

Software Testing Help

SEARCH

Home Resources FREE EBooks QA Testing ▾ Courses ▾ Automation ▾ Types Of Testing ▾ Tutorials ▾ Data ▾

Agile Methodology: A Beginner’s Guide To Agile Method And Scrum

Last Updated: November 1, 2021

A Complete Guide to Agile Methodology: (20+ Detailed Agile Scrum Methodology Tutorials)


This is the guide for software developers and testers to understand and start working on the very famous *Agile SCRUM methodology for software development and testing*. Learn the basic but important terminologies used in the Agile Scrum process along with a real example of the complete process.

We have listed all the Agile Tutorials in this series for your convenience. Hope they will be of immense help to you.


Topics covered: What is Agile, What is Scrum, Agile Methodology in Software Development and Testing, Agile Testing, Agile Scrum Process, Scrum Methodology with Scrum Team and Scrum Master.

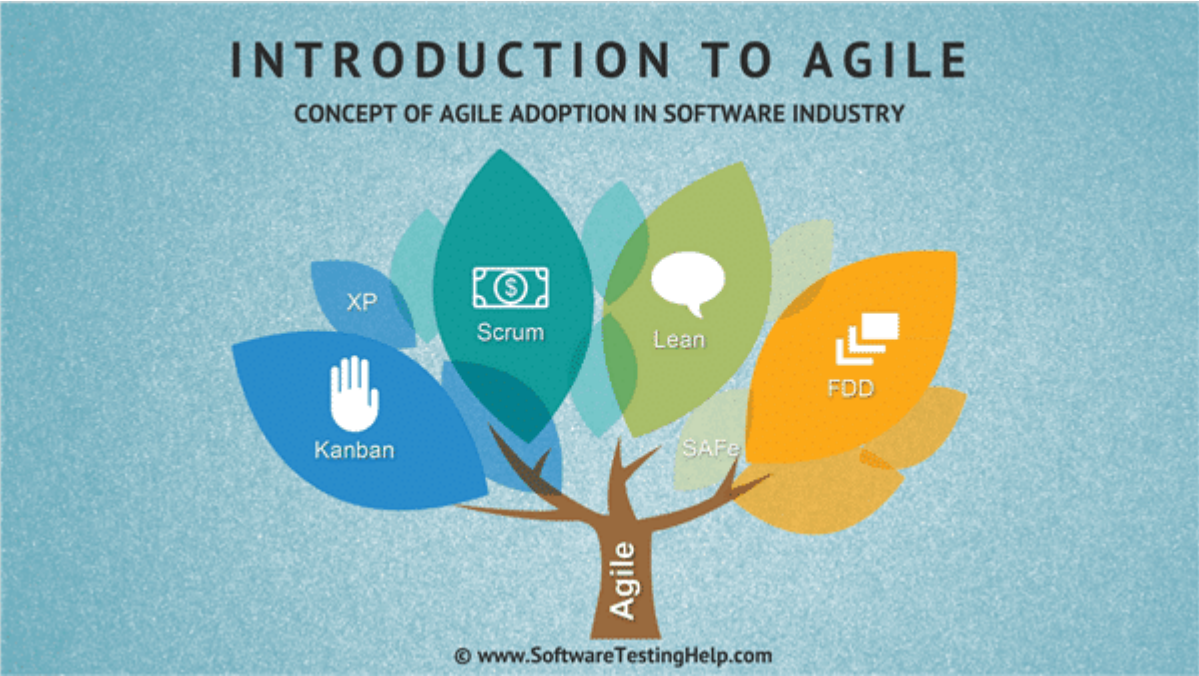
About SoftwareTestingHelp

Helping our community since 2006!
Most popular portal for Software professionals with 100 million+ visits and 300,000+ followers! You will absolutely love our tutorials on QA Testing, Development, Software Tools and Services Reviews and more!

An advertisement for Nokia. It features a woman with glasses and curly hair, smiling and making a peace sign. The word "Open" is written in large white letters. To the right of "Open" is a list of phrases: "to new opportunities", "to learning", "to growth", "to collaborate", "to challenges", and "to feedback". The Nokia logo is in the bottom left corner.

Open to new opportunities?

 Nokia



A promotional banner for BYBIT. It features the BYBIT logo at the top left. The main text reads "Ready for Blast-Off?" in a large, bold font. Below this, it says "Deposit & Earn Up to \$3,000 Bonus". A prominent orange button with the text "Join Now" is centered. To the right, there is an illustration of a yellow rocket launching upwards, with several gold coins floating around it. At the bottom left, a small disclaimer states "*For first time deposits only!".

Recommended Reading

- Agile Scrum Online Quiz: Test Your Knowledge of Agile Scrum
- Kanban vs Scrum vs Agile: A Detailed Comparison to Find Differences
- How to Deliver High Value Software Features in a Short Time Period using Agile Scrum Process
- Agile Manifesto: Understanding Agile Values and Principles
- SAFe Agile Tutorial: What is Scaled Agile Framework
- 30+ Top Scrum Interview Questions and Answers [2021 LIST]
- Agile Vs Waterfall: Which Is The Best Methodology For Your Project?



DevOps Practice Based on Agile Manifesto



What is System Testing



What Is Jest? - Meaning & Introduction



Jest Matchers And Its Types

What You Will Learn: [\[hide\]](#)

[Agile Methodology Tutorials List](#)

Introduction to Agile Development

History of Agile

Agile Challenges

What are Agile Promises?

