Scrum Adoption in our Experience: TGCFO Approach

Vadiraj R Thayur

Vadiraj_Thayur@McAfee.com

Abstract

Agile is a buzzword in the software industry and most companies are investing a huge amount in adopting Agile practices. Scrum, being one of the most prevalent agile methodologies is not very easy to adopt. There is no single way of adopting Scrum and that is why it is important to also learn from the experiences of teams who have adopted Scrum successfully.

Scrum is very flexible and this makes adoption tough and confusing. There are a huge percentage of teams worldwide who have been unsuccessful in adopting scrum, resulting in undesirable outcomes. An approach which works for one team might not work for another, even within the same company. If the project involves geographically distributed teams, the challenge gets tougher. So, adoption should be a mix of recommended practices and some real-world learnings.

The basic intent of this paper is to share the scrum adoption experience of a Distributed Agile team in McAfee. This approach also conforms to the Scaled Agile Framework adopted by McAfee for Distributed Projects. In McAfee, as part of the Business Platform Services (BPS) team, we adopted Scrum.

Because there is no best way for adopting Scrum, along with basic scrum principles, experiences of teams also come in handy. This paper shows the actual steps the team followed to successfully adopt scrum and its results. It could be a good case study for teams planning to adopt scrum in their projects.

Biography

Vadiraj Thayur is a Software QA Manager at McAfee, currently working in the McAfee India Center in Bangalore. He has been working for the past 10+ years in different QA roles on Enterprise as well as SaaS products. He is a Certified Scrum Master and has played the role of a Scrum Master for more than a year.

Vadiraj is a Bachelor of Engineering in Information Science from VTU, Karnataka, India. He also holds an M.S. in Quality Management from BITS Pilani, India.

1. Introduction

A long time ago when the software industry was in its **infancy**, projects were limited and market requirements were very much stable. There used to be market trends which were relevant for long period of time. Companies had adequate time to build products to suit the market requirements. They could afford to have projects running for several quarters and in many cases a few years. Another factor helping them was the lack of competition. This was the age of the **Waterfall Model** and its variants. Though there were other models existing at that time, Waterfall was the most widely used.

There were people **collecting requirements** from the field, and feed them into another team which **analyzed** them. They would then forward a set of streamlined requirements to a set of architects. The architects would prepare a **design**, first a high level view and then a detailed one. This design would get into the hands of focused team of Engineers. They would put their heads down and **implement** the detailed design to working software. A team of testers would then pick up the software and **test** it, log bugs, verify the fixes given and then certify the software for release. The **documentation** team would prepare product documents. It was then the job of the **Implementation** team to deploy the software into the customer's environment. These were the steps normally followed as part of the Waterfall approach.

As years passed by, the industry matured. Customer's needs became complex. It was the age of new inventions. Trends were no longer relevant for years together. Many players emerged and competition spiced up. Hence, requirements started changing more often. This was when the **iterative model** of software development evolved. Projects were broken down into iterations and at the end of all the iterations; the software was released to the customer. This was the **childhood** of the software industry.

Now it is **teenaged**. It is characterized by very frequent changes in technology. Customers many times have vague requirements which might change often. Trends sometimes last for a few months. If projects take several months to complete, the output might no longer appeal the customer. The competition is hot. Each space is clogged by many players. The more adaptive approach is at advantage. This is when **Agile** took birth. Agile could also be classified as a refined iterative development approach.

The motto of Agile is to **Inspect and Adapt.** Project cycles are broken down into smaller **sprints.** All stages of the old fashioned waterfall model are fit into each sprint. Teams are cross-functional; they have representatives from each area. At the end of each sprint, the software is ready to be delivered to the Product Owner. In some companies, at the end of each sprint, the software is released to Production. In fact, I know of a sustaining team in McAfee who release software to production at the end of each sprint (15 days). Some companies have gone to the optimal level of Agile, delivering builds on a daily basis to be deployed onto the live site.

There were some compelling reasons why McAfee chose to move towards Scrum. The market was becoming more and more dynamic. Competition was getting tougher. The security landscape was also changing very quickly. The PC business was going flat, and new devices were coming up in quick succession. Security threats were also coming up for the new devices. So, McAfee had to get new products to the market faster, they had to deliver features in increments and do that consistently. Whenever something new came up, their priorities had to change to keep up with the trend. Scrum was a perfect fit to handle this situation.

