**CONTENTS**

[**What is Agile Methodology?**](#_What_is_Agile)

[**What is the Agile Manifesto?**](#_What_is_the)

[**Cons of Agile**](#_Cons_of_Agile)

[**Scrum**](#_What_is_Scrum?)

[**Scrum Theory**](#_Scrum_Theory)

[**Scrum Values**](#_Scrum_Values)

[**Scrum Team**](#_Scrum_Team)

[**Scrum Events**](#_Scrum_Events)

[**Scrum Artifacts**](#_Scrum_Artifacts)

[**Stories, Epics, and Initiatives**](#_What_are_stories,)

[**Story point Estimation**](#_Story_point_Estimation)

[**Definition of Done vs Acceptance Criteria**](#_Definition_of_Done)

[**Definition of Ready**](#_Definition_of_Ready)

[**Velocity Chart**](#_Velocity_Chart)

[**Burndown Chart**](#_Burndown_Chart)

[**Kanban**](#_Kanban)

[**Kanban Practices**](#_What_Are_the)

[**Top 6 Benefits of Kanban**](#_Top_6_Benefits)

[**Kanban vs Scrum**](#_Kanban_vs_Scrum)

[**Scrumban**](#_Scrumban)

[**Scrum, Kanban, and Scrumban**](#_Scrum,_Kanban,_and)

## **[Scrum Metrices](#_Scrum_Metrices)**

[**Kanban Metrices**](#_Kanban_Metrices)

## What is Agile Methodology?

Agile methodology is a project management approach that allows successful and efficient execution of the project while emphasizing the improvement of a project and team collaboration. The approach is applicable in software development for flexibility, customer satisfaction, and collaboration. It divides projects into smaller phases and guides teams through cycles of planning, execution, and evaluation. The Agile framework is an iterative methodology. After every sprint, teams reflect and look back to see if there was anything that could be improved so they can adjust their strategy for the next sprint.

Agile methods or Agile processes generally promote a disciplined project management process that encourages frequent inspection and adaptation, a leadership philosophy that encourages teamwork, self-organization and accountability, a set of engineering best practices intended to allow for rapid delivery of high-quality software, and a business approach that aligns development with customer needs and company goals.

## What is the Agile Manifesto?

The Agile Manifesto is a document that focuses on four values and 12 principles for Agile software development.

**4 pillars of Agile** – In the Agile Manifesto, there are four main values of Agile project management

* Individuals over processes and tools: Agile teams value team collaboration and teamwork over working independently.
* Working software over comprehensive documentation: The software that Agile teams develop should work. Additional work, like documentation, is not as important as developing good software.
* Customer collaboration over contract negotiation: Agile teams allow customers to guide where the software should go. Therefore, customer collaboration is more important than the finer details of contract negotiation.
* Responding to change over following a plan: One of the major benefits of Agile project management is that it allows teams to be flexible. This framework allows for teams to quickly shift strategies and workflows without derailing an entire project.

**12 Agile principles** – The four values of Agile are the pillars of Agile methodology. From those values, the team developed 12 principles.

1. Satisfy customers through early, continuous improvement and delivery - When customers receive new updates regularly, they're more likely to see the changes they want within the product. This leads to happier, more satisfied customers—and more recurring revenue.
2. Welcome changing requirements, even late in the project - The Agile framework is all about adaptability. In iterative processes like Agile, being inflexible causes more harm.
3. Deliver value frequently- Delivering value to your customers or stakeholders frequently makes it less likely for them to churn.
4. Break the silos of your projects- The goal is for people to break out of their own individual projects and collaborate more frequently.
5. Build projects around motivated individuals- Agile works best when teams are committed and actively working to achieve a goal.
6. The most effective way to communicate is face-to-face- If you’re working on a distributed team, spend time communicating in ways that involve face-to-face communication.
7. Working software is the primary measure of progress- The most important thing that teams should strive is the product. The goal here is to prioritize functional software over everything else.
8. Maintain a sustainable working pac-. Some aspects of Agile can be fast paced, but it shouldn't be so fast that team members burn out. The goal is to maintain sustainability throughout the project.
9. Continuous excellence enhances agility- If the team develops excellent code in one sprint, they can continue it to the next. Continually creating great work allows teams to move faster in the future.
10. Simplicity is essential- Sometimes the simplest solution is the best solution. Agile aims to not overcomplicate things and find simple answers to complex problems.
11. Self-organizing teams generate the most value- Proactive teams become valuable assets to the company as they strive to deliver value.
12. Regularly reflect and adjust your way of work to boost effectiveness- Retrospective meetings are a common Agile practice. It's a dedicated time for teams to look back and reflect on their performance and adapt their behaviours for the future.

