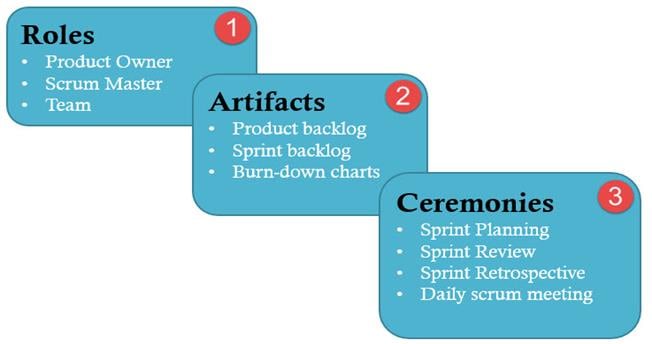
**Scrum in Software Testing** is a methodology for building complex software applications. It provides easy solutions for executing complicated tasks. **Scrum Testing** is a testing done in scrum methodology to verify the software application requirements are met. It involves checking non-functional parameters like security, usability, performance etc. There is no active role of tester in the process so it is usually performed by developers with Unit Test. Sometimes dedicated test teams are needed depending on nature & complexity of project.

Following are Key Features of Scrum-

* Scrum has a short fixed schedule of release cycles with adjustable scope known as sprints to address rapidly changing development needs. Each release could have multiple sprints. Each Scrum Project could have multiple Release Cycles.
* A repeating sequence of meetings, events, and milestones
* A practice of testing and implementing new requirements, known as stories, to make sure some work is released ready after each sprint

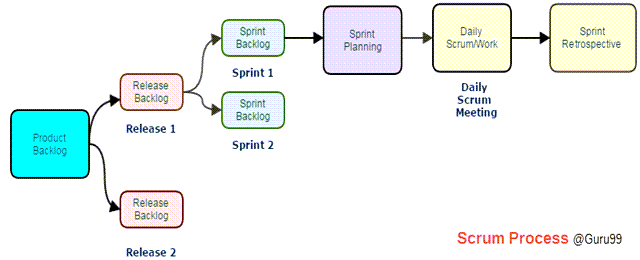


**1. Roles in Scrum**

There are three chief roles in Scrum Testing – Product Owner, Scrum Master and The Development Team. Let’s study them in detail

| **Product Owner** | **Scrum Master** | **The Team** |
| --- | --- | --- |
| He/She defines features of the product. | He/She manages the team and look after the team’s productivity | The team is usually about 5-9 members |
| Product Owner decides the release date and corresponding features | He/She maintains the block list and removes barriers in the development | It includes developers, designer and sometimes testers, etc. |
| They prioritize the features according to the market value and profitability of the product | He/She coordinates with all roles and functions | The team organizes and schedule their work on their own |
| He/She is responsible for the profitability of the product | He/She shields team from external interferences | Has right to do everything within the boundaries of the project to meet the sprint goal |
| He/She can accept or reject work item result | Invites to the daily scrum, sprint review and planning meetings | Actively participate in daily ceremonies |

**2. Scrum Artifacts**

[](https://www.guru99.com/images/2/scrum_testing_2.png)

A scrum process includes

* **User stories:** They are a short explanation of functionalities of the system under test. Example for Insurance Provider is – “Premium can be paid using the online system.”
* **Product Backlog:** It is a collection of user stories captured for a scrum product. **The product owner prepares** and maintains the product backlog. It is prioritized by the product owner, and anyone can add to it with approval from the product owner.
* **Release Backlog:** A release is a time frame in which the number of iterations is completed. **The product owner co-ordinates** with the scrum master to decide which stories should be targeted for a release. Stories in the release backlog are targeted to be completed in a release.
* **Sprints:** It is a set period of time to complete the user stories, decided by the product owner and developer team, usually 2-4 weeks of time.
* **Sprint Backlog:** It’s a set of user stories to be completed in a sprint. During sprint backlog, work is never assigned, and the team signs up for work on their own. It is owned and managed by the team while the estimated work remaining is updated daily. It is the list of task that has to be performed in Sprint
* **Block List:** It is a list of blocks and unmade decisions owned by scrum master and updated daily
* **Burndown chart:** Burn-down chart represents overall progress of the work in progress and work completed throughout the process. It represents in a graph format the stories and features not completed