What Exactly is Agile?

How to Practice Agile?

Agile Methodology

Advantages of Agile Methodology

Disadvantages of Agile Methodology

Types of Agile Methodologies

#1) Scrum

#2) Kanban

#3) Lean

#4) Extreme Programming (XP)

Scrum Methodology

Important SCRUM Terminologies

Activities Done in SCRUM Methodology

#1) Planning Meeting

#2) Execution of Sprint Tasks

#3) Daily Standup

#4) Review Meeting

#5) Retrospective Meeting

How the Process is done? An Example!

SCRUM Activity Tools

Recommended Reading

Top 31 Agile Interview Questions and Answers

JOIN Our Team!

WRITE AND EARN

EARN \$\$\$ WORKING FROM HOME
AS A FREELANCE WRITER

★

APPLY NOW!

★

www.SoftwareTestingHelp.com

Agile Methodology Tutorials List

- Tutorial #1: Agile Scrum Methodologies *(This Tutorial)*
- Tutorial #2: Agile Manifesto
- Tutorial #3: Scrum Team & their roles and responsibilities
- Tutorial #4: Scrum Artifacts
- Tutorial #5: Scrum Events

Tutorial #6: Defect Triaging In Scrum

Tutorial #7: Self Sufficient Scrum Teams

Tutorial #8: Three Amigo Principle

Tutorial #9: SAFe – Scaled agile framework

Tutorial #10: Agile Scrum Quiz

MORE Recommended Agile Scrum Tutorials:

Tutorial #11: Top Agile Estimation Techniques

Tutorial #12: Agile Waterfall Hybrid Model

Tutorial #13: Kanban vs Scrum vs Agile

Tutorial #14: JIRA Agile Tutorial

Tutorial #15: Agile Retrospective Meetings

Tutorial #16: Role of Business Analysts in SCRUM

Tutorial #17: QA's Role in Scrum

Tools and Interview Questions:

Tutorial #18: Agile Testing Tools

Tutorial #19: Best Agile Project Management Tools

Tutorial #20: Top Agile Interview Questions

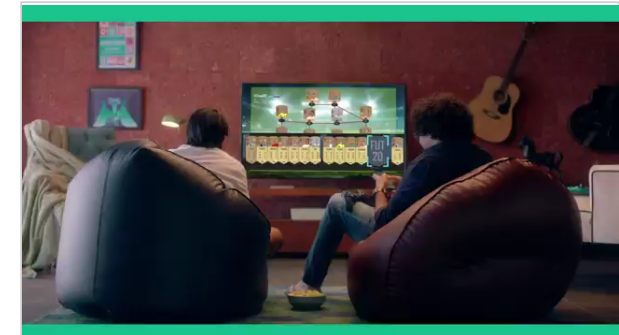
Tutorial #21: Top Scrum Interview Questions

Let's start with the first tutorial in the series – Agile Scrum Introduction.

Introduction To Agile Development

Agile in Software Development:

Agile is one of the world's most widely used and recognized software development framework.



SPONSORED BY MUTUAL FUNDS SAHI HAI

Is There A Dashboard For Mutual Fund Performance?

Mutual Fund performance comparison can be done via dashboard. Watch our video to know how you can analyse and evaluate the performance of Mutual Funds. #MutualFundsSahiHai

SEE MORE

Most of the organizations have adopted it in some form or the other but there is still a long way to go in the maturity of their adoption programs. The sole aim of this series of tutorials is to onboard technology and non-technology professionals into the Agile World.

We will take you through the agile journey in a step by step manner until you understand the philosophy behind using Agile, its advantages and how to practice it. This series aims to equip and enable the readers to apply Agile and Scrum learning into their work.

This particular tutorial is dedicated to explaining to you why there was a need for Agile and how it got created. The fundamental here is to make you understand the concept of Agile Adoption in Software Development Industries.

History of Agile

Agile was born when on one fine day when 17 people with different development methodologies background, got together to brainstorm if there was a possible alternative solution to software development which could lead to faster development time and was less documentation heavy.

At that time, software development used to happen so long that by the time projects were ready to be delivered, the business had moved ahead and the requirements had changed. Thus a project was not able to meet the business needs even if it was able to meet its defined objectives.

Thus these champions of different software engineering techniques got together and the end result of their meeting was what they called the “agile manifesto” which we will be discussing in detail in the next tutorial of this series.

But the agile that was born that day is not what we see today in organizations. The methodology those experts agreed upon was described as ‘lightweight’ and fast. But the main win out of this meeting was the thought that faster delivery of a product and constant feedback were the keys to achieve success in software development.

The existing waterfall techniques were too cumbersome and had no provision for feedback until the final product was ready to be delivered. This meant that there was no scope for course correction and the

customer had no view on the progress until the whole product was ready. And that was what these experts wanted to avoid.

They wanted a solution which would have scope for constant feedback in order to avoid the cost of rework at a later stage.

Agile Challenges

The existing waterfall techniques at that time were too cumbersome and had no provision for feedback until the final product was ready to be delivered. It was called a waterfall model of development because the teams first finished one step completely and only after that they moved ahead to the next step.

This meant that there was no scope for course correction and the customer had no view on the progress until the whole product was ready. And that was what these experts wanted to avoid. They wanted a solution which would have scope for constant feedback in order to avoid the cost of rework at a later stage.

