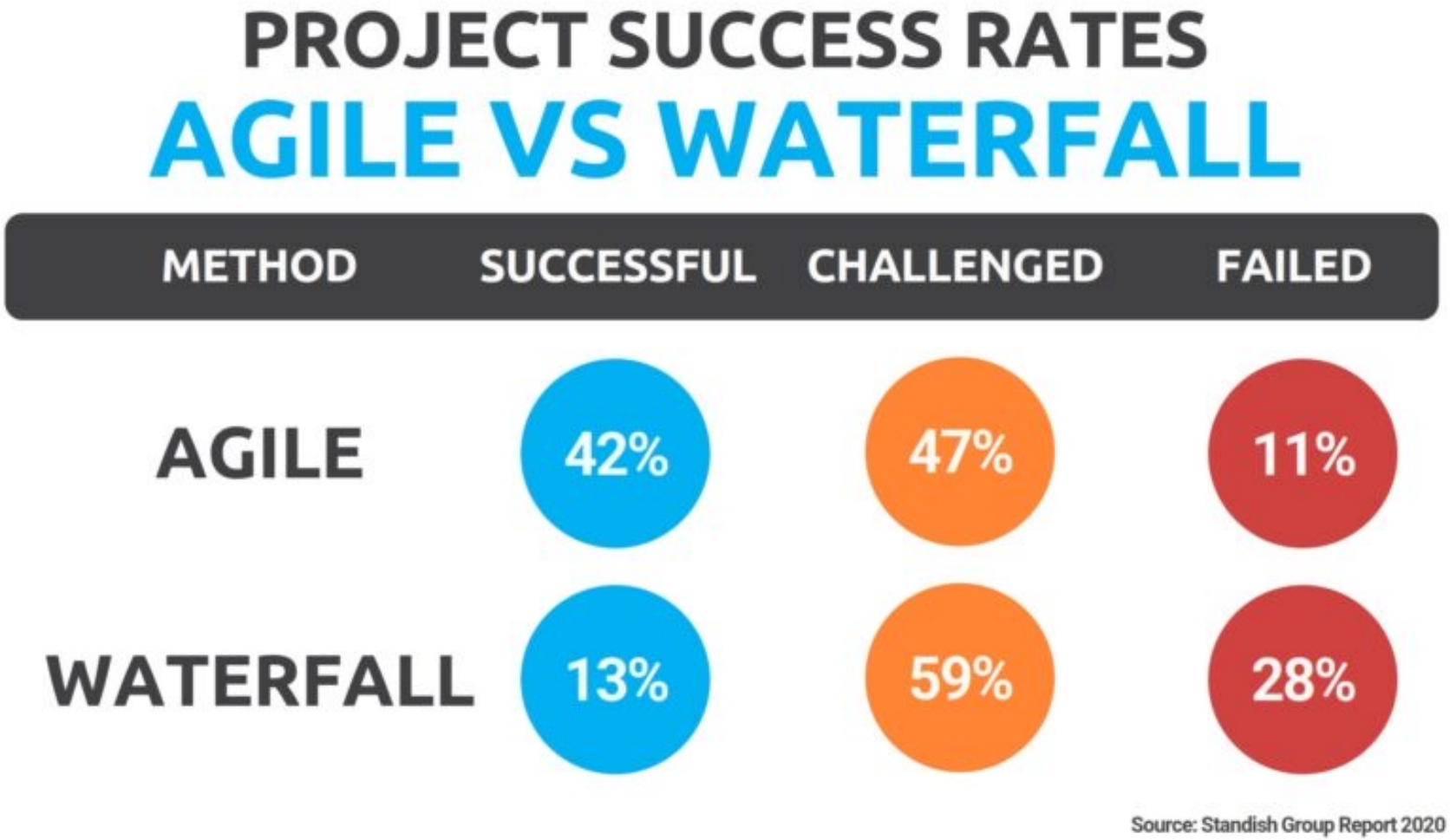
- Scrum
- eXtreme Programming (XP)
- Lean
- Scaled Agile Framework (SAFe)
- Crystal Clear
- Kanban

Nevertheless, the most used framework is “Scrum, but” or “My own implementation of Scrum” ☺

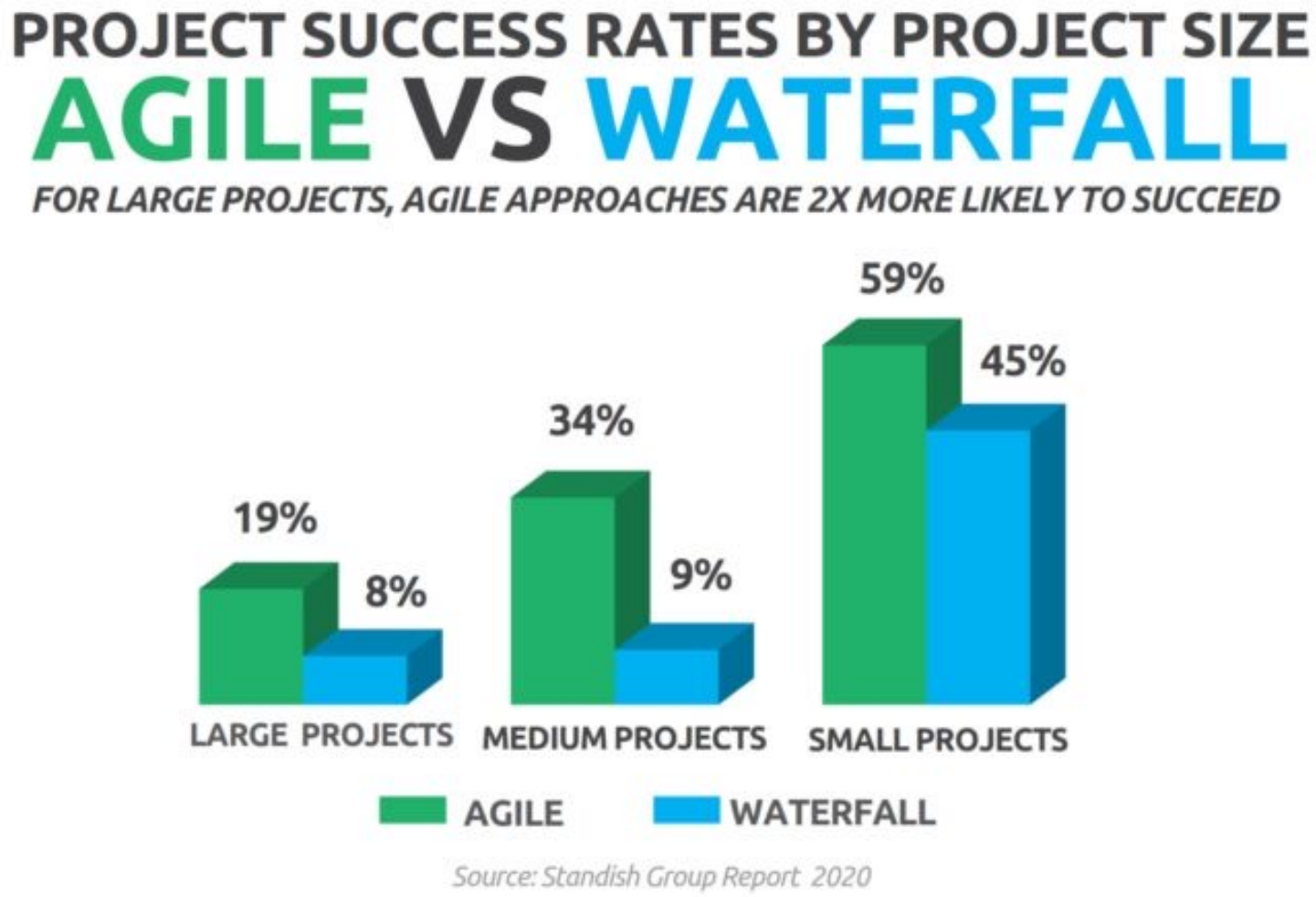
Comparison: Project Success Rates (2015)