**3. Ceremonies (Processes) in Scrum**

* **Sprint Planning:** A sprint begins with the team importing stories from the release backlog into the sprint backlog; it is hosted by scrum master. The Testers estimate effort to test the various stories in the Sprint Backlog.
* **Daily Scrum:** It is hosted by scrum master, it last about 15 minutes. During Daily Scrum, the members will discuss the work completed the previous day, the planned work for the next day and issues faced during a sprint. During daily stand-up meeting team progress is tracked.
* **Sprint Review/ Retrospective:** It is also hosted by scrum master, it last about 2-4 hours and discuss what the team has accomplished in the last sprint and what lessons were learned.

**Role of Tester in Scrum**

[](https://www.guru99.com/images/11-2014/112714_1232_ScrumTestin3.png)

**There is no active role of Tester in the Scrum** Process. Usually, testing is carried out by a developer with Unit Test. While product owner is also frequently involved in the testing process during each sprint. **Some Scrum projects do have dedicated test teams depending on the nature & complexity of the project**.

The next question is, what tester do in a scrum? Following note will answer

**Testing Activities in Scrum**

Testers do following activities during the various stages of Scrum-

**Sprint Planning**

* In sprint planning, a tester should pick a user-story from the product backlog that should be tested.
* As a tester, he/she should decide how many hours (Effort Estimation) it should take **to finish** testing for each of selected user stories.
* As a tester, he/she must know what sprint goals are.
* As a tester, contribute to the prioritizing process

**Sprint**

* Support developers in unit testing
* Test user-story when completed. **Test execution is performed** in a lab where both tester and developer work hand in hand. Defect are logged in [Defect Management tool](https://www.guru99.com/top-20-bug-tracking-tools.html) which are tracked on a daily basis. Defects can be conferred and analyzed during the scrum meeting. Defects are retested as soon as it is **resolved** and deployed for testing
* As a tester, he/she attends all daily standup meeting to speak up
* As a tester, he/ she can bring any backlog item that cannot be completed in the current sprint and put to the next sprint
* Tester is responsible for developing automation scripts. He schedules automation testing with [Continuous Integration (CI) system](https://www.guru99.com/continuous-integration.html). Automation receives the importance due to short delivery timelines. Test Automation can be accomplished by utilizing various open source or paid tools available in the market. This proves effective in ensuring that everything that needs to be tested was covered. Sufficient Test coverage can be achieved with a close communication with the team.
* Review CI automation results and send Reports to the stakeholders
* Executing non-functional testing for approved user stories
* Coordinate with customer and product owner to define acceptance criteria for Acceptance Tests
* At the end of the sprint, the tester also does acceptance testing(UAT) in some case and confirms testing completeness for the current sprint

**Sprint Retrospective**

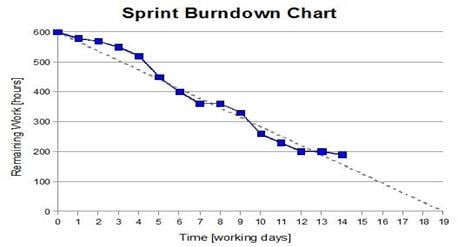
* As a tester, he will figure out what went wrong and what went right in the current sprint
* As a tester, he identifies lesson learned and best practices

**Test Reporting**

Scrum Test metrics reporting provides transparency and visibility to stakeholders about the project. The metrics that are reported allow a team to analyze their progress and plan their future strategy to improve the product. There are two metrics that are frequently used to report.

**Burn down chart:** Each day, Scrum Master records the estimated remaining work for the sprint. This is nothing but the Burn Down Chart. It is updated daily.

A burndown chart gives a quick overview of the project progress, this chart contains information like the total amount of work in the project that must be completed, amount of work completed during each sprint and so on.

