International Journal of Scientific Research & Engineering Trends
Volume 6, Issue 3, May-June-2020, ISSN (Online): 2395-566X

Scrum: An Agile Development Gateway

Garima
CSE, Dronacharya College Of Engineering, Gurgaon

Abstract **– Scrum is an amalgamation of several processes used in Agile Methodology in order to complete the development of software in a well defined time span which is generally faster and shorter. Scrum is a framework that was introduced with the prime objective of delivering new software capability every 2-4 weeks in an organization. Scrum provides a smooth foundation of fundamental principles in order to achieve the development of high quality software faster and towards a well defined goal.**

Keywords**– Amalgamation, Authenticatically, Comprehensive, Framework, Foundation, Grooming, Iteration, Methodology, Paramount, Principles, Requisites, Transmute**

I. INTRODUCTION

Scrum is a major paramount of agile development method, used by renowned industries and others around the world. Scrum exists to Transmute the way people handle and accomplish complex projects, Thereby extending the Scrum framework and Agile principles beyond Software development to the much stabilized world of work. Scrum authentically was corroborated for software development projects, but it works conscientiously for any serpentine and newfangled task or project. With the emergence of agile manifesto on February 11-13, 2001, the position of Scrum alsoemerged in the software industry

II. HISTORY OF THE AGILE MANIFESTO

The Agile Manifesto and the Twelve Principles of Agile Software were the consequences of industry frustration in the 1990s. The enormous time lag between business requirements (the applications and features customers were requesting) and the delivery of technology that answered those needs, led to the cancelling of many projects. Business, requirements, and customer requisites changed during this lag time, and the final product did not meet the then current needs. The software development models of the day, led by the Waterfall model, were not meeting the demand for speed and did not take advantage of just how quickly software could be updated or changed
.
In 2000, a group of seventeen "thought leaders," including Jon Kern, Kent Beck, Ward Cunningham, Arie van Bennekum, and Alistair Cockburn, met first at a resort in Oregon and later, in 2001, at The Lodge at Snowbird ski resort in Utah. It was at the second meeting where the Agile Manifesto and the Twelve Principles were formally written. The Manifesto reads
"We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:
"Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan
"That is, while there is value in the items on
the right, we value the items on the left more."

III.THE FOUR VALUES OF THE AGILE MANIFESTO

The Agile Manifesto is comprised of four foundational values and 12 supporting principles which lead the Agile approach to software development. Each Agile methodology applies the four values in different ways, but all of them rely on them to guide the development and delivery of high-quality, working software.

1. Individuals and Interactions over Processes and Tools
The first value in the Agile Manifesto is "Individuals and interactions over processes and tools." Valuing people more highly than processes or tools is easy to understand because it is the people who respond to business needs and drive the development process. If the process or the tools drive development, the team is less responsive to change and less likely to meet customer needs. Communication is an example of the difference between valuing individuals versus process. In the case of individuals, communication is fluid and happens when a need arises. In the case of process, communication is scheduled and requires specific content.

2. Working Software over Comprehensive Documentation
Historically, enormous amounts of time were spent on documenting the product for development and ultimate delivery. Technical specifications, technical requirements, technical prospectus, interface design documents, test plans, documentation plans, and approvals required for each. The list was extensive and was a cause for the long delays in development. Agile does not eliminate documentation, but it streamlines it in a form that gives the developer what is needed to do the work without

getting bogged down in minutiae. Agile documents requirements as user stories, which are sufficient for a software developer to begin the task of building a new function.
The Agile Manifesto values documentation, but it values working software more.

Customer Collaboration over Contract Negotiation
Negotiation is the period when the customer and the product manager work out the details of a delivery, with points along the way where the details may be renegotiated. Collaboration is a different creature entirely. With development models such as Waterfall, customers negotiate the requirements for the product, often in great detail, prior to any work starting. This meant the customer was involved in the process of development before development began and after it was completed, but not during the process. The Agile Manifesto describes a customer who is engaged and collaborates throughout the development process, making. This makes it far easier for development to meet their needs of the customer. Agile methods may include the customer at intervals for periodic demos, but a project could just as easily have an end-user as a daily part of the team and attending all meetings, ensuring the product meets the business needs of the customer.

