

**AGILE AND SCRUM**

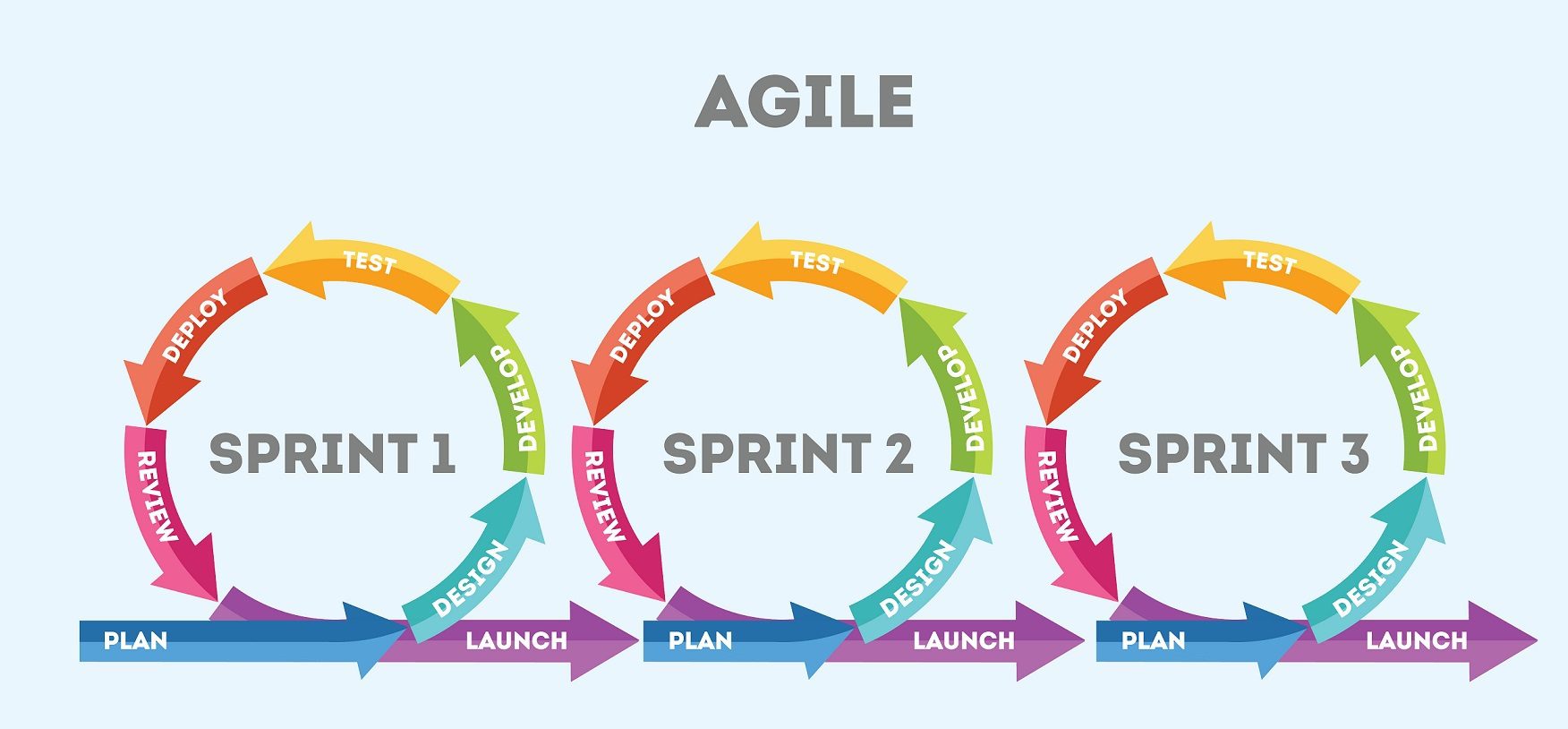
**Agile -> Methodology**

**Scrum -> Agile Framework**

**Introduction:**

In today’s fast-paced software development industry, organizations need approaches that allow them to respond quickly to changes, deliver high-quality products, and maintain customer satisfaction. Traditional project management models like the **Waterfall Model** often fail to meet these requirements because they follow a rigid, linear process. To overcome these limitations, **Agile Methodology** was introduced, providing flexibility, collaboration, and iterative progress.

Among different Agile frameworks, **Scrum** is one of the most widely used. It provides a structured yet flexible approach to delivering projects in small increments, ensuring continuous improvement and stakeholder involvement.