And that is why agile is also about being adaptive and continuous improvement, as much as it is about constant feedback and speed of delivery.

What are Agile Promises?



Agile is not only about applying the set practices in developing software. It also brings in a change in the Team's mindset which drives them towards building better software, working together and eventually landing them a happy Customer.

Agile values and principles enable the team to shift their focus and change their thought process of building better software.

What Exactly is Agile?

Agile is not a set of rules. Agile is not a set of guidelines. Agile is not even a methodology. Rather, Agile is a set of principles that encourage flexibility, adaptability, communication and working software over plans and processes. It is very succinctly captured in what is called the agile manifesto.

Agile software development allows the team to work together more efficiently and effectively in developing complex projects. It consists of practices that exercise iterative and incremental techniques which are easily adopted and display great results.

In apply Agile into action, we have various Agile-based methods and methodologies. These methods and methodologies cater all the needs of a software development industry right from the software design and architecture, development & testing to project management and deliveries.

Not just that, Agile methods and methodologies also open scope for process improvement as an integral part of each delivery.

Agile is a software development approach where a self-sufficient and cross-functional team works on making continuous deliveries through iterations and evolves throughout the process by gathering feedback from the end users.

How to Practice Agile?

There are various Agile Methodologies that are in practice in various diversified industries.



However, the most popular methodologies amongst all of them are:

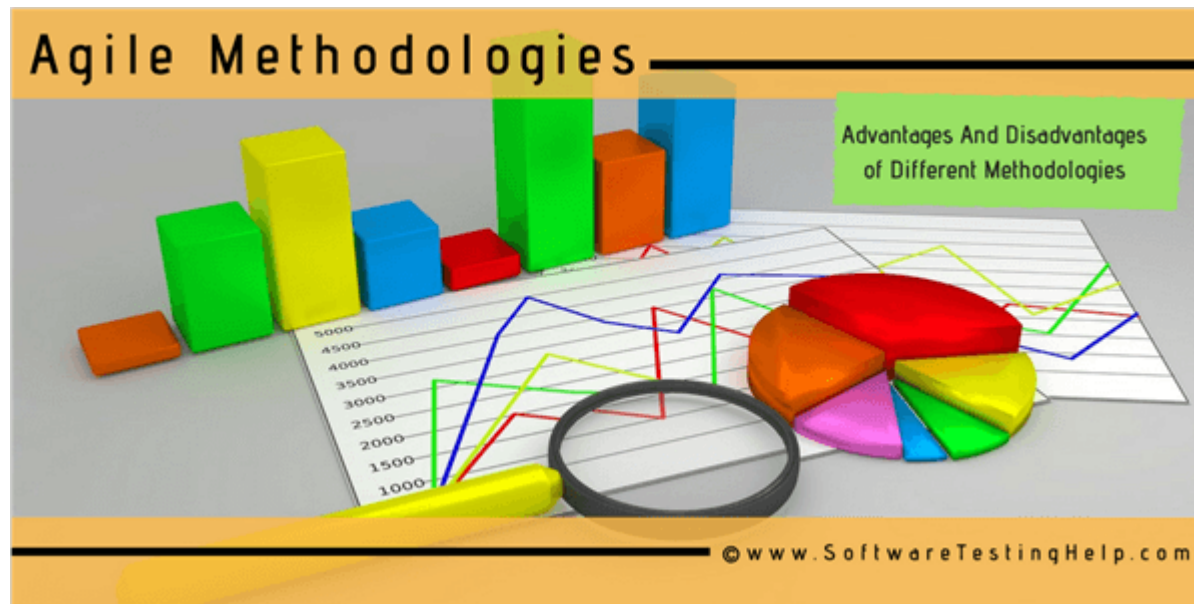
- Scrum
- Kanban
- Extreme Programming

All these methodologies focus on lean software development and help in building better software effectively and efficiently.

That is all with Agile Introduction. The part is structured to help you to understand the core values and principles that shall be adopted for a team to be working in an Agile mode and mindset.

Agile Methodology

Introduction to Agile Models:



As we all know, Agile is a software development methodology.

We have also learned about the values and principles which were mentioned in the agile manifesto by the founders of agile. In our initial discussions, we also skirted upon the differences between agile and the traditional waterfall models.

In this tutorial, we will get to know about the advantages and disadvantages of the agile methodology.

We will see what is scrum? and how is it different from agile. Then we will understand the various agile methodologies that are being used by different organizations and how can we implement agile using them.

You will also be able to appreciate the difference and also the advantages/disadvantages of these methodologies.

Advantages of Agile Methodology

Given below are the various advantages of Agile Methodology:

- The customers continuously get a look and feel of the project progress at the end of each iteration/sprint.
- Each sprint provides the customer with a working software which meets their expectations as per the definition of done provided by them.
- The development teams are quite responsive to the changing requirements and can accommodate changes even in the advanced stages of development.
- There is constant two-way communication which keeps the customers involved, thus all stakeholders – business and technical – have clear visibility on the project's progress.
- The design of the product is efficient and fulfills the business requirements.

Disadvantages of Agile Methodology

Though there are several advantages of Agile methodology, there are certain disadvantages involved in it too.