[](https://www.guru99.com/images/11-2014/112714_1232_ScrumTestin4.jpg)

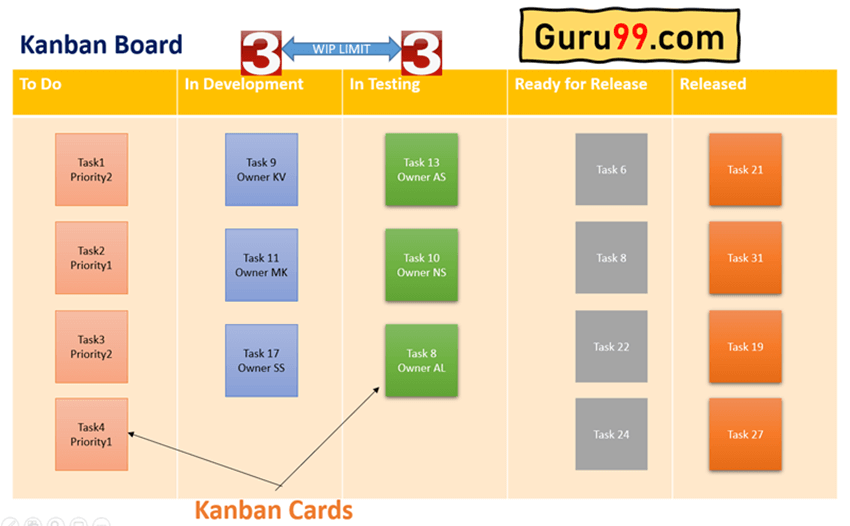
**Velocity history graph:** The velocity history graph predicts the velocity of the team reached in each sprint. It is a bar graph and represents how teams output has changed over time.

The additional metrics that may be useful are schedule burn, budget burn, theme percent complete, stories completed – stories remaining and so on.

**Kanban** is a very popular framework for development in the agile software development methodology. It provides a transparent way of visualizing the tasks and work capacity of a team. It mainly uses physical and digital boards to allow the team members to visualize the current state of the project they are working on.

* Kanban can be used in any domain, and it can be used very effectively in software development. Kanban project management helps in improving the efficiency of the team.
* It is a pull-based system. Tasks are being pulled as soon as an individual is free.
* Kanban should be used when you want to release your work at any time. It requires git branching, but it is doable.
* Kanban should be used when you want to change the priorities on the fly. For that, all you need to do is to put this story on the top of the to-do queue.
* It should be used when you want to visualize your work, and you want to see the progress of your tasks visually.

The Kanban system recommends visualization of work. It suggests the use of the physical and the digital board.

[](https://www.guru99.com/images/2/what-is-Kanban-1.png)Kanban Cards

The Kanban cards are essential pieces on the Kanban board as it represents the work that the team is working on. These cards will have

1. Priority
2. Owner
3. Type
4. Due date

A column in Kanban board represents the work stage, and you can place a WIP (Work in Progress) limit on the column. **The WIP limit means the maximum number of cards that can stay on that column**.

Since Kanban project management uses a pull-based system, as and when a developer is free, he/she can pull a card from the to-do column to the dev column.

**Kanban Board**

**Kanban Board** is an agile project management tool that helps implement Kanban to manage projects for personal and business purposes. It is a physical or digital (JIRA) board designed to help teams visualize their work at different stages and processes. It also helps represent the stages of work with columns using cards.

It has columns that represent the status of the work like

1. To-do,
2. Dev
3. Testing
4. Done.

Each of these columns can have cards <=the WIP limit. The cards represent the actual work.

You can use positive numbers to limit work-in-progress, and this limit number can be placed on the top of the columns in both physical and digital Kanban boards. Any individual of the team can manage the state of his card, and the entire team can visualize the workflow. Next in this Kanban tutorial, we will learn about Kanban Workflow.