## Cons of Agile

* Poor resource planning - Because Agile is based on the idea that teams won’t know what their end result (or even a few cycles of delivery down the line) will look like from day one, it’s challenging to predict efforts like cost, time and resources required at the beginning of a project.
* Limited documentation - In Agile, documentation happens throughout a project, and often “just in time” for building the output, not at the beginning. As a result, it becomes less detailed and often falls to the back burner.
* Fragmented output -Incremental delivery may help bring products to market faster, due to which the complete output often becomes very fragmented rather than one cohesive unit.
* No finite end - The fact that Agile requires minimal planning at the beginning makes it easy to get sidetracked delivering new, unexpected functionality. Additionally, it means that projects have no finite end, as there is never a clear vision of what the “final product” looks like.
* Difficult measurement - Since Agile delivers in increments, tracking progress requires you to look across cycles. And the “see-as-you-go” nature means you can’t set many KPIs at the start of the project. That long game makes measuring progress difficult.

Mitigating the disadvantages of Agile methodology requires taking more of a Lean approach by emphasizing the importance of:

* Delivering value through a quality end product rather than simply delivering a working product.
* Managing a clear process for delivering that product, not an uncertain route determined along the way.

## What is Scrum?

Scrum is an agile project management framework that helps teams’ structure and manage their work through a set of values, principles, and practices. Scrum encourages teams to learn through experiences, self-organize while working on a problem, and reflect on their wins and losses to continuously improve.

Scrum is a lightweight framework that helps people, teams and organizations generate value through adaptive solutions for complex problems. Scrum is simple. Try it as is and determine if its philosophy, theory, and structure help to achieve goals and create value.

## Scrum Theory

Scrum is founded on empiricism and lean thinking. Empiricism asserts that knowledge comes from experience and making decisions based on what is observed. Lean thinking reduces waste and focuses on the essentials.

Scrum employs an iterative, incremental approach to optimize predictability and to control risk.

* Transparency - The emergent process and work must be visible to those performing the work (development team) as well as those receiving the work (client, PO). Transparency enables inspection. Inspection without transparency is misleading and wasteful.
* Inspection - The Scrum artifacts and the progress toward agreed goals must be inspected frequently and diligently to detect potentially undesirable variances or problems. Inspection enables adaptation. Inspection without adaptation is considered pointless.
* Adaptation - If any aspects of a process deviate outside acceptable limits or if the resulting product is unacceptable, the process being applied, or the materials being produced must be adjusted. The adjustment must be made as soon as possible to minimize further deviation. Adaptation becomes more difficult when the people involved are not empowered or self-managing.

## Scrum Values

Successful use of Scrum depends on people becoming more proficient in living five values:

Commitment, Focus, Openness, Respect, and Courage

## Scrum Team

The fundamental unit of Scrum is a small team of people, a Scrum Team. The Scrum Team consists of one Scrum Master, one Product Owner, and Developers. Within a Scrum Team, there are no sub-teams or hierarchies. It is a cohesive unit of professionals focused on one objective at a time, the Product Goal. They are self-managing & they internally decide who does what, when, and how. The Scrum Team is small enough to remain nimble and large enough to complete significant work within a Sprint, typically 10 or fewer people. The Scrum Team is responsible stakeholder collaboration, verification, maintenance, operation, experimentation, research and development, and anything else that might be required.

* Developers - Developers are the people in the Scrum Team that are committed to creating any aspect of a usable Increment each Sprint. Developers are always accountable for
  + - Creating a plan for the Sprint, the Sprint Backlog.
    - Instilling quality by adhering to a Definition of Done.
    - Adapting their plan each day toward the Sprint Goal.
    - Holding each other accountable as professionals.
* Product Owner - The Product Owner is accountable for maximizing the value of the product. He is also accountable for effective Product Backlog management. The Product Owner is one person, not a committee. Product Backlog can be change by convincing the Product Owner.
  + - Developing and explicitly communicating the Product Goal.
    - Creating and clearly communicating Product Backlog items.
    - Ordering Product Backlog items.
    - Ensuring that the Product Backlog is transparent, visible, and understood.
  + Scrum Master - The Scrum Master is accountable for establishing Scrum. They do this by practice, both within the Scrum Team and the organization. He/she is accountable for the Scrum Team’s effectiveness. Scrum Masters are true leaders who serve the Scrum Team and the larger organization.