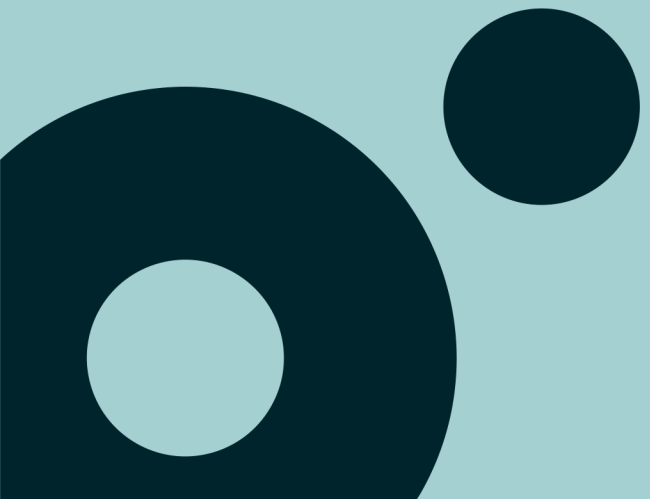
Comparison: Project Success Rates (2020)



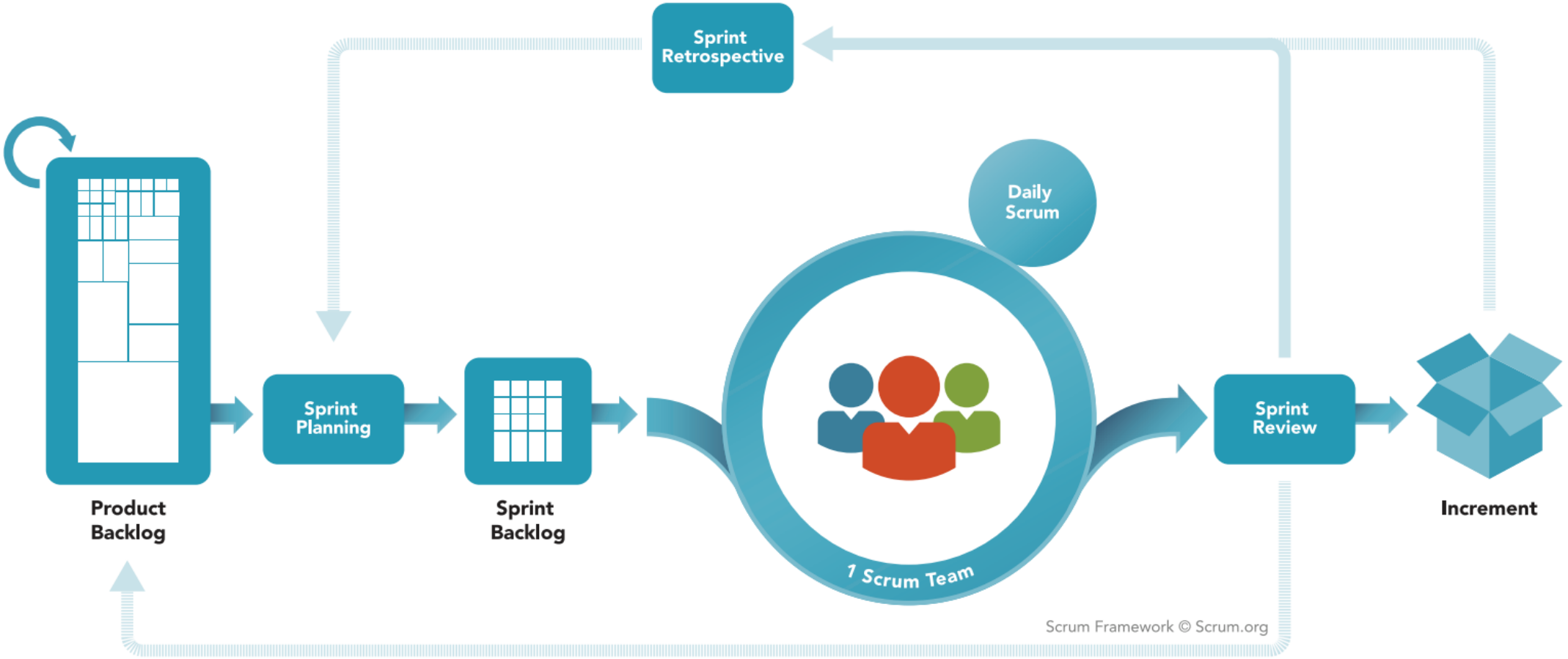
Comparison: Project Success Rates by Size



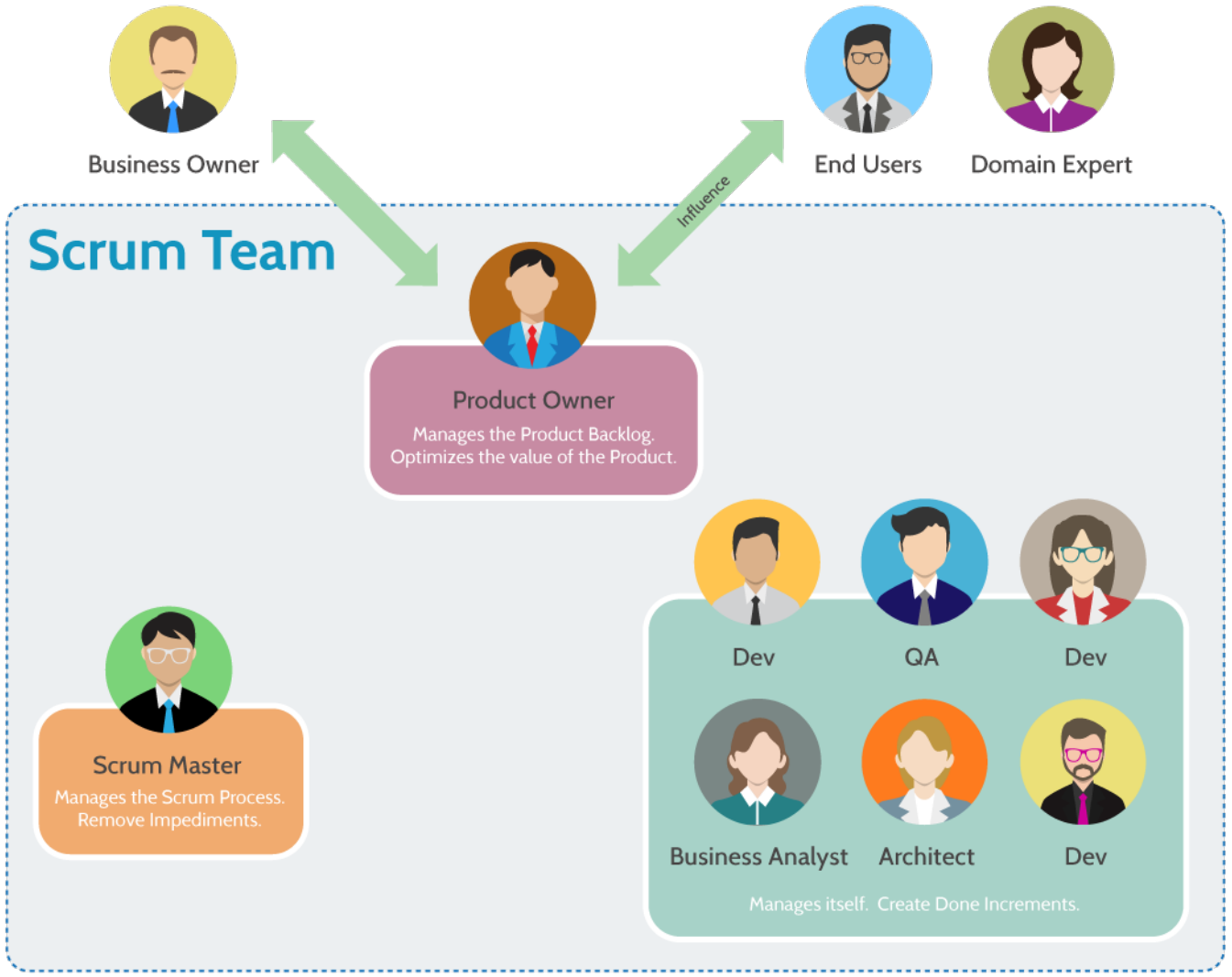
Scrum



Scrum Framework



Scrum Framework: Scrum Team



- Small (up to 10 elements)
- One Product Owner, one Scrum Master and Developers
- No hierarchy
- Self-managed
- Multi-skilled
- Responsible for all activities
- Works in Sprints/Iterations
- Keeps same pace

moOngy.