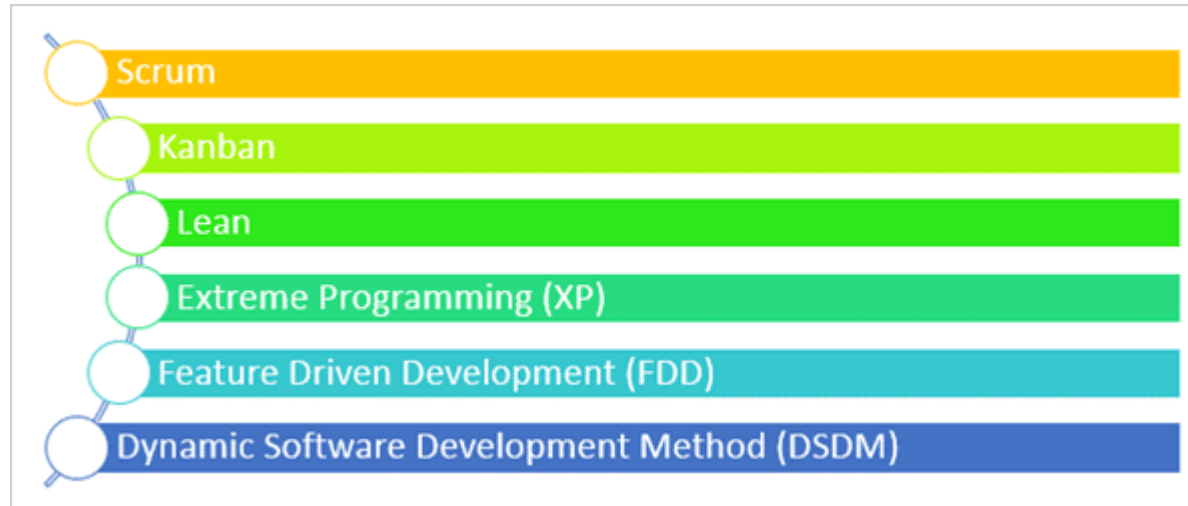
They are:

#1) Comprehensive documentation is not preferred which can lead to agile teams incorrectly interpreting this as agile doesn't require documentation. So the rigor gets lost on documentation. This should be avoided by continuously asking yourself if this is sufficient information to proceed or not.

#2) Sometimes, at the beginning of the projects, the requirements are not crystal clear. The teams might proceed and find that the customers' vision got realigned and in such situations, the teams need to incorporate many changes and it is difficult to gauge the end result as well.

Types of Agile Methodologies

There are several agile methodologies in practice across the world. We are going to learn more in detail about four of the most popular ones.



#1) Scrum

Scrum can easily be considered to be the most popular agile framework. The term 'scrum' is much considered synonymously to 'agile' by most practitioners. But that is a misconception. Scrum is just one of the frameworks by which you can implement agile.

The word scrum comes from sports rugby. Where the players huddle together in an interlocked position pushing against the opponents. Each player has a defined role in their position and can play both offensive and defensive as per the demand of the situation.

Similarly, the scrum in IT believes in empowered self-managed development teams with three specific and clearly defined roles. These roles include – Product Owner (PO), Scrum Master (SM) and the development team consisting of the programmers and testers. They work together in iterative time boxed durations called sprints.

The first step is the creation of the product backlog by the PO. It's a to-do list of stuff to be done by the scrum team. Then the scrum team selects the top priority items and tries to finish them within the time box called a sprint.

An easier way to remember all of this is to memorize the 3-3-5 framework. It means that a scrum project has 3 roles, 3 artifacts, and 5 events.

These are –

Roles: PO, Scrum master, and development team.

Artifacts: Product Backlog, Sprint Backlog and Product increment.

Events: Sprint, Sprint planning, Daily Scrum, Sprint review and Sprint retrospective.

We will get to know more in detail about each of these in our subsequent tutorials.

#2) Kanban

Kanban is a Japanese term which means a card. These cards contain details of the work to be done on the software. The purpose is visualization. Every team member is aware of the work to be done through these visual aids.

Teams use these Kanban cards for continuous delivery. Just like Scrum, Kanban is also for helping the teams work effectively and promotes self-managed and collaborative teams.

But there are differences between these two as well – like during a scrum sprint, the items being worked upon by a team are fixed and we cannot add items to the sprint whereas, in Kanban, we can add items if there is available capacity. This is particularly useful when the requirements change frequently.

Similarly, another difference is that while the scrum has defined roles of a PO, scrum master, and development teams, there are no such pre-defined roles in Kanban.

Another difference is that while the scrum suggests a prioritization of product backlogs, Kanban has no such requirement and it is totally optional. Thus Kanban requires less organization and avoids non-value adding activities and is suitable for the processes which require responsiveness towards changes.

#3) Lean

Lean is a philosophy that focuses on waste reduction. How does it do that?

In lean, you divide a process into value-adding activities, non-value adding activities and essential non-value adding activities. Any activity which can be classified as a non-value adding activity is a waste and we should try to remove that wastage in the process to make it leaner.

A leaner process means faster delivery and less effort wasted in tasks which don't help to achieve the team goals. This helps to optimize every step in the software development cycle. That is why the lean principles were adapted from lean manufacturing into software development.

Lean software development can be used in any IT project by applying the seven lean principles which are shown below:

These are quite self-explanatory as their names suggest. Eliminating wastage is the first and most important lean principle and we saw how to do that, we just classify activities as value and non-value add.

A non-value add activity can be any part of the code which might make it less robust, increase the effort involved and take up a lot of time while not adding justifiable business value. It can also be vague user stories or poor testing or adding features that are not required in the big picture.