The Scrum Master serves the Scrum Team in several ways, including:

* + - Coaching the team members in self-management and cross-functionality.
    - Helping the Scrum Team focus on creating high-value Increments that meet the Definition of Done.
    - Causing the removal of impediments to the Scrum Team’s progress.
    - Ensuring that all Scrum events take place and are positive, productive, and kept within the timebox.

The Scrum Master serves the Product Owner in several ways, including:

* + - Helping find techniques for effective Product Goal definition and Product Backlog management.
    - Helping the Scrum Team understand the need for clear and concise Product Backlog items.
    - Helping establish empirical product planning for a complex environment.
    - Facilitating stakeholder collaboration as requested or needed.

The Scrum Master serves the organization in several ways, including:

* + - Leading, training, and coaching the organization in its Scrum adoption.
    - Planning and advising Scrum implementations within the organization.
    - Helping employees and stakeholders understand and enact an empirical approach for complex work.
    - Removing barriers between stakeholders and Scrum Teams.

## Scrum Events

The Sprint is a container for all other events. Each event in Scrum is a formal opportunity to inspect and adapt Scrum artifacts. These events are specifically designed to enable the transparency required. Failure to operate any events as prescribed results in lost opportunities to inspect and adapt.

The Sprint - Sprints are the heartbeat of Scrum, where ideas are turned into value. They are fixed length events of one month or less to create consistency. A new Sprint starts immediately after the conclusion of the previous Sprint.

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.

Sprints enable predictability by ensuring inspection and adaptation of progress toward a Product Goal. Shorter Sprints can be employed to generate more learning cycles and limit risk of cost and effort to a smaller time frame. Various practices exist to forecast progress, like burn-downs, burn-ups, or cumulative flows. A Sprint could be cancelled if the Sprint Goal becomes obsolete. Only the Product Owner has the authority to cancel the Sprint.

**Sprint Planning** - Sprint Planning initiates the Sprint by laying out the work to be performed for the Sprint. This resulting plan is created by the collaborative work of the entire Scrum Team. The Product Owner ensures that attendees are prepared to discuss the Product Backlog items and how they map to the Product Goal. The Scrum Team may also invite other people to attend Sprint Planning to provide advice.

Sprint Planning addresses the following topics:

* Why is this Sprint valuable - The Product Owner proposes how the product could increase its value. The whole Scrum Team then collaborates to define a Sprint Goal. The Sprint Goal must be finalized prior to the end of Sprint Planning.
* What can be Done this Sprint - Through discussion with the Product Owner, the Developers select items from the Product Backlog to include in the current Sprint. The Scrum Team may refine these items during this process. Developers know about their past performance, their upcoming capacity, and their Definition of Done, the more confident they will be in their Sprint forecasts.
* How will the chosen work get done - Developers plan the work necessary to create an Increment that meets the Definition of Done. This is often done by decomposing Product Backlog items into smaller work items of one day or less. Sprint Planning is timeboxed to a maximum of eight hours for a one-month Sprint.

**Daily Scrum** - The purpose of the Daily Scrum is to inspect progress toward the Sprint Goal and adapt the Sprint Backlog as necessary, adjusting the upcoming planned work. The Daily Scrum is a 15-minute event for the Developers of the Scrum Team. It is held at the same time and place every working day of the Sprint. If the Product Owner or Scrum Master are actively working on items in the Sprint Backlog, they participate as Developers. The Developers can select whatever structure and techniques they want, as long as their Daily Scrum focuses on progress toward the Sprint Goal and produces an actionable plan for the next day of work. Daily Scrums improve communications, identify impediments, promote quick decision-making, and consequently eliminate the need for other meetings. The Daily Scrum is not the only time Developers are allowed to adjust their plan. They often meet throughout the day for more detailed discussions about adapting or re-planning the rest of the Sprint’s work.