Scrum Framework: Events

Events AKA Ceremonies, are non-negotiable activities

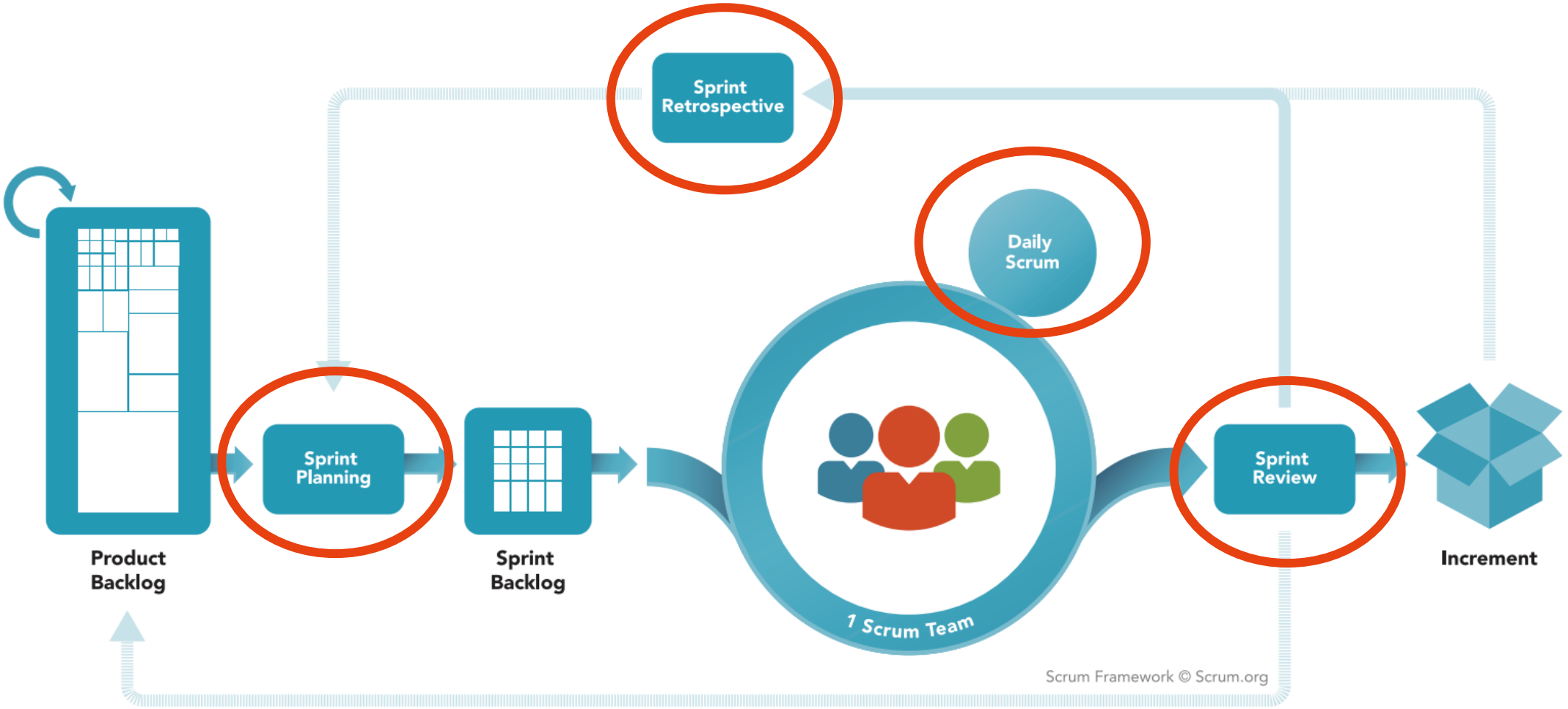
Have a pre-defined duration and must always be executed

They act as collaboration activities inside Scrum Team and help oh self-management

They are defined with clear goals to have all the Scrum Team focused

All of these events occurs inside “master” event called Sprint

Scrum Framework: Events



Scrum Events: Sprint

Sprints are the heartbeat of Scrum, where ideas are turned into value

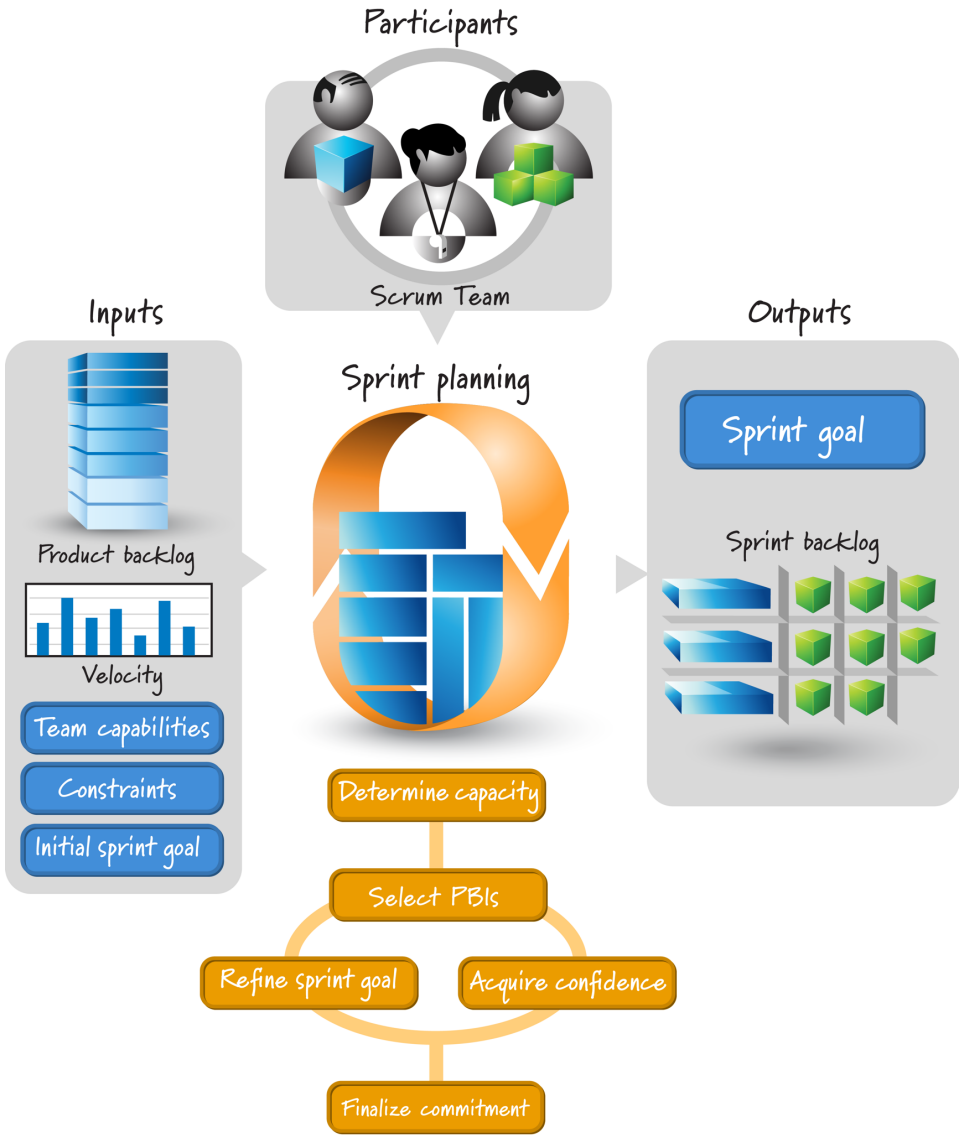
They are fixed length events of 2-4 weeks to create consistency. A new Sprint starts immediately after the conclusion of the previous Sprint.