The second principle amplifying learning is again easy to understand as a team needs various skills to deliver products in a rapidly changing environment with new technologies cropping up in quick durations.

Making late decisions can be rewarding in circumstances where it reduces rework like if there are any changes expected then better delay it so that the team does not have to redo the work as the business needs change.

But there is always a trade-off here as the teams need to balance this with the fourth principle of delivering faster. Delaying of decisions should not impact the overall delivery and must not reduce the pace of work. One eye should always be on the complete picture.

Having empowered teams is also very common nowadays and this is something that even agile suggests. Empowered teams are more responsible and can take decisions faster. The sense of ownership in an empowered team leads to better results. To empower a team, they should be allowed to organize themselves and take decisions on their own.

Thus we see that lean and agile have a lot in common with one stark difference – while lean teams can help to refine a product, agile teams are the ones who actually build the product.

#4) Extreme Programming (XP)

Extreme programming is another most popular agile techniques. As per extremeprogramming.org, the first XP project was started on March 6, 1996. They also mention that XP impacts software project development in 5 different ways – communication, simplicity, feedback, respect, and courage. These are called the values of XP.

Out of these, it all starts with communication. XP teams collaborate with business teams and fellow programmers on a regular basis and start building code from the very first day. The focus here is on face to face communication as far as possible with the help of the other visual aids.

Extreme programmers also build a simple code and start getting feedback on it from the first day itself. The focus is on not going overboard or predicting requirements which have not been shared. This keeps the design simple and produces just the minimum product which will serve the requirements.

Feedback helps the team to improve and produce a better quality of work. This helps them build respect for each other as they learn from each other and learn how to share their views.

This also gives them the courage as they know that they have gathered everyone's best ideas and produced a good product with feedback from others. Thus they are also not afraid to include changes or receive further feedback on their work.

This is particularly useful in projects where the requirements are going to change often. Constant feedback will help the teams in including these changes with courage.

Thus we have seen the different agile methodologies like Scrum, XP, Kanban and Lean along with their respective advantages and disadvantages.

Now, we can easily differentiate between them and also appreciate the subtler differences amongst them. We also learned the fundamentals of each of these methodologies and saw how to apply them in our projects as and when required.

In next part, let's understand everything about Scrum.

Scrum Methodology

SCRUM is a process in agile methodology which is a combination of the Iterative model and the incremental model.

One of the major handicaps of the traditional **Waterfall model** was that – until the first phase is complete, the application does not move to the other phase. And by chance, if there are some changes in the later stage of the cycle, then it becomes very challenging to implement those changes, as it would involve revisiting the earlier phases and redoing the changes.

Some of the key characteristics of SCRUM include:

- Self-organized and focused team.
- No huge requirement documents, rather have a very precise and to the point stories.
- The cross-functional teams work together as a single unit.
- Close communication with the user representative to understand the features.
- Has a definite timeline of maximum one month.
- Instead of doing the entire “thing” at a time, Scrum does a little of everything at a given interval.
- Resources capability and availability are considered before committing anything.

To understand this methodology well, it's important to understand the key terminologies in SCRUM.

Also read => [How to Deliver High-Value Software Features in a Short Time Period using Agile Scrum Process](#)

Important SCRUM Terminologies

1) Scrum Team

The scrum team is a team comprising of 7 with + or – two members. These members are a mixture of competencies and comprise of developers, testers, database people, support people etc. along with the

product owner and a scrum master.

All these members work together in close collaboration for a recursive and definite interval, to develop and implement the said features. SCRUM team sitting arrangement plays a very important role in their interaction, they never sit in cubicles or cabins, but a huge table.

2) Sprint

Sprint is a predefined interval or time frame in which the work has to be completed and make it ready for review or ready for production deployment. This time box usually lies between 2 weeks to 1 month.

In our day to day life when we say that we follow 1-month Sprint cycle, it simply means that we work for one month on the tasks and make it ready for review by the end of that month.

3) Product Owner

The product owner is the key stakeholder or the lead user of the application to be developed. The product owner is the person who represents the customer side. He/she has the final authority and should always be available for the team.

He/she should be reachable when anyone has any doubts that need clarification. It is important for the product owner to understand and not to assign any new requirement in the middle of the sprint or when the sprint has already started.

4) Scrum Master

Scrum Master is the facilitator of the scrum team. He/she makes sure that the scrum team is productive and progressive. In case of any impediments, scrum master follows up and resolves them for the team. SCRUM Master is the mediator between the PO and the team.

He/she keeps the PO informed about the progress of the Sprint. If there are any roadblocks or concerns for the team, discusses with the PO and gets them resolved. Like the team's Daily Standup, a standup of the SCRUM Master with the PO happens every day.

Recommended read => [How To Be a Good Team Mentor, Coach and a True Team-Defender in an Agile Testing World?](#)

5) Business Analyst (BA)

A Business Analyst plays a very important role in SCRUM. This person is responsible for getting the requirement finalized and drafted in the requirement docs (based on which the user stories are created).

If there are any ambiguities in the User Stories / Acceptance criteria, he/she is the one who is approached by the technical (SCRUM) team and he then takes it up to the PO or else if possible resolves on his own. In large scale projects, there may be more than 1 BA but in small-scale projects, the SCRUM Master may be acting as the BA as well.

It is always a good practice to have a BA when the project kick starts.