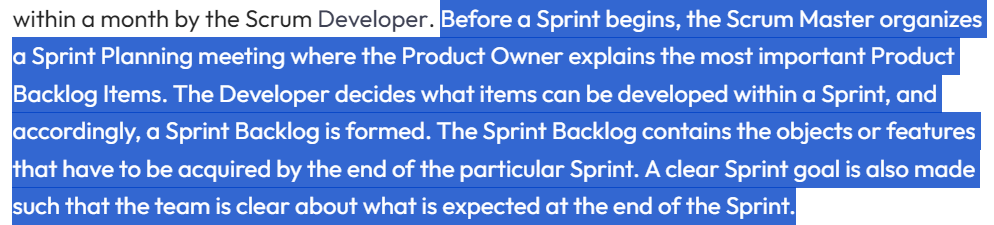
**Sprint Review** - The purpose of the Sprint Review is to inspect the outcome of the Sprint and determine future adaptations. The Scrum Team presents the results of their work to key stakeholders and progress toward the Product Goal. During the event, the Scrum Team and stakeholders review what was accomplished in the Sprint and what has changed in their environment. The Product Backlog may also be adjusted to meet new opportunities. The Sprint Review is a working session, and the Scrum Team should avoid limiting it to a presentation.

**Sprint Retrospective** - The purpose of the Sprint Retrospective is to plan ways to increase quality and effectiveness. The Scrum Team inspects how the last Sprint went with regards to individuals, interactions, processes, tools, and their Definition of Done. The Scrum Team discusses what went well during the Sprint, what problems it encountered, and how those problems were (or were not) solved. The Scrum Team identifies the most helpful changes to improve its effectiveness. The most impactful improvements are addressed as soon as possible. They may even be added to the Sprint Backlog for the next Sprint. The Sprint Retrospective concludes the Sprint. It is timeboxed to a maximum of three hours for a one-month Sprint.

## Scrum Artifacts

**Product Backlog** - A product backlog is a prioritized list of work for the development team that is derived from the roadmap and its requirements. The most important items are shown at the top of the product backlog so the team knows what to deliver first. The development team doesn't work through the backlog at the product owner's pace and the product owner isn't pushing work to the development team. Instead, the development team pulls work from the product backlog as there is capacity for it.

**Release backlog:** Features that need to be implemented for a particular release.

**Sprint backlog:** User stories that need to be completed during a specific period of time. The sprint backlog comes from the product backlog, but it contains only the product backlog items that can be completed during each agile sprint. 

## What are stories, epics, and initiatives?

* A user story is the smallest unit of work in an agile framework. It’s an end goal, not a feature, expressed from the software user’s perspective. A user story is an informal, general explanation of a software feature written from the perspective of the end user or customer. The purpose of a user story is to articulate how a piece of work will deliver a particular value back to the customer.
* Epics are a helpful way to organize your work and to create a hierarchy. The idea is to break work down into shippable pieces so that large projects can actually get done and you can continue to ship value to your customers on a regular basis. Epics help teams break their work down, while continuing to work towards a bigger goal.
* Initiatives are collections of epics that drive toward a common goal.

## Story point Estimation

When the engineering team begins its estimation process, questions usually arise about requirements and user stories. And that's good: those questions help the entire team understand the work more fully. For product owners specifically, breaking down work items into granular pieces and estimates via story points helps them prioritize all (and potentially hidden!) areas of work. And once they have estimates from the dev team, it's not uncommon for a product owner to reorder items on the backlog.

## Definition of Done vs Acceptance Criteria

Definition of Done (DoD) is a list of requirements that a user story must adhere to for the team to call it complete. While the Acceptance Criteria of a User Story consist of set of Test Scenarios that are to be met to confirm that the software is working as expected. The DoD is common for all the User Stories whereas the Acceptance Criteria is applicable to specific User Story. Acceptance Criteria of each User Story will be different based on the requirements of that User Story.

Both DoD and Acceptance Criteria must be met in order to complete the User Story.

The definition of Done is structured as a list of items, each one used to validate a Story, which exists to ensure that the Development Team agree about the quality of work they’re attempting to produce. It serves as a checklist that is used to check each Product Backlog Item.

Example – Definition of Done –

* Code peer reviewed?
* Code completed?
* Code reviewed?
* Code checked-in?
* Unit tests passed?
* Functional tests passed?
* Acceptance tests completed?
* Product Owner reviewed and accepted?