Responding to Change Over Following a Plan
With Agile, the shortness of an iteration means priorities can be shifted from iteration to iteration and new features can be added into the next iteration. Agile's view is that changes always improve a project; changes provide additional value.
Perhaps nothing illustrates Agile's positive approach to change better than the concept of Method Tailoring, defined in An Agile Information Systems Development Method in use as: "A process or capability in which human agents determine a system development approach for a specific project situation through responsive changes in, and dynamic interplays between contexts, intentions, and method fragments." Agile methodologies allow the Agile team to modify the process and make it fit the team rather than the other way around.

IV. PRINCIPLES OF AGILE MANIFESTO

The Agile Manifesto lists 12 principles to guide teams on how to execute with agility. These are the principles:
Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
Deliver working software frequently, from a couple of weeks to a couple of months, with preference to the shorter timescale.
Business people and developers must work together daily throughout the project.
Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
Working software is the primary measure of progress.
Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
Continuous attention to technical excellence and good design enhances agility.
Simplicity -- the art of maximizing the amount of work not done -- is essential.
The best architectures, requirements, and designs emerge from self-organizing teams.
At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

V. METHODOLOGIES THAT ARE USED TO IMPLEMENT AGILE

Agile is a framework and there are a number of specific methods within the Agile movement. You can think of these as different flavors of Agile:
Extreme Programming (XP): Also known as XP, Extreme Programming is a type of software development intended to improve quality and responsiveness to evolving customer requirements. The principles of XP include feedback, assuming simplicity, and embracing change.

Feature-driven development (FDD): This iterative and incremental software development process blends industry best practices into one approach. There are five basic activities in FDD: develop overall model, build feature list, plan by feature, design by feature, and build by feature.

Adaptive system development (ASD): Adaptive system development represents the idea that projects should always be in a state of continuous adaptation. ASD has a cycle of three repeating series: speculate, collaborate, and learn.

Dynamic Systems Development Method (DSDM): This Agile project delivery framework is used for developing software and non-IT solutions. It addresses the common failures of IT projects, like going over budget, missing deadlines, and lack of user involvement. The eight principles of DSDM are: focus on the business need, deliver on time, collaborate, never compromise quality, build incrementally from firm foundations, develop iteratively, communicate continuously and clearly, and demonstrate control.

Lean Software Development (LSD): Lean Software Development takes Lean manufacturing and Lean IT

principles and applies them to software development. It can be characterized by seven principles: eliminate waste, amplify learning, decide as late as possible, deliver as fast as possible, empower the team, build integrity in, and see the whole.

Kanban: Kanban, meaning "visual sign" or "card" in Japanese, is a visual framework to implement Agile. It promotes small, continuous changes to your current system. Its principles include: visualize the workflow, limit work in progress, manage and enhance the flow, make policies explicit, and continuously improve.

Crystal Clear: Crystal Clear is part of the Crystal family of methodologies. It can be used with teams of six to eight developers and it focuses on the people, not processes or artifacts. Crystal Clear requires the following: frequent delivery of usable code to users, reflective improvement, and osmotic communication preferably by being co-located.

Scrum: Scrum is one of the most popular ways to implement Agile. It is an iterative software model that follows a set of roles, responsibilities, and meetings that never change. Sprints, usually lasting one to two weeks, allow the team to deliver software on a regular basis.

VI. INTRODUCTION TO SCRUM

With Scrum, an entire project is split into a sequence of iterations called Sprints. Each Sprint is time-boxed for not more than one month and planned well in advance. Planning is completed not according to a set of prescribed tools, but according to the requirements as decided by the Scrum team. As such, a self-organizing and a cross-functional team is the backbone of the Scrum method. In order to ensure maximum cooperation among team members, face-to-face communication is encouraged. Also, the stakeholders and the technical team work in close collaboration, thereby ensuring the delivery of high-quality, working software.

VII. ADVANTAGES OF SCRUM

Scrum is a highly prescriptive framework with specific roles and ceremonies. While it can be a lot to learn, these rules have a lot of advantages. The benefits of Scrum include:
More transparency and project visibility: With daily stand-up meetings, the whole team knows who is doing what, eliminating many misunderstandings and confusion. Issues are identified in advance, allowing the team to resolve them before they get out ofhand.