**Kanban Workflow**

**Kanban Workflow** is a set of steps that helps teams to define explicit policies and principles in Kanban. It represents the rules and procedures while the work is going on across various stages of development and delivery cycles. Kanban workflow consists of step-by-step processes between starting and the delivery of a particular task.

The basic principal Kanban follows is, **“stop starting, start finishing”.** With the help of WIP limits, it gets more work done. There are customizable Kanban workflows and states available in any modern tool like JIRA.

Below are the basic states that many software teams follow for their workflow management.

| **States** | **Understanding of tasks** |
| --- | --- |
| **To-do** | Tasks arrive here for the first time in this state. |
| **Ready for analysis** | Analyze the task and add requirements completely. |
| **Ready for development** | Analysis completed and development can start. |
| **In the development** | Tasks are being developed. |
| **Ready for testing** | Development completed, and now testing can start. |
| **In the testing** | Tasks are being tested. |
| **Ready for release** | Testing completed; release can happen. |
| **Released/Done** | Released. |

**The Four Principles of Kanban**

Below are the main Four core principles of Kanban:

1. **Start with what you have now**: Kanban system suggests working incrementally and start with what you have currently. Since one of its practice is to improve continuously, you must improve the system gradually.
2. **Agree to Pursue Incremental, Evolutionary Change:** Kanban recommends an incremental change in the process, and you must not make a big change in the process in one go.
3. **Respect the Current Process, Roles & Responsibilities:** Once again, start with what you have now and change the process, role, and responsibilities in an incremental manner.
4. **Encourage Acts of Leadership at All Levels**: Every individual can act as a leader and provide ideas to improve the efficiency of the overall Kanban system. You should not think that this is a management level activity, and even the youngest member of the team can act as a leader.

**The Six Kanban Core Practices**

Following are the main Six core practices of Kanban:

1. **Visualize the workflow**: This principle suggests having a Kanban board (physical or digital) to visualize the workflow. Each individual of a team must see his card and cards of other team members. You can move your cards in different columns as per the above image. It brings lots of transparency within the team and also makes it easier to resolve blockers
2. **Limit work in progress**: Kanban is a pull-based system, and it improves the efficiency of a team to limit work in progress and have tasks that can be completed in the given time frame by the team. This WIP limit applies from the beginning to the end of the workflow. You can apply the limit on top of the column using a positive integer.
3. **Focus on flow**: This principle focuses on flow and on any interruptions. If there are interruptions or blockers, they must be fixed permanently.
4. **Explicit Policies**: Policies can be established in a team to reduce the rework and focus on the areas which require attention or where it is more effective.
5. **Feedback Loop**: Feedback loops are very essential in Kanban. It is not just within the team but between multiple teams, coaches, etc. This helps in improving the overall health of the Kanban system.
6. **Continuous Improvement**: This is the core principle of the Kanban system. It states that you can always improve the process, and that will result in better efficiency.

**Pull Based System**

Kanban is a pull-based method where tasks are being pulled rather being pushed. As soon as you have completed your current card, you can pull a new card from the previous column of the Kanban board.

With the WIP limit, Kanban helps in the improvement of Lead Time and Cycle Time. There should be the least possible gap between these two timings. For example, we have 5 developers and just 1 tester; what will happen in this case? There would always be many cards that require testing, and they will be sitting idle and waiting.

To overcome the issues mentioned above and improve efficiency, Kanban follows the pull-based approach with WIP limits, where there would be a limited number of cards to be pulled.

So, a tester will pull a task from the “ready for testing” stage when he has finished his current task in hand. With the WIP limit in Kanban columns (stages of development), you will not have many unattended cards in the Kanban workflow.

The pull-based system also helps in finding the correct velocity for the team. With the right velocity in place, the team will perform better.

**Lead Time and Cycle Time**

In Kanban method, lead time and cycle time are widely used, there is a difference between the two, and it is important to understand that to avoid the confusion.

| **Lead Time** | **Cycle Time** |
| --- | --- |
| Lead time is measured as the time between the task’s arrival in your workflow and its departure from the workflow, meaning it has been released. | Cycle time is measured as the time between the task’s arrival in “in progress” state and the arrival of the task in “ready for release”. |

Here it is also important to understand not to include the time it takes between ready for release and actual release.