Acceptance Criteria - The acceptance criteria gives guidance about the details of said functionality and how the customer will accept them. Some of the Acceptance Criteria will be discovered in Ongoing Backlog Refinement events before the Sprint starts, and others will be discovered right after Sprint Planning.

The goals of Acceptance Criteria –

* Clarify what the team should build before they start work
* Ensure everyone has a common understanding of the problem
* Help the team members know when the Story is complete
* Help verify the Story via automated tests.

Example – Acceptance Criteria

* A user cannot submit a form without completing all the mandatory fields
* Information from the form is stored in the registrations database
* Payment can be made via credit card
* An acknowledgment email is sent to the user after submitting the form

## Definition of Ready

The Definition of Ready (DoR) is a list of criteria that must be met in order for a PBI to be considered "ready" for development. It serves as a checklist to ensure that the team has all the necessary information, requirements, and dependencies clarified and ready before work begins. The DoR helps prevent disruptions and delays during the sprint and promotes a smoother flow of work.

## Velocity Chart

<https://www.atlassian.com/agile/project-management/metrics>

Velocity is how a scrum team measures the amount of work they can complete in a typical sprint. Velocity is measured historically, from one sprint to the next. By tracking the number of story points the team can complete according to their own definition of done, they can build up a reliable and predictable sense of how long it will take them to complete new stories based on their relative point value.

Keeping track of the velocity is the responsibility of the scrum master. At the end of each sprint demo, the scrum master should calculate the number of points that were estimated for the stories.

Velocity in Scrum is a critical metric that helps teams estimate the amount of work they can complete in a given time frame, typically a sprint. It is calculated by adding up the estimates of the work (usually in story points) completed in the last sprint. This provides a useful benchmark for planning future sprints.

Improving a team’s velocity involves enhancing their efficiency and effectiveness. This could be achieved through regular retrospectives to identify and address issues, providing necessary training and resources, improving communication and collaboration, and fostering a supportive work environment.

## Burndown Chart

A burndown chart shows the team’s progress toward completing all of the points they agreed to complete within a single sprint. This chart starts with the total number of points the team has taken on for the sprint, and tracks on a day-to-day basis how many of those points have been completed and are ready for the sprint demo.

The burndown chart is usually maintained by the scrum master, and may be updated on a daily basis, perhaps after the daily stand up.

A typical burndown chart starts with a straight diagonal line from the top left to the bottom right, showing an “ideal” burndown rate for the sprint.

Burndown charts are visual representations of work left to do versus time. They provide a quick overview of the project’s progress and the amount of work remaining. This helps teams and stakeholders understand if the project is on track to meet its deadlines.

If the burndown chart indicates the project isn’t on track, it’s crucial to identify and address the issues causing the delay. This could involve re-evaluating task estimates, removing blockers, increasing resources, or adjusting the scope of work.

## Kanban

**Kanban** is one of the most popular Lean workflow management methods for defining, managing, and improving services that deliver knowledge work. It helps you visualize work, maximize efficiency, and improve continuously. The approach represents a pull system. This means that production is based on customer demand rather than the standard push practice of producing goods and pushing them to the market.

The 6 Core Kanban Principles

* Start with what you do now: Kanban is about continuous improvement, but it starts with an understanding of the current processes and workflows.
* Agree to pursue incremental, evolutionary change: Rather than attempting a large-scale transformation all at once, Kanban advocates for small, incremental changes.
* Encourage acts of leadership at all levels: Kanban is not just for managers or team leads but for everyone involved in the work.
* Focus on customer needs and expectations: Kanban promotes understanding the needs and expectations of your customers to elevate the quality of the provided services.
* Manage the work, not the workers: Kanban respects the existing roles and responsibilities of team members and empowers people’s abilities to self-organize around the work.
* Regularly review the network of services: Kanban encourages collaboration and encourages team members to share their observations, ideas, and feedback for improving the work.