**AGILE:**

**Agile** is a methodology or mindset for software development that emphasizes:

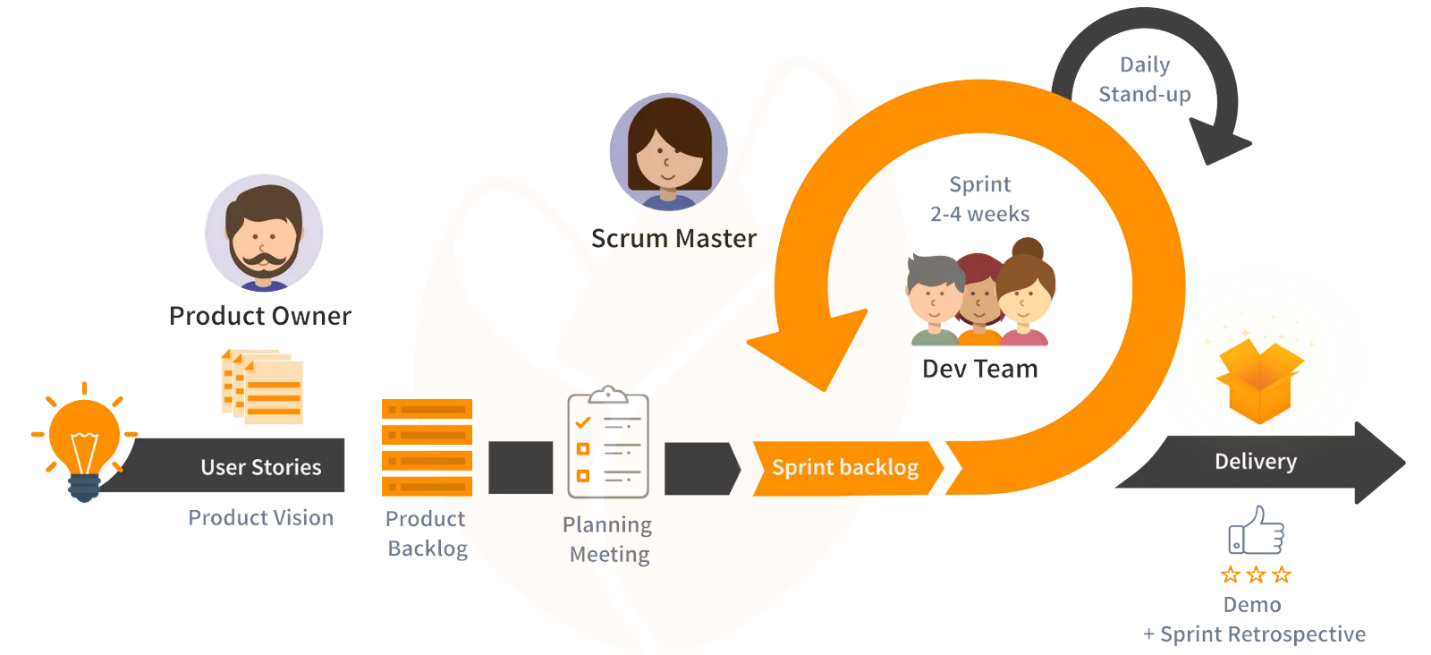
* **Iterative Development**: Work is divided into small, manageable units of time (called iterations or sprints).
* **Collaboration**: Encourages constant communication between developers, testers, product owners, and customers.
* **Adaptability**: Responds quickly to changes in customer requirements.
* **Customer Focus**: Involves the client throughout the process to ensure the final product meets their expectations.

**Key Principles of Agile:**

The **Agile Manifesto** outlines four core values:

1. Individuals and interactions over processes and tools.
2. Working software over comprehensive documentation.
3. Customer collaboration over contract negotiation.
4. Responding to change over following a plan.

Agile methods are not tied to a specific tool or process; rather, they form a set of guiding principles that can be implemented through various frameworks such as **Scrum, Kanban, XP (Extreme Programming), and Lean**.



**SCRUM**

**SCRUM:**

**Scrum** is an Agile framework designed to help teams deliver value in small increments. It focuses on collaboration, accountability, and continuous improvement. Scrum divides work into fixed-length iterations called **Sprints**, usually lasting 2–4 weeks.

**Features of Scrum:**

* A clear structure with defined roles and responsibilities.
* Transparency through regular meetings (Daily Stand-ups, Sprint Reviews, Retrospectives).
* Adaptability through backlog refinement and sprint planning.
* Continuous delivery of potentially shippable product increments.

Scrum is simple to understand but challenging to master because it requires discipline, communication, and commitment from the whole team.

**Key Roles in Scrum:**

Scrum defines **three core roles** within the Scrum Team:

**1. Product Owner**

* Represents the voice of the customer.
* Responsible for maximizing the value of the product.
* Manages the **Product Backlog**, which contains all features, enhancements, and fixes needed for the product.
* Prioritizes backlog items to ensure that the development team works on the most valuable tasks first.
* Communicates regularly with stakeholders to ensure alignment between business needs and development progress.