Cycle Time = Work in Progress/Throughput

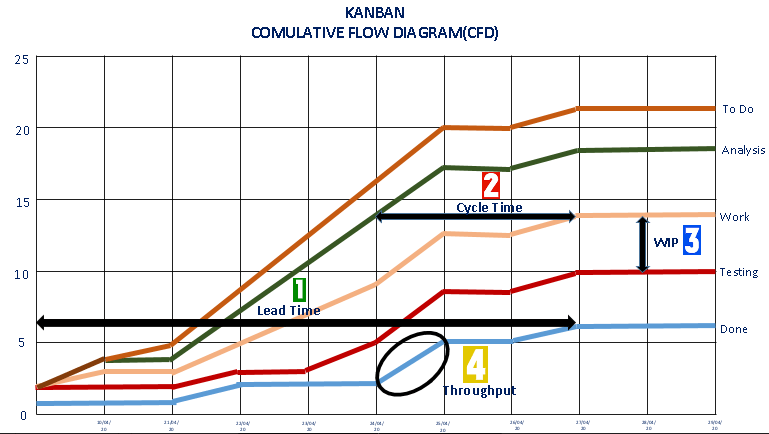
In the ideal scenario, the gap between lead time and cycle time should be minimal, and Kanban uses a cumulative flow diagram (CFD) to measure lead and cycle time historical data.

**Cumulative Flow Diagram (CFD)**

CFD is a chart which is available in all leading [workflow management tools](https://www.guru99.com/workflow-management-software-tool.html) like JIRA. This chart measures the total amount of work cards/tasks that entered the workflow and amassed completed cards/tasks over time.

It helps you to have an estimate of average lead time and cycle time for prespecified time.

CFD diagram will give you indicators or problem areas to fix. It will provide you with a clear picture, and based on this diagram. You can correct your team’s lead time and cycle time.

[](https://www.guru99.com/images/2/what-is-Kanban-2.png)Kanban Cumulative Flow Diagram

1. **Lead Time**: It is the duration between a new card’s arrival in your workflow and its final departure from the workflow.
2. **Cycle Time**: It is a duration between the card’s arrival in the working state and when the card is ready for release.
3. **WIP**: Work in progress (WIP) limits the maximum amount of work items in the different stages of the workflow.
4. **Throughput**: It is the actual performance, and it tells the actual number of cards delivered in a given timeframe.

Throughput = WIP/Cycle Time

**Limiting WIP(Work-in-progress)**

In Kanban development methodology, WIP limits the number of tasks/cards which can be worked on by a team member or entire at one time.

The WIP limits ensure that the team stabilizes their work and increases the predictive nature, which is essential in the pull-based system. Usually, the WIP limit decision is taken by the team itself.

**Reason to set the WIP Limits**

Here are reasons to set the WIP Limits:

* It shifts focus on getting things done as an individual focuses on a single task at a time.
* It helps teams to understand their capacity.
* It improves productivity lead and cycle time.
* It helps in avoiding the piling up tasks (in waiting mode).
* It helps in the movement of the workflow and tasks keep moving.
* It also helps to resolve blockers as an individual don’t switch between different tasks.