6) User Story

User stories are nothing but the requirements or feature which has to be implemented.

In the scrum, we don't have those huge requirements documents, rather the requirements are defined in a single paragraph, typically having the format as:

As a <User / type of user>

I want to <Some achievable goal/target>

To achieve <some result or reason for doing the thing>

For Example:

As an Admin I want to have a password lock in case the user enters an incorrect password for 3 consecutive times to restrict unauthorized access.

There are some characteristics of user stories which should be adhered. The user stories should be short, realistic, could be estimated, complete, negotiable and testable. A user story is never altered or changed in the middle of the Sprint.

It is the responsibility of the SCRUM Master and the BA (if applicable) to make sure that the PO has drafted the User Stories correctly with a proper set of the Acceptance Criteria". If any changes which will impact the sprint release are to be made, then such stories are pulled out of the sprint or they are done as per the hours available.

Every user story has an acceptance criterion which should be well defined and understood by the team.

Acceptance criteria details down the user story that provide the supporting documents. It helps to further refine the user story. Anybody from the team can write down the acceptance criteria. Testing team bases their test cases/conditions on these acceptance criteria.

7) Epics

Epics are equivocal user stories or we can say that these are the user stories which are not defined and are kept for future sprints.

Just try to relate it with life, imagine you are going for a vacation. As you are going next week, you have everything in place like your hotel bookings, sightseeing, travelers check, etc. But what about your vacation plan for next year? You only have a vague idea that you may go to XYZ place, but you have no detailed plan.

An Epic is just like your next year's vacation plan, where you just know that you may want to go, but where, when, with whom, all these details you have no idea at this point of time.

In a similar way, there are features which are required to be implemented in the future whose details are not yet known. Mostly a feature begins with an Epic and then is broken down to stories which could be implemented.

8) Product Backlog

The product backlog is a kind of bucket or source where all the user stories are kept. This is maintained by the Product Owner. The product backlog can be imagined as a wishlist of the product owner who prioritizes it as per the business needs.

During the planning meeting (see next section), one user story is taken from the product backlog, then the team does the brainstorming, understands it and refines it and collectively decides which user stories to take, with the intervention of the product owner.

9) Sprint Backlog

Based on the priority, user stories are taken from the Product Backlog as one at a time. The Scrum team brainstorms on it, determines the feasibility and decides on the stories to work on a particular sprint. The collective list of all the user stories which the scrum team works on a particular sprint is known as Sprint backlog.

10) Story Points

Story points are a quantitative indication of the complexity of a user story. Based on the story point, estimation and efforts for a story are determined.

A story point is relative and not absolute. In order to make sure that our estimate and efforts are correct, it's important to check that the user stories are not big. The more precise and smaller is the user story, the more accurate will be the estimation.

Each and every user story is assigned to a story point based on the Fibonacci series (1, 2, 3, 5, 8, 13&21). Higher is the number, the complex is the story.

To be precise

- If you give 1 / 2 / 3 story point it means that the story is small and of low complexity.
- If you give points as 5 / 8, it is a medium complex and

- 13 and 21 are highly complex.

Here complexity consists of both development plus testing effort.

To decide a story point, brainstorming happens within the scrum team and the team collectively decides a story point.

It may happen that the development team gives a story point of 3 to a particular story, because for them it may be 3 lines of code change, but the testing team gives 8 story point because they feel that this code change will affect larger modules so the testing effort would be larger. Whatever story point you are giving, you have to justify it.

So in this situation, brainstorming happens and the team collectively agrees to one story point.

Whenever you are deciding on a story point, keep the below factors in mind:

- The dependency of the story with other application/module.
- The skill-set of the resource.
- The complexity of the story.
- Historical learning.
- Acceptance criteria of the user story.

If you are not aware of a particular story, don't size it.

Whenever a story is = or > 8 points, it is broken down into 2 or more stories.

11) Burn down chart

Burn down chart is a graph which shows the estimated v/s actual effort of the scrum tasks.

It is a tracking mechanism by which for a particular sprint the day to day tasks are tracked to check whether the stories are progressing towards the completion of the committed story points or not.

Example: To understand this, check the below figure:

I have assumed:

- 2 weeks Sprint (10 days)
- 2 resources actual working on the sprint.

“Story” -> This column shows the user stories taken for the sprint.

“Task” -> This column shows the list of the task associated with the user story.

“Effort” -> This column shows the effort. Now, this measure is the total effort to complete the task. It does not depict the effort put in by any specific individual.

“Day 1 – Day 10” -> This column(s) shows the hours which are left to complete the story. Please see that the hour is NOT the hour which is already done BUT the hours which are still left.

“Estimated Effort” -> Is the total of the effort. For the “Start” it is simply the sum of the entire individual task: SUM (C5: C15)

A total number of effort that has to be completed in 1 day is $70 / 10 = 7$. So at the end of day 1, the effort should reduce to $70 - 7 = 63$. In a similar way, it is calculated for all the days till day 10, when the

estimated effort should be 0 (Row 16)

“Actual Effort Left” -> As the name suggests, is the effort actually left to complete the story. It may also happen that the actual efforts increases or decreases than the estimated one.

You can use the inbuilt functions and Chart in Excel to create this burndown chart.