In short, the Product Owner answers **“What needs to be built?”** and **“Why is it important?”**

**2. Scrum Master**

* Acts as a **servant leader** for the Scrum Team.
* Ensures that the team understands and follows Scrum principles and practices.
* Removes obstacles or impediments that block the team’s progress.
* Facilitates Scrum ceremonies such as Daily Stand-ups, Sprint Planning, Sprint Reviews, and Retrospectives.
* Coaches the Product Owner, Developers, and the organization on Agile mindset and Scrum adoption.

In short, the Scrum Master answers **“How can the team work most effectively?”**

**3. Developers (or Development Team)**

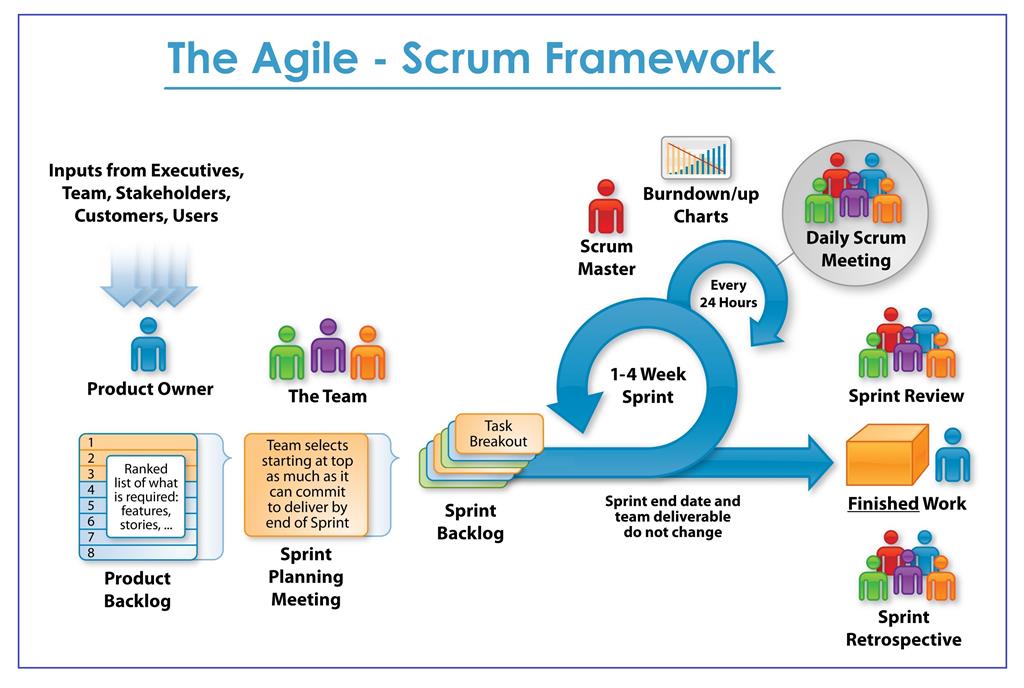
* A cross-functional group responsible for delivering potentially shippable increments at the end of each sprint.
* Self-organizing and collectively accountable for meeting sprint goals.
* Works collaboratively without strict hierarchies.

**Benefits of Agile and Scrum:**

* Faster time-to-market due to shorter iterations.
* Higher product quality through continuous testing and feedback.
* Greater customer satisfaction due to involvement throughout the process.
* Increased team collaboration and ownership.
* Adaptability to changing business environments.

**Conclusion:**

Agile is a philosophy that emphasizes adaptability, collaboration, and customer satisfaction in software development. **Scrum**, as one of its most popular frameworks, provides a practical way to implement Agile principles through well-defined roles, responsibilities, and ceremonies. The **Product Owner** ensures the right product is built, while the **Scrum Master** ensures the team follows best practices and works efficiently. Together, with the development team, they form the foundation of a successful Agile project.



**Step-by-step Scrum flow** with who does what at each stage. Think of it as “from idea → continuous delivery,” repeated every sprint.

1. **Project kickoff (vision and goals)**

* What happens: A product idea/need is identified and a high-level vision is set.
* Who does what:
  + Product Owner (PO): defines product vision, target users, and success metrics.
  + Stakeholders: share needs, constraints, and desired outcomes.
  + Scrum Master (SM): explains Scrum, sets up ways of working.
  + Developers: advise on feasibility and risks.

1. **Form the Scrum Team**

* What happens: A small, cross-functional team (3–9 developers) is created.
* Who does what:
  + PO: available for clarifications and decisions.
  + SM: coaches the team, removes impediments.
  + Developers: commit to delivering increments each sprint.