Increased team accountability: There is no project manager telling the Scrum Team what to do and when. Instead, the team collectively decides what work they can complete in each sprint. They all work together and help each other, improving collaboration and empowering each team member to be independent.

Easy to accommodate changes: With short sprints and constant feedback, it's easier to cope with and accommodate changes. For example, if the team discovers a new user story during one sprint, they can easily add that feature to the next sprint during the backlog refinement meeting.

Increased cost savings: Constant communication ensures the team is aware of all issues and changes as soon as they arise, helping to lower expenses and increase quality. By coding and testing features in smaller chunks, there is continuous feedback and mistakes can be corrected early on, before they get too expensive to fix.

VIII. DISADVANTAGES OF SCRUM

While Scrum offers some concrete benefits, it also has some downsides. Scrum requires a high level of experience and commitment from the team and projects can be at risk of scope creep.
Here are the disadvantages of Scrum:
Risk of scope creep: Some Scrum projects can experience scope creep due to a lack of specific end date. With no completion date, stakeholders may be tempted to keep requesting additional functionality.

difficult and sprints can take more time than originally estimated
Team requires experience and commitment: With defined roles and responsibilities, the team needs to be familiar with Scrum principles to succeed. Because there are no defined roles in the Scrum Team (everyone does everything), it requires team members with technical experience. The team also needs to commit to the daily Scrum meetings and to stay on the team for the duration of the project.

The wrong Scrum Master can ruin everything: The Scrum Master is very different from a project manager. The Scrum Master does not have authority over the team; he or she needs to trust the team they are managing and never tell them what to do. If the Scrum Master tries to control the team, the project will fail.

Poorly defined tasks can lead to inaccuracies: Project costs and timelines won't be accurate if tasks are not well defined. If the initial goals are unclear, planning becomes difficult

IX. ROLES IN SCRUM

The governing team of scrum in any organization consists of the following:
Product Owner: The Scrum Product Owner has the vision of what he or she wants to build and conveys that vision

International Journal of Scientific Research & Engineering Trends
Volume 6, Issue 3, May-June-2020, ISSN (Online): 2395-566X

to the team. The Product Owner focuses on business and market requirements, prioritizing all the work that needs to be done. He or she builds and manages the backlog, provides guidance on which features to ship next, and interacts with the team and other stakeholders to make sure everyone understands the items in the product backlog. The Product Owner is not a project manager. Instead of managing the status and progress, his or her job is to motivate the team with a goal and vision.

Scrum Master: Often considered the coach for the team, the Scrum Master helps the team do their best possible work. This means organizing meetings, dealing with roadblocks and challenges, and working with the Product Owner to ensure the product backlog is ready for the next sprint. The Scrum Master also makes sure the team follows the Scrum process. He or she doesn't have authority over the team members, but he or she does have authority over the process. For example, the Scrum Master can't tell someone what to do, but could propose a newsprint cadence.

Scrum Team: The Scrum Team is comprised of five to seven members. Everyone on the project works together, helps each other, and shares a deep sense of camaraderie. Unlike traditional development teams, there are not distinct roles like programmer, designer, or tester. Everyone completes the set of work together. The Scrum Team owns the plan for each sprint; they anticipate how much work they can complete in each iteration.

X. SPRINT PLANNING MEETING

The Sprint Planning Meeting is the primarily a detailed examination held by a Scrum team with the goal of concurring which task shall be executed during a set sprint period. In the preparation of the Sprint Planning Meeting the SCRUM Master surrounds the team with the following confections and Discussion blocks:

1. Product Backlog
2. Sprint Backlog
3. Burn-down Chart

The Sprint Planning Meeting committee consists of the Product Owner (voice of the customer), Scrum Master and the Development Team. This team discussion is congregated to concoct the execution of user stories over the current Sprint and is held in co-located facilities.

In this meeting, the product owner prepares product backlog items to fit known team's sprint velocity and is concerned in communicating the sprint goal that will result in a shippable product.