All the work necessary to achieve the Product Goal, including Sprint Planning, Daily Scrums, Sprint Review, and Sprint Retrospective, happen within Sprints.

During the Sprint:

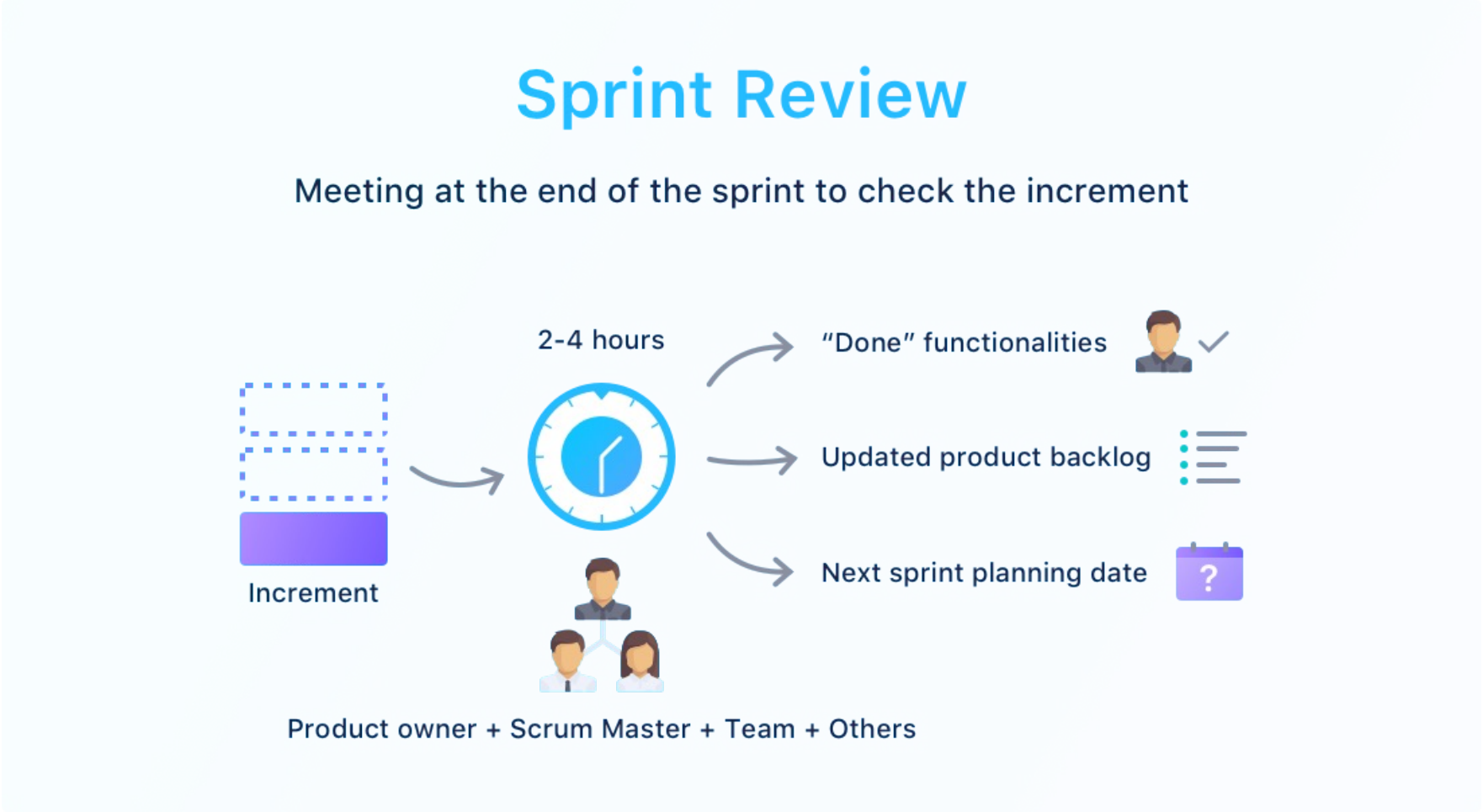
- No changes are made that would endanger the Sprint Goal
- Quality does not decrease
- The Product Backlog is refined as needed
- Scope may be clarified and renegotiated with the Product Owner as more is learned