## What Are the Kanban Practices?

For a successful Kanban implementation, the method relies on six essential practices:

* Visualizing the workflow: Creating a visual representation of the workflow helps to identify bottlenecks.
* Limiting work in progress: Limiting the amount of work in progress helps to prevent multitasking and improve focus on completing one task at a time.
* Managing flow: Kanban aims to help in optimizing flow which can be achieved by monitoring flow metrics.
* Making process policies explicit: Defining and communicating process policies clearly helps to ensure that everyone understands how work is supposed to be done.
* Implementing feedback loops: Kanban emphasizes the importance of getting feedback from customers, stakeholders, and team members to identify areas for improvement.
* Improving collaboratively: Kanban is a continuous improvement process that encourages collaboration and experimentation to identify and solve problems.

## Top 6 Benefits of Kanban

* Increased visibility of the flow
* Improved delivery speed
* Alignment between goals and execution
* Improved predictability
* Improved dependencies management
* Increased customer satisfaction

## Kanban vs Scrum

|  |  |  |
| --- | --- | --- |
|  | **Kanban** | **Scrum** |
| Nature | Kanban is an adaptive method | Scrum is a prescriptive framework |
| Principles | 1. Start with what you do now  2. Agree to pursue evolutionary change  3. Encourage acts of leadership at all levels  4. Focus on customer’s needs  5. Manage the work  6. Regularly review the network of services | 1. Empiricism  2. Transparency  3. Inspection  4. Adaptation |
| Cadences | - Team-level cadences  - Service-oriented cadences | - Sprint with a fixed length  - Sprint planning  - Daily Scrum  - Sprint Review  - Sprint Retrospective |
| Roles | - Service Delivery Manager\*  - Service Request Manager\*  (\*no pre-defined roles are required) | - Product Owner  - Scrum Master  - Development Team |
| Metrics | - Cycle Time  - Throughput  - Work In Progress | - Velocity  - Planned Capacity |

| **Methodology** | **Kanban** | **Scrum** |
| --- | --- | --- |
| **Roles** | No defined roles | Scrum master, product owner, and development team |
| **Delivery cycle** | Continuous | Sprint cycle lasts one to four weeks |
| **Change policy** | Can be incorporated any time | Generally, not made during sprint |
| **Artifacts** | Kanban board | Product backlog, sprint backlog, product increments |
| **Tools** | Jira Software, Kanbanize, SwiftKanban, Trello, Asana | Jira Software, Axosoft, VivifyScrum, Targetprocess |
| **Key concepts or pillars** | Effective, efficient, predictable | Transparency, adaptation, inspection |

## Scrumban

<https://www.atlassian.com/agile/project-management/scrumban>

The Scrumban methodology combines the best features of Scrum and Kanban into a hybrid project management framework. It uses Scrum's stable structure of sprints, standups, and retrospectives. Then it adds Kanban's visual workflow and work-in-progress limitations. The result is a truly flexible method for managing projects of any size.

Scrumban initially started as a way for teams to easily transition from Scrum to Kanban (or vice-versa) but has evolved into a mature system that allows teams to tackle complex, ongoing projects. It's flexible due to its hybrid approach and because it gives teams a wide range of [Agile tools](https://www.atlassian.com/en/software/jira/agile#scrum).

But is Scrumban an Agile approach? Yes. It combines Scrum and Kanban—both Agile methodologies—and draws elements from their [Agile workflows](https://www.atlassian.com/en/agile/project-management/workflow) to create a hybrid process.

How Scrum contributes to Scrumban

Scrum contributes three vital ingredients to the Scrumban framework, including:

* **Sprints**: Teams complete all work in a Scrumban project within a defined period called a sprint. Sprints typically last two weeks, though teams may opt for shorter (or longer) periods. Once the team agrees on what tasks to work on within the sprint, they can't receive new tasks until the sprint ends.
* **Daily standups**: To clarify who does what, Scrumban teams meet for a daily standup—a meeting of no more than 10 minutes. There, participants briefly answer three questions:
  + What tasks did I complete?
  + What tasks am I working on?
  + What are my roadblocks?
* **Retrospectives**: At the end of every sprint, the team meets to analyze their performance. What processes went well and what should they repeat? What didn't work and what should they stop? The team documents the wisdom they glean from each retrospective and uses it to inform future sprints.

A quick clarification: Scrum is not the same as Agile, so don't mix them up. While Scrum is a framework for completing tasks, Agile is the set of principles or mindset you need to allow the Scrum process to succeed.

In short, you can't use Agile without a framework (such as Scrum, Kanban, or Scrumban). And you can't use Scrum without [cultivating an Agile mindset](https://www.atlassian.com/en/agile/advantage/agile-mindset).