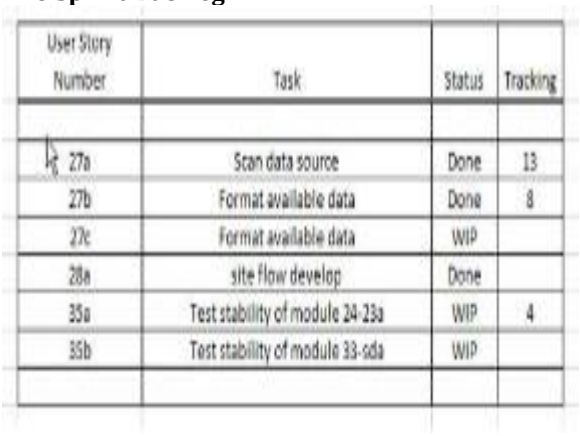
The meeting is devoted to defining the sprint goal which together with the object definition – a Q & A period where the PO details his priorities, the team decomposes user stories from the Product Backlog and devotes time to estimation –where tasks are defined according to time/risk/complexity. Upon agreement a number of these are moved onto the current Sprint Backlog that the team will volunteer to work on and revisit during the sprint.

The Product Backlog

Fig.1. The Product backlog.

In the example above we have taken a snapshot of a Product backlog and its initial stages of decomposition. Please note that some of the entries were introduced not by the PO but by members of the development team as items found during grooming.

The Sprint Backlog



User Story Number	Task	Status	Tracking
27a	Scan data source	Done	13
27b	Format available data	Done	8
27c	Format available data	WIP	
28a	site flow develop	Done	
35a	Test stability of module 24-23a	WIP	4
35b	Test stability of module 33-sda	WIP	

Fig.2. The Sprint Backlog.

An output of the Sprint Review Meeting, the Sprint Backlog is shown above. There can be many varieties of what is listed but for the most part it identifies the User Story from where the task originated the description of the task, the status and the estimate value. The estimate is the measure of the task relative to the velocity and the team accomplishment value.

International Journal of Scientific Research & Engineering Trends
Volume 6, Issue 3, May-June-2020, ISSN (Online): 2395-566X

The Burn-down Chart

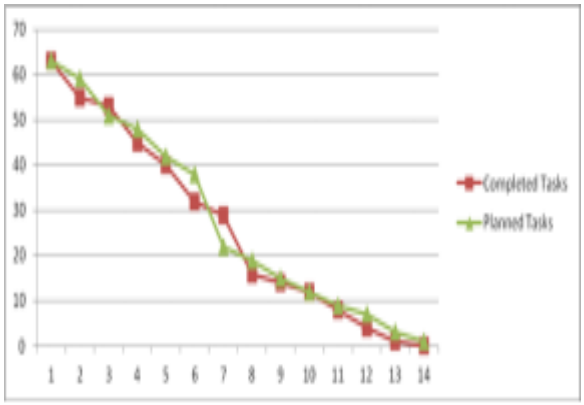


Fig.3. The Burndown Chart.

One of the best sprint status reporting artifacts, the Burn-down Chart is used to assess the success of the sprint remaining days relative to the target velocity. The chart is updated towards the end of the sprint day by the team deducting the amount of completed work from the sprint backlog. Unfinished tasks are moved back to the product backlog and may be prioritized on the next sprint iteration.

XI. SCRUM ARTIFACTS

Product Backlog

The product backlog is the single most important document that outlines every requirement for a system, project or product. The product backlog can be thought of as a to-do list consisting of work items, each of which produces a deliverable with business value. Backlog items are ordered in terms of business value by the Product Owner.

Sprint Backlog

A sprint backlog is the specific list of items taken from the product backlog which are to be completed in a sprint.

Increment

An Increment is the sum of all product backlog items that have been completed since the last software release. While it is up to the Product Owner to decide on when an increment is released, it is the team's responsibility to make sure everything that is included in an increment is ready to be released. This is also referred to as the Potentially Shippable Increment (PSI).