Scrum Events: Sprint Planning



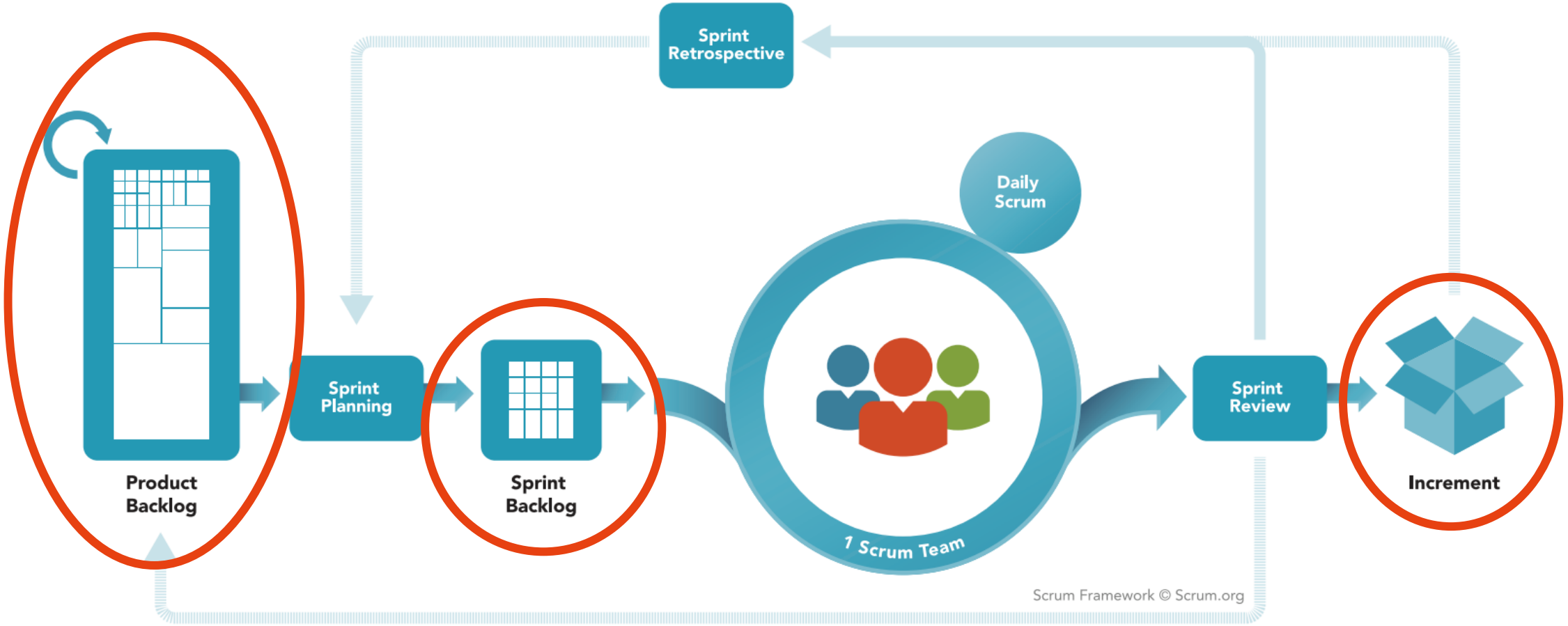
Scrum Events: Daily Scrum

Scrum Events: Sprint Review

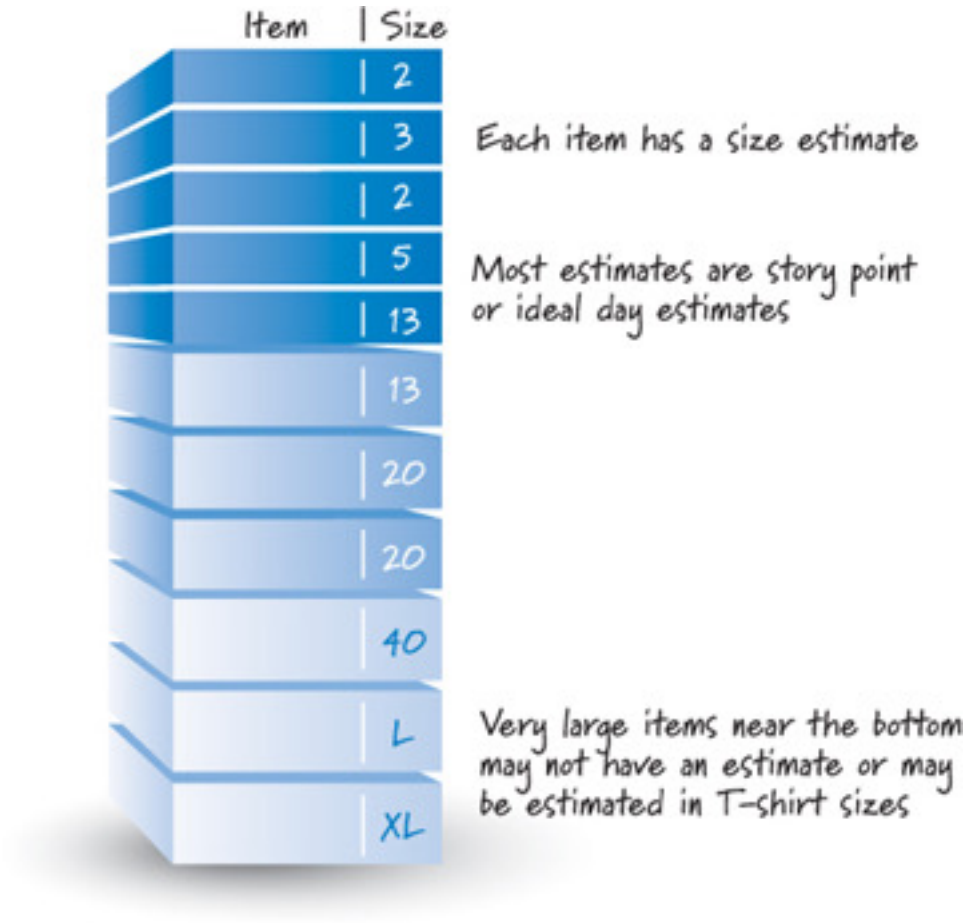


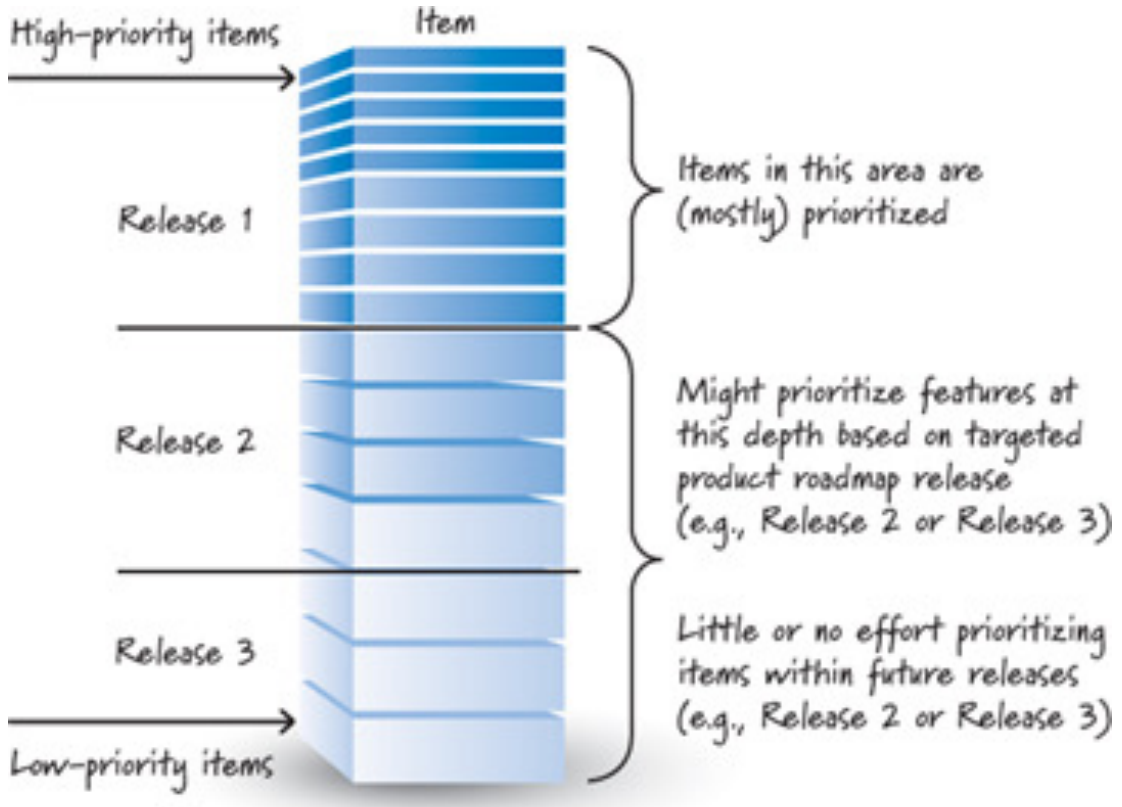
Scrum Events: Sprint Retrospective

Scrum Framework: Artifacts

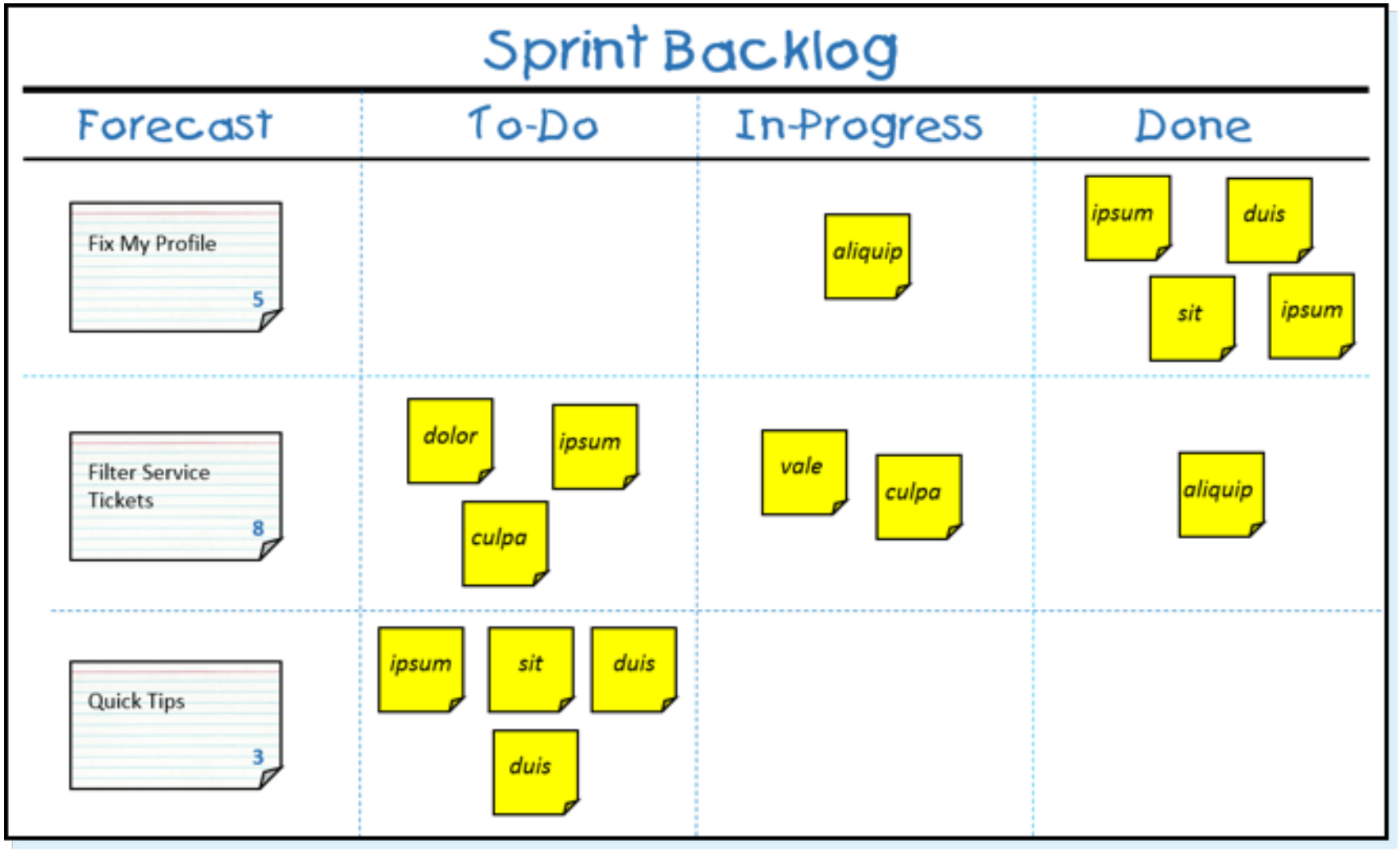


Scrum Artifacts: Product Backlog





Scrum Artifacts: Sprint Backlog



Scrum Artifacts: Increment

Increment is a concrete step toward the Product Goal

Each Increment is additive to all prior Increments, all Increments must work together, and the Increment must be usable

Represent the work done on items defined on Sprint Backlog

Never included unfinished work.

Work cannot be considered part of an Increment unless it meets the Definition of Done

Scrum Artifacts: Definition of Done (DoD)

Criteria defined and agreed by Scrum Team

Each product/project may have different DoD

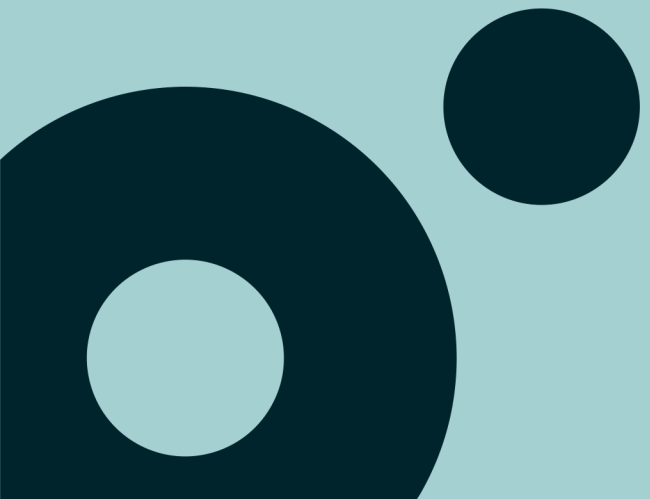