| **Scrum** | **Kanban** |
| --- | --- |
| Scrum **stresses on planning**. It starts with sprint planning and ends up with sprint retrospective.There are many meetings held which help to assure that the team is aligned with the next steps, priorities, and learnings from previous sprints. | Kanban is open to making changes on the go. It means there is less rigidity and **things can change frequently**. |
| It recommends collection of **time measurements** made during sprints | Kanban **recommends graphs** to get an overview of team’s progress over time. |
| Scrum **no longer** asks for a commitment from teams. Instead, it is about the sprint goals and forecasts. | Kanban relies on **time-boxing and forecasts**. |
| It stresses on planning, and so **estimation has a very important role** in Scrum | Kanban has **no mandatory requirements** for estimation. |
| Every **individual has their role** and responsibilities. | No **set roles so flexibility** in term of individual responsibilities. |
| The iterations/Sprints are fixed in duration. This duration varies from 2 weeks to 1 month. | Kanban is **not based on duration**. This thing is measured regarding Cycle times. |
| Teams are **required to commit** a specific amount of work. | **Commitment not necessary** it is optional for teams. |
| In this method, **cross-functional teams** are important as they can deal with any disruption that may cause a bottleneck in the software development. | Having **specialized team** is important. |
| It is **not possible to add items** to ongoing iterations. | New **items can easily add** if the additional capacity is available. |
| A sprint backlog is owned by only by a **single team**. | **Multiple team**s can share Kanban board. |
| Deliverables are **determined by sprints**, which a set of work must be completed and ready for review. | Products and processes are **delivered continuously** on a needed basis. So testing and review process goes on simultaneously. |
| Scrum software development method **focuses on the backlog**. | Kanban method entirely **focuses on process dashboard**. |
| Every **team member has a specific role** in Scrum master decide timelines, product owner set goals and objectives, and team members conduct the development work. | There are no pre-defined roles for a team. However, there may still be a Project Manager; the team is encouraged to collaborate and works together. |
| Best for projects with **changing priorities**. | Ideal for teams with **stable priorities** that unlikely to change over time. |
| Measures production **using velocity** through sprints. | Measures production using **cycle time** or the exact time it takes to complete one full piece of a project. |
| Scrum requires a **complete shift from the traditional model** to the Agile Scrum model that would be implemented the project. | Kanban **doesn’t allow drastic changes** in the project. |
| It is an ideal method for projects with **widely-varying priorities**. | Best suitable for **teams with stable priorities**. |
| In Scrum, the entire t**eam focuses on to collaborate and complete the task** to provide quality development work. | **Teams work to achieve goals** and reduce the time to complete the entire process. Thus, reduction in the time cycle is biggest indicators of success here. |
| Scrum **emphasis on its schedules**; new items cannot be added to ongoing iterations. | Kanban is more iterative by nature as it **does not have specific timeframes**. So that, new items can be continually added whenever additional capacity is available. |
| The total work is done in **batches/Sprints**. | The entire project is performed on the movement of **single-threaded work item** flows. |
| **Scrum master** acts as a problem solver. | Kanban encourages **every team member is a leader** and sharing responsibility amongst them all. |
| Scrum prescribes **time-boxed iterations**. | Kanban focuses on **planning a different duration** for individual iteration. |
| Scrum helps firms to **save time and money**. | Kanban method **focus on continuous improvement**, productivity, and efficiency. |
| Achieve **stable and consistent communication** of performance at all levels. | Team members are more likely to **accomplish their goals much easier** because of visual nature of Kanban boards. |
| Project are **coded and tested during the sprint** review | Team members are more likely to **accomplish their goals much easier** because of visual nature of Kanban boards. |
| It is **easier to adapt to the constant changes** because of the short sprints and regular feedback. | It is **designed for a regular, steady output**, major changes in customer demand can make Kanban fail. |
| The total cost of the project is minimal which may lead to **quicker and cheaper result**. | If a task is not correctly estimated, the **total project cost will never be accurate**. In such cases, the task can be spread over several sprints. |
| This methodology **requires experienced team members** only. So, If the team consists of people who are not an expert, the project cannot be completed in time. | No **specific timeframes** are allocated with each phase, so team members never get the idea how much time they can take in every phase. |
| In this Agile Scrum method, it is **easier to deliver a quality product** at a scheduled time. | It is designed for a **regular, steady output,** major changes in customer demand can make Kanban fall. |
| The **project plan will never disturb** even if a team member leaves the team. | If any of the team members exit during development, it can **hurt the project development**. |
| Daily meetings sometimes **frustrate** team members. | **Outdated Kanban board** can lead to issues in the development process. |
| **Large projects can easily divided** into easily manageable sprints. |  |