XII. SCRUM VERSUS EXTREME PROGRAMMING

Scrum and Extreme Programming are the two main models of Agile Framework

Scrum Framework:

Scrum is the type of Agile framework. It is a framework within which people can address complex adaptive problem while productivity and creativity of delivering product is at highest possible values. Scrum favors Iterative process and incremental approach.

Extreme Programming (XP):

Extreme Programming is also one of the most important models of Agile framework. This model emphasizes team-work and customer satisfaction as well. The five basic component of Extreme Programming are:

1.Communication
2.Simplicity
3.Feedback
4.Respect
5.Courage

Parameters	Scrum	XP
BUSINESS REPRESENTATIVE TO PROJECT	Product Owner	Customer
STATUS MEETING	Daily Scrum	Daily standup
PROCESS	Iterative	Iterative
PHASES	Product Backlog Sprints Scrum Meeting Demo	Planning Designing Coding Testing
EMPHASIS	Self Organization	Engineering Practices

Fig.4. Comparative Analysis of Scrum And Extreme Programming.

XIII. FUTURE CHALLENGES TO SCRUM

1. The Power of the Possible Product

From having implemented Scrum for the 'how' of product development, adding more focus now to 'what' needs to be built is crucial. That shift will help organizations discover the power of the possible product, reduce the amount of product built, instead of merely optimizing the way that the product is being developed.

2. Upstream Adoption

More than about process and techniques, moving from the old, industrial paradigm to the new Agile paradigm is about culture and behavior. The common bottom-up enthusiasm that arises from doing Scrum is unlikely to be sufficient for such a transformation. For a lasting effect, the common bottom-up enthusiasm needs to be supported and facilitated by upstream adoption.

Besides these ones, there are obviously many more challenges related to implementing and adopting Scrum, related to getting more benefits, a higher agility out of Scrum, a higher ability to adapt:

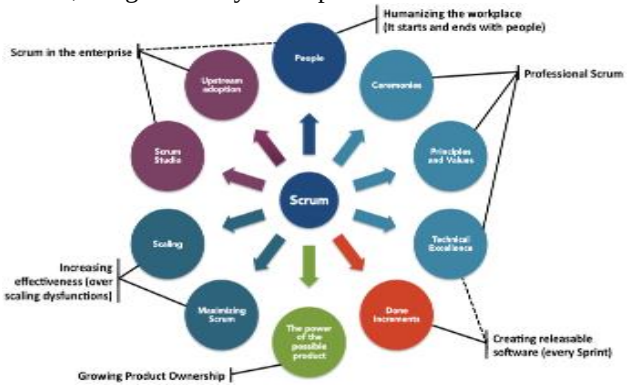


Fig.5. The Scrum.

Let alone the secondary-order challenges that many organizations get themselves into with attempts to re-define, wrap, package, and rename Scrum, although the core engine they build upon is still… Scrum.

XIV. CONCLUSION

When one is working on a project where the scope is changing rapidly and new requirements are emerging every other day, Scrum's iterative and incremental model of development permits modifications to be made to the project system rapidly and responsively.

When there is no consensus on a particular project management approach, the team can adopt Scrum, because its combination of some of the best and most pliable project management principles will help everyone be comfortable.
When there is marked ambiguity and indecisiveness on how to get software or a product developed per industry requirements, the increased productivity and decreased risk rate that comes with Scrum makes it the logical choice.

REFERENCES

[1]. https://www.academia.edu/43057679/Cost_Efficient_Scrum_Process_Methodology_to_Improve_Agile_Software_Development
[2]. https://www.sciencedirect.com/science/article/abs/pii/S0363811114001805
[3]. https://resources.collab.net/agile-101/what-is-scrum
[4]. https://www.scrumalliance.org/ScrumRedesignDEVSite/media/ScrumAllianceMedia/Public%20Relations_2/What-is-Scrum-Backgrounder-2014.pdf
[5]. https://www.tutorialspoint.com/extreme_programming/scrum_plus_extreme_programming.htm
[6]. https://medium.com/serious-scrum/my-response-to-the-talk-modern-alternatives-to-scrum-78b71d796c59
[7]. http://www.ijetmas.com/admin/resources/project/paper/f201411181416300558.pdf
[8]. Scrum – A Pocket Guide, November 2013