DoD clearly defines when a Sprint Backlog item is considered “Done” by the team

DoD doesn’t mean customer acceptance but team acceptance

Example:

- Code complete
- Unit tests code coverage is X% or higher
- Automated build
- QA Tested

Kanban



Kanban

Kanban is a popular Lean workflow management method for defining, managing, and improving services that deliver knowledge work.

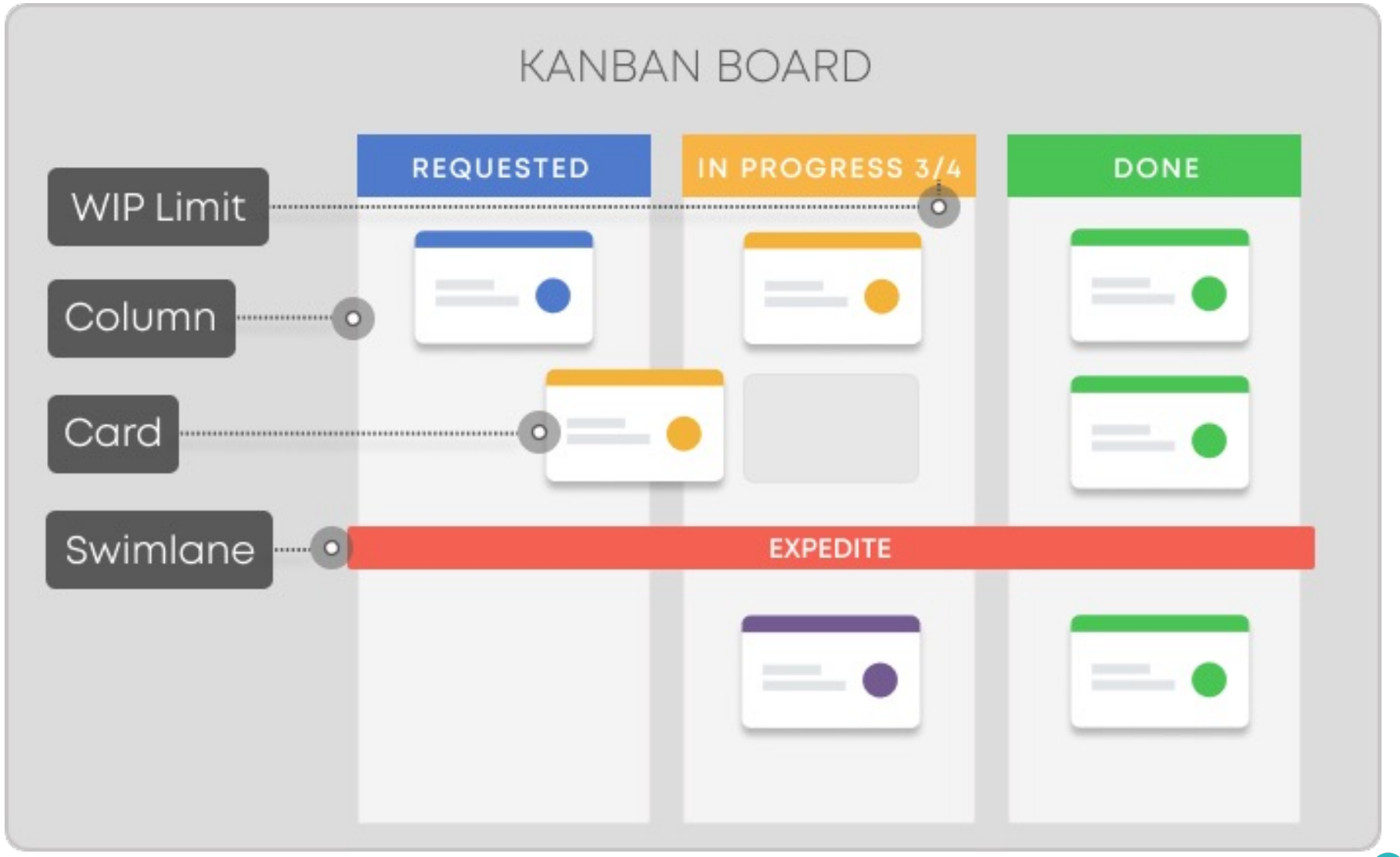
It helps you visualize work, maximize efficiency, and improve continuously.

Work is represented on Kanban boards, allowing you to optimize work delivery across multiple teams and handle even the most complex projects in a single environment

Originating from manufacturing (Toyota), it later became a territory claimed by Agile software development teams.

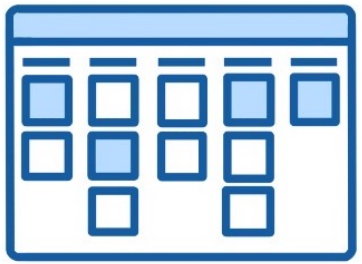
Recently, it started getting recognized by business units across various industries.