1. **Create the initial Product Backlog**

* What happens: PO drafts a prioritized list of user stories/features with acceptance criteria.
* Who does what:
  + PO: writes and orders backlog items by value/risk.
  + Developers: estimate effort (e.g., story points) and flag technical dependencies.
  + SM: facilitates, ensures items are clear enough to plan.

1. **Define quality bar and ways of working**

* What happens: Team agrees on Definition of Done (DoD), coding/testing standards, branching strategy, CI/CD, and tooling.
* Who does what:
  + Developers + PO: align on DoD and non-functional requirements.
  + SM: facilitates agreements and makes them visible.

1. **(Optional) Release roadmap/forecast**

* What happens: PO and team sketch a lightweight roadmap (no fixed promises, just forecasts).
* Who does what:
  + PO: outlines milestones and value drops.
  + Developers: provide capacity/velocity assumptions.
  + SM: keeps it empirical and non-prescriptive.

1. **Sprint Planning (start of each sprint, usually 1–4 weeks long)**

* What happens: Team selects the top Product Backlog Items (PBIs) and sets a Sprint Goal; PBIs are broken into tasks.
* Who does what:
  + PO: presents priorities and clarifies intent; negotiates scope.
  + Developers: pull PBIs, plan tasks, and forecast what they can finish.
  + SM: facilitates, timeboxes, keeps focus on Sprint Goal.
* Output: Sprint Goal, Sprint Backlog (selected PBIs + tasks), capacity plan.

1. **Execute the Sprint (build, test, integrate every day)**

* What happens: Team builds a potentially shippable Increment meeting the DoD.
* Who does what:
  + Developers: design, code, test, integrate, document, and keep the board updated.
  + SM: removes impediments, protects the team from distractions.
  + PO: stays available for quick decisions and clarifications.

1. **Daily Scrum (15 minutes, every day)**

* What happens: Developers inspect progress toward the Sprint Goal and adapt the plan for the next 24 hours.
* Who does what:
  + Developers: self-organize, surface risks, adjust tasks.
  + SM: ensures it stays timeboxed and focused.
  + PO: attends if useful, mainly to listen; not mandatory.

1. **Backlog Refinement (lightweight, mid-sprint)**

* What happens: Upcoming PBIs are clarified, split, and estimated so they’re “ready” for a future sprint.
* Who does what:
  + PO: refines and reorders based on feedback/learning.
  + Developers: estimate, propose slicing, call out technical needs.
  + SM: facilitates, keeps it just-enough and timeboxed.

1. **Complete the Increment (meets Definition of Done)**

* What happens: By sprint end, the Increment is fully integrated, tested, and usable.
* Who does what:
  + Developers: ensure all acceptance criteria and DoD checks pass (code, tests, security, docs as agreed).
  + PO: validates the work against intent.
  + SM: helps remove last-mile blockers.

1. **Sprint Review (inspect the product)**

* What happens: Team demos the Increment to stakeholders, gathers feedback, and adapts the Product Backlog.
* Who does what:
  + Developers: demonstrate the working product.
  + PO: explains what was achieved vs. goal; captures feedback and new ideas.
  + Stakeholders: react, ask questions, suggest changes.
  + SM: facilitates collaborative, non-status-meeting vibe.
* Output: Updated Product Backlog and next-steps insights.

1. **Sprint Retrospective (inspect and improve the process)**

* What happens: Team reflects on people, tools, and practices; decides 1–3 concrete improvements for the next sprint.
* Who does what:
  + Developers + PO + SM: share what went well, what to change; agree on actionable experiments.
  + SM: facilitates, ensures psychological safety and follow-through.
* Output: Improvement actions added to the next Sprint plan.

1. **Release (when it makes sense)**

* What happens: If the Increment delivers enough value and meets release criteria, the PO may release it (Scrum allows releasing anytime).
* Who does what:
  + PO: decides release timing based on value and risk.
  + Developers: package/deploy and monitor (often via CI/CD).
  + SM: helps clear organizational blockers to frequent releases.

1. **Repeat**

* What happens: Next Sprint starts immediately with new priorities and the improvement actions from the retrospective.

**Quick role summary**

* Product Owner: owns value and priority (what/why).
* Developers: own delivery quality and predictability (how/how much).
* Scrum Master: owns flow and continuous improvement (enablement/coaching).
* Stakeholders: provide feedback and context, not day-to-day direction.

**Typical timings (for a 2-week sprint)**

* Sprint Planning: ~2–4 hours
* Daily Scrum: 15 minutes per day
* Refinement: up to ~1–2 hours total
* Sprint Review: ~1–2 hours
* Retrospective: ~1–1.5 hours