Burn Down Chart steps would be:

1. Enter all the stories (Column A5 – A15).
2. Enter all the Tasks (Column B5 – B15).
3. Enter the Days (Day 1 – Day 10).
4. Enter the starting efforts (Sum the tasks C5 – C15).
5. Apply the formula to calculate the “Estimated Efforts” for each day (Day 1 to Day 10). Enter the formula at D15 ($C16 - \$C\$16/10$) and drag it for all the days.
6. For each day, enter the actual efforts. Enter the formula at D17 ($SUM (D5:D15)$) for summing up the actual efforts left, and drag it for all the other days.
7. Select it and create the chart as follows:

12) Velocity

The total number of story point which a scrum team archives in a sprint, is called Velocity. The Scrum team is judged or referenced by its velocity. Having said that, it should be kept in mind that the objective here is NOT achieving the maximum story points, but to have a quality deliverable, respecting the scrum team's comfort level.

For Example: For a particular sprint: the total number of user stories are 8 having story points as shown below.

So here the velocity will be the sum of the story points = 30

Definition of Done:

A Sprint is marked as Done when all the stories are completed, all dev, research, QA tasks are marked 'Completed', all bugs are fixed-closed else the ones that can be done later (like not completely related or are less important) are pulled out and added in the backlog, the code review and unit testing is completed, the estimated hours have met the actual hours put up in the tasks and most importantly a successful demo has been given to the PO and the stakeholders.

Activities Done in SCRUM Methodology

#1) Planning Meeting

A planning meeting is the starting point of Sprint. It is the meeting where the entire scrum team gathers, the SCRUM Master selects a user story based on the priority from the product backlog and the team brainstorms on it.

Based on the discussion, the scrum team decides the complexity of the story and sizes it as per the Fibonacci series. The team identifies the tasks along with the efforts (in hours) which would be done to complete the implementation of the user story.

Many a time, the planning meeting is preceded by a “Pre-Planning meeting”. It’s just like homework which the scrum team does before they sit for the formal planning meet. The team tries to write down the dependencies or other factors which they would like to discuss in the planning meeting.

#2) Execution of Sprint Tasks

As the name suggests, these are the actual work done by the scrum team to accomplish their task and take the user story into the “Done” state.

#3) Daily Standup

During the sprint cycle, every day the scrum team meets for, not more than 15 minutes (could be a stand-up call, recommended to have during the beginning of the day) and state 3 points:

1. What did the team member do yesterday?
2. What did the team member plan to do today?
3. Any impediments (roadblocks)?

It is the Scrum master who facilitates this meeting. In case, any team member is facing any kind of difficulties, the scrum master follows up to get it resolved. In Stand ups, the board is also reviewed and in itself shows the progress of the team.

#4) Review Meeting

At the end of every sprint cycle, the SCRUM team meets again and demonstrates the implemented user stories to the product owner. The product owner may cross verify the stories as per its acceptance criteria. It's again the responsibility of the Scrum master to preside over this meeting.

Also in the SCRUM tool, the Sprint is closed and the tasks are marked done.

#5) Retrospective Meeting

The retrospective meeting happens after the review meeting.

The SCRUM team meets, discusses & document the following points:

- What went well during the Sprint (Best practices)?
- What did not go well in the Sprint?
- Lessons learned
- Action Items.

The Scrum team should continue to follow the best practice, ignore the “not best practices” and implement the lessons learned during the consequent sprints. The retrospective meeting helps to implement the continuous improvement of the SCRUM process.

How the Process is done? An Example!

Having read about the technical jargons of SCRUM. let me try to demonstrate the whole process with an example.

Example:

Step #1: Let's have a SCRUM team of 9 people comprising of 1 product owner, 1 Scrum master, 2 testers, 4 developers and 1 DBA.

Step #2: The Sprint is decided to follow a 4 weeks cycle. So we have 1-month Sprint starting 5th June to 4th of July.

Step #3: The Product Owner has the prioritized list of user stories in the product backlog.

Step #4: The team decides to meet on 4th June for the “Pre Planning” meeting.

- The product owner takes 1 story from the product backlog, describes it and leaves it to the team to brainstorm on it.
- The entire team discusses and communicates directly to the product owner to have clearly understood the user story.
- In a similar way, various other user stories are taken. If possible, the team can go ahead and size the stories as well.

After all the discussion, Individual team members go back to their workstations and

- Identify their individual tasks for each story.
- Calculate the exact number of hours on which they will be working. Let's check how the member concludes these hours.

Total number of working hours = 9

Minus 1 hour for a break, minus 1 hour for meetings, minus 1 hour for emails, discussions, troubleshooting etc.

So the actual working hours = 6.

A total number of working days during the Sprint = 21 days.

Total number of hours available = $21 \times 6 = 126$.

The member is on leave for 2 days = 12 hours (This varies for each member, some may take leave and some may not.)

Number of actual hours = $126 - 12 = 114$ hours.

This means that the member will actually be available for 114 hours for this sprint. So he will break down his individual sprint task in such a way that a total of 114 hours is reached.

Step #5: On the 5th of June the entire Scrum team meets for the “Planning Meeting”.

- The final verdict of the user story from the product backlog is done and the story is moved to the Sprint Backlog.
- For each story, each team member declares their identified tasks, if required they can have a discussion on those tasks, can size or resize it (remember the Fibonacci series!!).
- The Scrum master or the team enter their individual tasks along with their hours for each story in a tool.
- After all the stories are completed, Scrum master notes the initial Velocity and formally starts the Sprint.

Step #6: Once the Sprint has started, based on the tasks assigned, each team member starts working on those tasks.