Kanban Boards



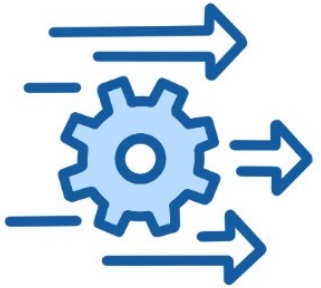
Kanban Boards

Kanban Practices



VISUALIZE

LIMIT WORK
IN PROGRESS



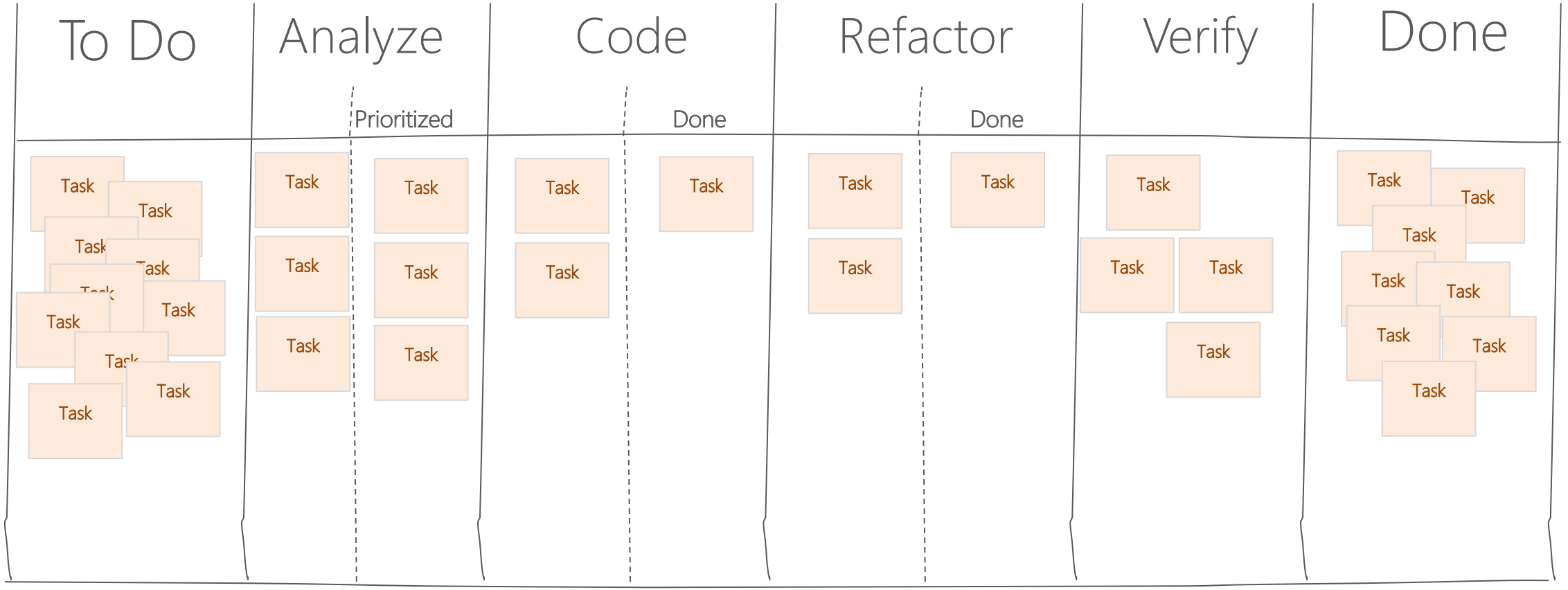
MANAGE
FLOW

MAKE POLICIES
EXPLICIT

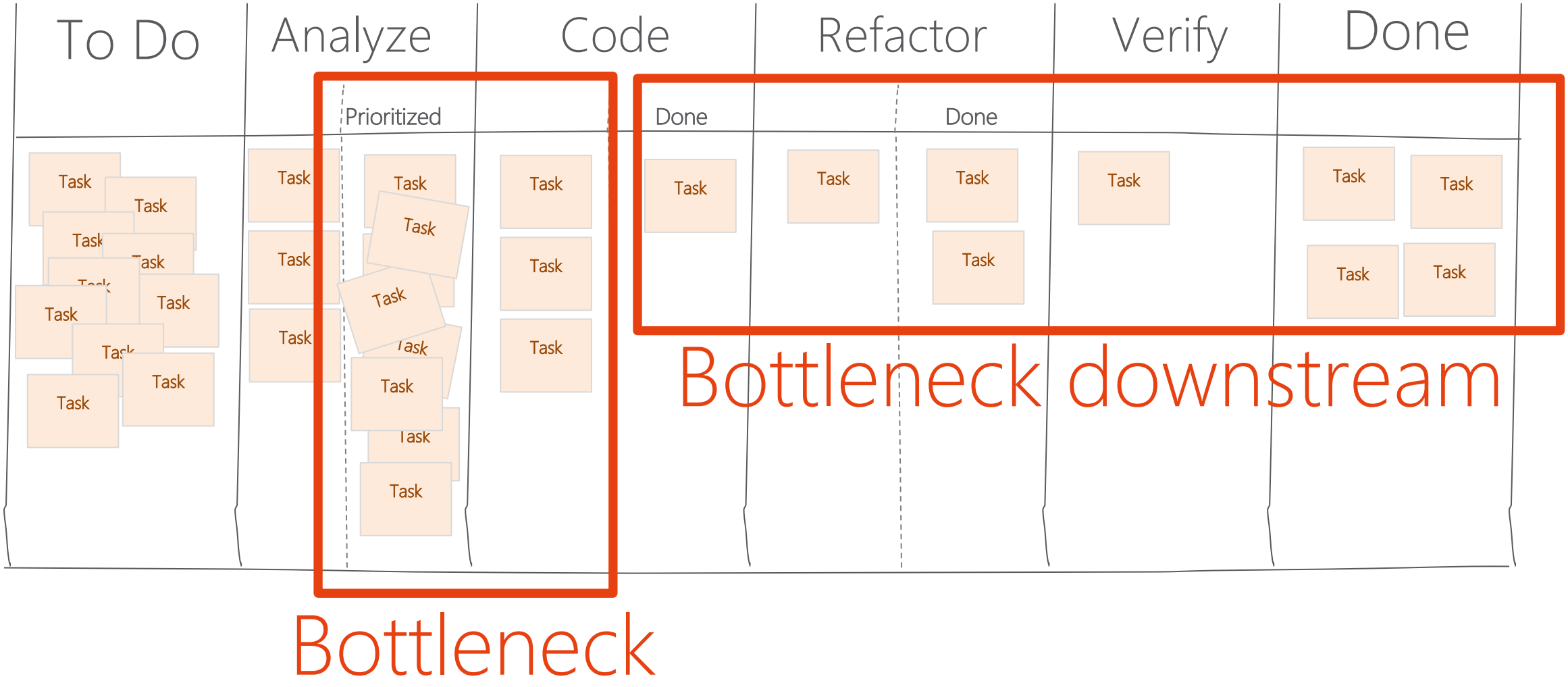
IMPLEMENT
FEEDBACK LOOPS

IMPROVE
COLLABORATIVELY

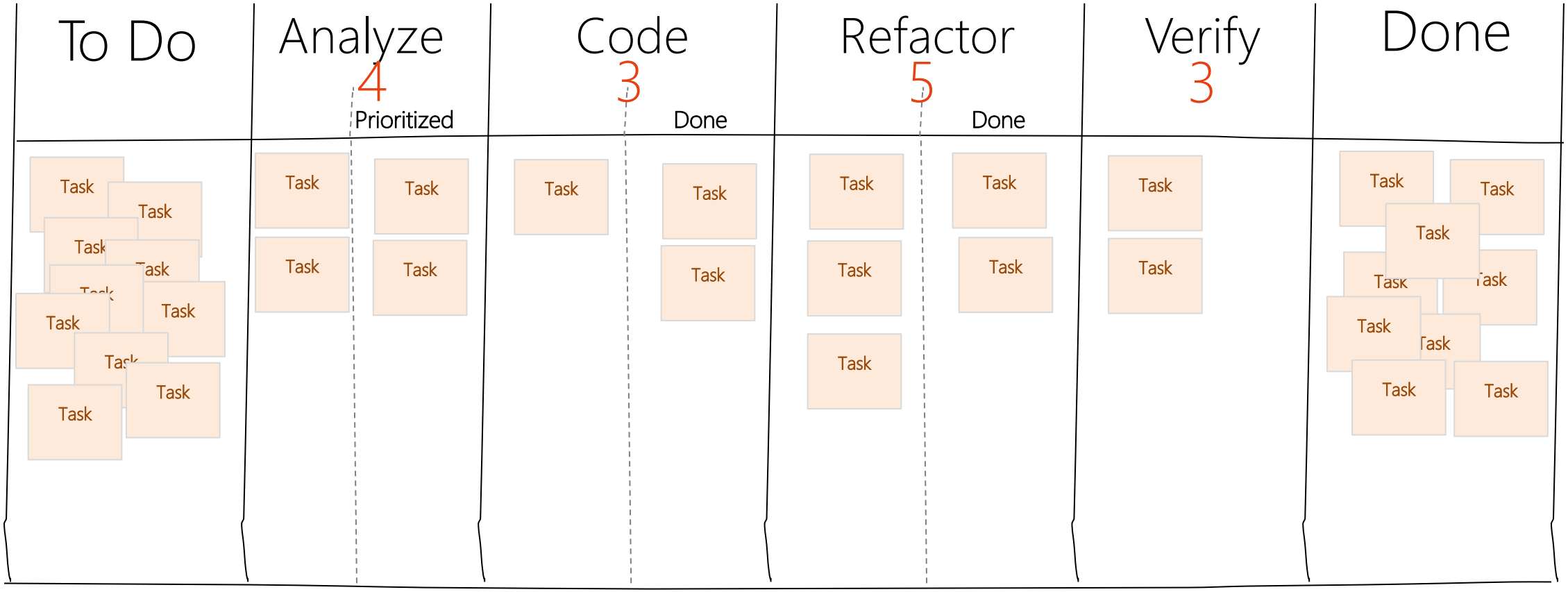
Kanban Principles: Limit work in progress (WIP)



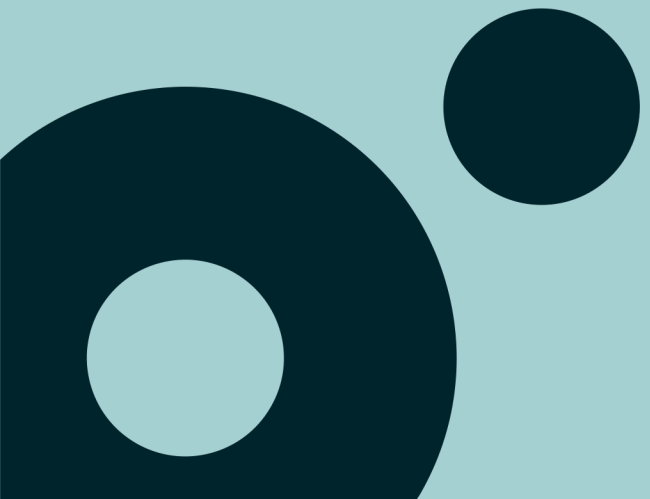
Kanban Principles: Limit work in progress (WIP)



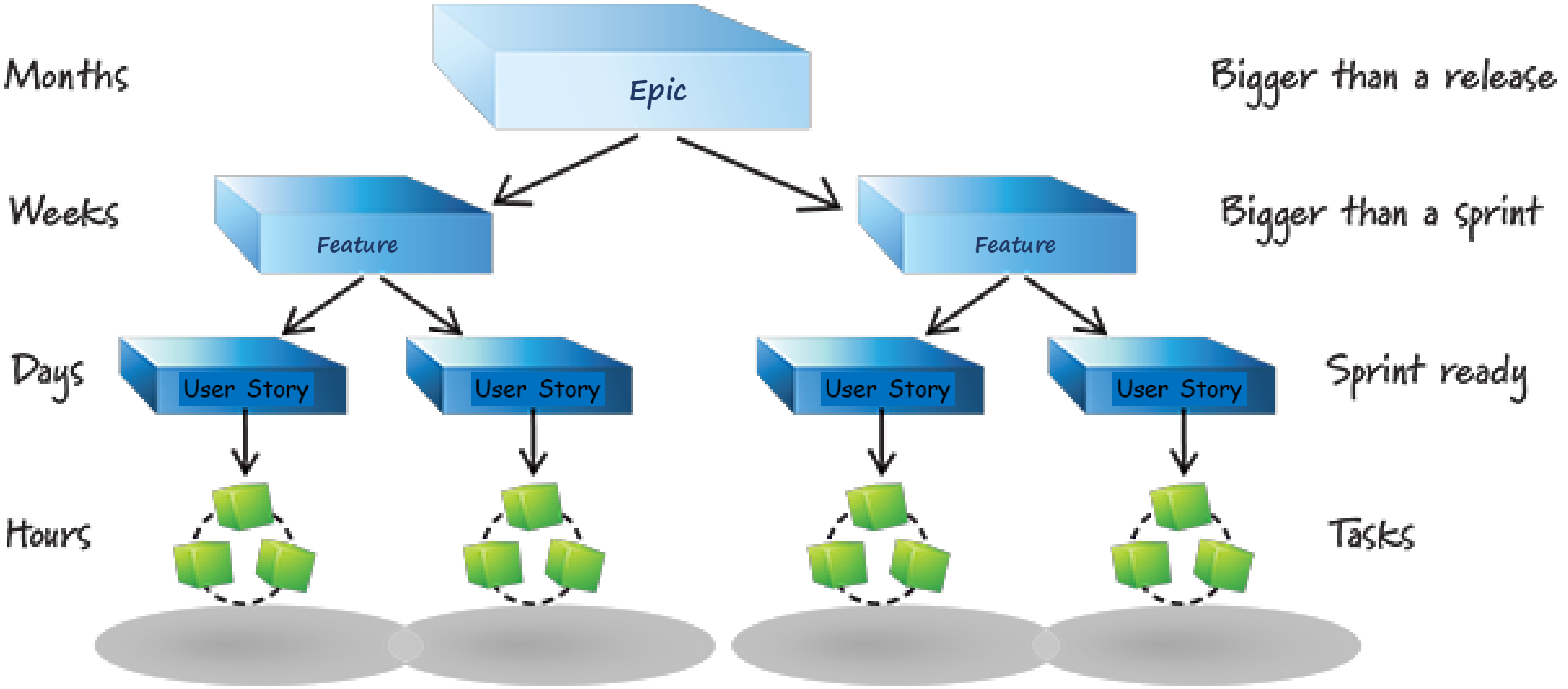
Kanban Principles: Limit work in progress (WIP)



Best Practices



Best Practices: Release Planning



Best Practices: Prioritization

MUST HAVE

The most vital things you can't live without

SHOULD HAVE

Things you consider as important, but not vital

COULD HAVE

Things that are nice to have

WON'T HAVE

Things that provide little to no value you can give up on

Best Practices: Estimation

Use story points to estimate complexity (User Stories)

Different classification number: Sequential, Fibonacci, T-Shirt Sizes

Planning Poker: Everyone gives its classification without knowing the others. When everyone shows up, an agreement must be reach

Use hours to estimate effort (Tasks)

Continuous Best Practices

User Story Definition

INVESTing in well-written user stories
Independent
Negotiable
Valuable
Estimable
Small
Testable

Backlog Grooming

Review backlog to re-prioritize and refinement
Remove what don't make sense anymore

Sprint Lifecycle

Follow Agile Ceremonies
Sprint Planning, Dailies, Sprint Review, Retrospective
Retrospective is the opportunity to improve on planning & tracking process

Demo: GitHub Issues & Projects

moOngy.
Minds on the move

Rua Sousa Martins, nº 10

1050-218 Lisboa | Portugal