How Kanban contributes to Scrumban

Kanban has three main elements that bring value to the Scrumban methodology:

* **The board**: A [Kanban board](https://www.atlassian.com/en/agile/kanban/boards) typically has columns denoting each phase of a project: a "to do" column, an "in progress" column, and a "done" column.
* **The cards**: The board holds the project's tasks, or cards (since historically teams used index cards or sticky notes on whiteboards). As each member begins work on a card, they move it from "to do" to "in progress" and, once finished, into the "done" column.
* **The work-in-progress limits**: For a team to work efficiently, they must know what they can realistically handle within a defined work period. How many cards can a team member work on within a day? By defining this limit, the team prevents overwork (and burnout) while showing stakeholders exactly what they're accomplishing.

## Scrum, Kanban, and Scrumban

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Scrum** | **Kanban** | **Scrumban** |
| Methodology | Fixed length sprints  Fixed roles  Consistent delivery | Limit work in progress  Track tasks visually  Continuous flow of work | Fixed length sprints  Limit work in progress  Track tasks visually  Continuous flow of work |
| Roles | Product Owner  Scrum Master  Development team | None | None |
| Artifacts | Product backlog  Sprint backlog  Increment finished | Kanban board  Kanban cards | Scrumban board  Scrumban cards |
| Events | Sprint planning  Daily standup  Sprint review  Sprint retrospective | Kanban meeting | Sprint planning  Daily standup  Sprint retrospective |
| Process flow | Product backlog  Sprint backlog  In progress  Review  Done | To Do  In Progress  Done | To Do  In Progress  Done |

Benefits of the Scrumban methodology

Why choose the Scrumban methodology over just Scrum or Kanban? The power lies in its combination of the best parts of both methodologies:

* **Increased flexibility**: Because Scrumban delivers incremental work with every sprint, it allows for project changes even in the middle of the process. Meanwhile, there's progress toward project completion.
* **Continuous delivery**: With a board and a continuous flow of work, teams can deliver features as they complete them—no need to wait for a sprint to end.
* **Reduced overloading**: Limiting work in progress to the team's capabilities allows for progress without burnout.
* **Faster issue resolution**: Laying out cards on a board gives Scrumban the transparency teams need for better collaboration while quickly identifying problems and solutions.
* **Ability to tackle large-scale projects**: Because both Scrum and Kanban are about continuous and incremental improvement, Scrumban allows teams to work toward the completion of even the most complex projects.

Limitations of the Scrumban methodology

While there are many advantages of the Scrumban methodology, there are also some limitations to be aware of, including:

* **Ambiguity**: Scrumban is relatively new, so there isn't much documentation on its implementation. As such, there could be some difficulty in finding guidance or best practices.
* **Less control**: Since Scrumban forgoes traditional Scrum roles, team members self-manage their sprints. The absence of a clear leader may lead to confusion over responsibilities.
* **Complexity**: Taking elements from two methodologies may be confusing to team members who are familiar with another system or who have never used an Agile system.

When to use Scrumban

Scrumban is perfect for specific use cases where Scrum or Kanban alone aren't sufficient. Some examples of this include:

Software development projects with evolving requirements

Software projects with scope creep is the perfect opportunity to use the Scrumban methodology. Scrumban allows for this with sprints and continual development. Even as requirements change, teams can continue to complete work incrementally.

Projects with multiple, concurrent initiatives

In larger companies, it's not uncommon to have simultaneous ongoing projects. Sometimes, they even involve the same teams. Because of its flexibility, Scrumban allows for simultaneous initiatives, so even small teams can tackle multiple requirements.

Startups or rapidly changing environments

Startups often have constantly changing environments and projects. Every day brings a new challenge, and there are rarely any resources. Scrumban brings capability into that environment and allows even small teams to thrive with its flexible framework for doing work.

In the end, it's not [Kanban vs. Scrum](https://www.atlassian.com/en/agile/kanban/kanban-vs-scrum). Taking the principles that work best for your team from each methodology makes for a more efficient way to tackle any project.

## Scrum Metrices

Velocity Charts

Burndown Charts

Epic Burndown Charts

## Kanban Metrices

Lead Time

Cycle Time

Work In Progress

Throughput