Step #7: The team meets daily for 15 minutes and discusses 3 things:

- What did they do yesterday?
- What they plan to do today?
- Any impediments (roadblocks)?

Step #8: The scrum master tracks the progress on a daily basis with the help of “Burn down chart”.

Step #9: In case of any impediments, the Scrum master follows up to resolve those.

Step #10: On 4th July, the team meets again for the review meeting. A member demonstrates the implemented user story to the product owner.

Step #11: On 5th July, the Team meets again for the Retrospective, where they discuss

- What went well?
- What did not go well?
- Action Items.

Step #12: On 6th July, the Team again meets for pre-planning meeting for the next sprint and the cycle continues.

SCRUM Activity Tools

There are several tools that can be used extensively for tracking scrum activities.

Some of them include:

- [Jira](#)
- [XPlanner](#)
- [VersionOne](#)
- [Sprintometer](#)
- [ScrumNinja](#)

In the upcoming tutorial, we would be shedding light on the Agile Manifesto which is a notion that drives effective Agile Teams.

Next Tutorial

About the Authors: This series is written by the following STH team members:

Shruti Shrivastava – A Professional Scrum Master with 9 years of experience across BFSI, E-commerce and B2B domains. Shruti is an expert in automation testing and leading scrum teams.

Anshul Kumar Srivastava – A result-oriented Product Management professional and Agile practitioner with 7 years of experience across BFSI and Telecom sectors.

Recommended Reading

- [Agile Scrum Online Quiz: Test Your Knowledge of Agile Scrum](#)
- [Kanban vs Scrum vs Agile: A Detailed Comparison to Find Differences](#)
- [How to Deliver High Value Software Features in a Short Time Period using Agile Scrum Process](#)
- [Agile Manifesto: Understanding Agile Values and Principles](#)
- [SAFe Agile Tutorial: What is Scaled Agile Framework](#)
- [30+ Top Scrum Interview Questions and Answers \[2021 LIST\]](#)
- [Agile Vs Waterfall: Which Is The Best Methodology For Your Project?](#)

- [Top 31 Agile Interview Questions and Answers](#)

- [Agile Testing, Testing Methodologies](#)
 - < [Shift-Left of Quality: How is it Equally Important as Shift-Left in Testing?](#)
 - > [Making API Testing Simple with Katalon Studio](#)

117 thoughts on “Agile Methodology: A Beginner’s Guide To Agile Method and Scrum”

[← Older Comments](#)

Srinivas Bothala

Can PO be part of daily stand up given that stand up is time bond. Additionally, in a 02 weeks sprint, how late is the latest that we can accomodate a change in requirement/user story without impacting teams burndown & velocity numbers ?

[Reply](#)

JS

PO can be part of stand up as only the observer as stand up is time-boxed and everyone has to update on their status, however if developers need any quick clarification they can discuss with PO and request for his/her time for detailed dicussion.

[Reply](#)

Sonali Pawar Bhosale

it is very descriptive and helpful session for the tester.I loved it, i understand each and every concept with short time.thank you so much for the great work.

[Reply](#)

Deepak Kumar

This is really great work done my authors.. The concept very clearly explained with clear examples.
Thanks a lot... Much Appreciated.

[Reply](#)

SOMASHEKHAR TERAGUNTI

This is an excellent tutorial work done by authors , Agile concepts and terms are explained very clearly with supportive examples !. Keep up the good work !.

[Reply](#)

Srikanth D

This is so helpful and truly it's Described like spoon feeding to new comers to understand the scrum end to end process

Thanks for providing this.

Reply

Rajesh

Best tutorial till i ever read on agile process.

Reply

Malathi G

Great tutorial

Reply

Stacy Lawrence

This agile approach ensures enhanced communication with close collaboration, and brings about organizational adoption to changes to deliver value to the business owner. Various agile methodologies can be leveraged by businesses based on the project size and based on the project need. Read more to know more!!

Reply

Stacy Lawrence

Thank you for sharing this informative blog. Came across a lot of insightful information.

Reply

Sachin

HI,

As per my experience, there is also a Pre review meeting before the last day of Sprint where Scrum Team demonstrate the user stories to the PO and based on that feedback given by the PO.The scrum team do it and show them on the next day.

[Reply](#)

Rakshatha

The tutorial is very detailed and helpful to beginners and experienced professionals alike.

[Reply](#)

swati sharma

content is written in very easy language , enjoyed learning it.

[Reply](#)

Dheeraj Gadave

Explained in very detail and clear. Great tutorial.

[Reply](#)

prasad

I really liked the content. we follow the same process mentioned here so its pretty clear in detail.
thanks!!

Reply

lucy

I have been reading explanations of Agile methodology, I think this is the best so far.
thank you so much.

Reply

Venessa Serrao

Thanks for pondering. This is a very clear and informative article. Very well written!

Reply

← Older Comments

Leave a Comment

Name *

Email *

POST COMMENT

[ABOUT US](#) | [CONTACT US](#) | [ADVERTISE](#)

ALL ARTICLES ARE COPYRIGHTED AND CANNOT BE REPRODUCED WITHOUT PERMISSION.

© COPYRIGHT SOFTWARETESTINGHELP 2021 — READ OUR [COPYRIGHT POLICY](#) | [PRIVACY POLICY](#) | [TERMS](#) | [COOKIE POLICY](#) | [AFFILIATE DISCLAIMER](#)