2. Problem Statement

The principles of Agile are very close to a person's life. In both, changes are constant, the future is unpredictable. Competition is fierce. Both are likely to fail at times, but the key is to learn from failures and move ahead. The key is again **Inspect and Adapt.**

More and more companies are embracing agile. It has become a buzzword in the industry.

If Agile is a Software Development Model, following are some popular Agile Development methodologies:

i. Scrum

ii. Lean and Kanban

iii. Extreme Programming (XP)

iv. Crystal

v. Dynamic Systems Development Model (DSDM)

vi. Feature-Driven Development (FDD)

Scrum has emerged as a popular approach among these, mainly due to its simplicity, improved productivity, and ability to act as a wrapper for various engineering practices promoted by other agile methodologies. The following references provide some insight into the popularity of Scrum:

- https://www.techwell.com/2013/02/why-scrum-so-popular
- http://scrummethodology.com/has-scrum-become-the-face-of-agile/

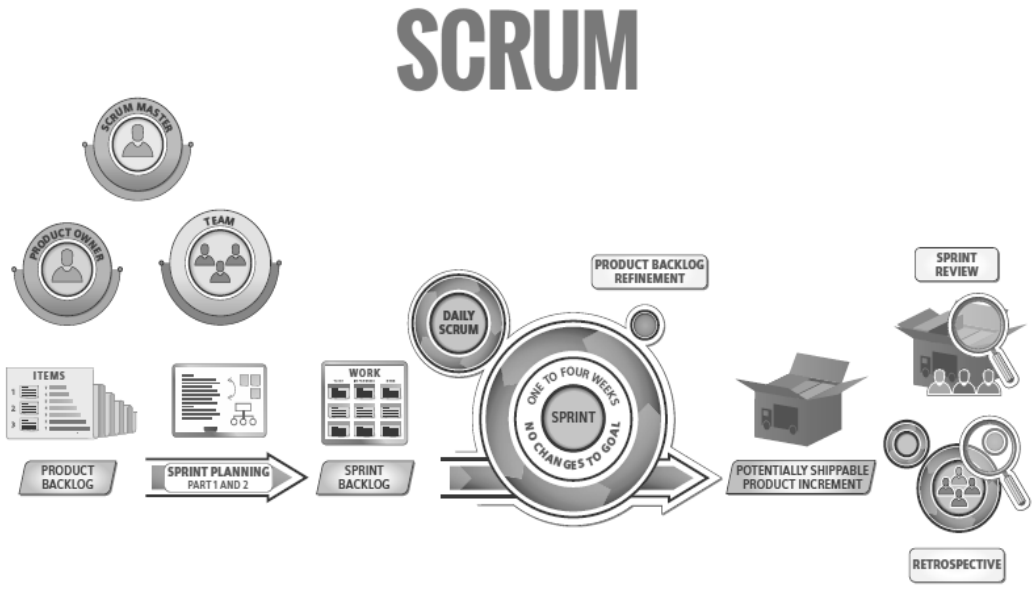

Fig 2.1: Scrum Overview (Source: Scrum Primer, A Lightweight Guide to the Theory and Practice of Scrum. Authors: Pete Deemer, Gabrielle Benefield, Craig Larman and Bas Vodde)

In Scrum, the "Product Owner" works closely with the team to identify and prioritize system functionality in form of a "Product Backlog". The Product Backlog consists of features, bug fixes, non-functional requirements, etc. - whatever needs to be done in order to successfully deliver a working software system. With priorities driven by the Product Owner, cross-functional teams estimate and sign-up to deliver "potentially shippable increments" of software during successive Sprints, typically lasting 30 days. Once a Sprint's Product Backlog is committed, no additional functionality can be added to the Sprint except by the team. After a Sprint has been delivered, the Product Backlog is analyzed and reprioritized, if necessary, and the next set of functionality is selected for the next Sprint.

Many companies are adopting Scrum for their teams. They feel they can deliver more quickly and also better accommodate frequently changing requirements.

Fig 2.2: Agile Adoption Statistics (Source: http://www.onedesk.com/2013/05/agile-adoption-statistics-2012/)

The adoption of scrum has not been easy. The process appears to be simple but it is just a framework to build upon. It is not a panacea. Each team will have to customize the approach based on their requirements. Scrum just points at the problems but the resolution is still left to the implementing team. This acts as an advantage and disadvantage. It makes the process flexible but, it is also one of the main reasons for failure.

In an interview, Ken Schwaber, the iconic leader of the scrum community said “I estimate that 75% of those organizations using Scrum will not succeed in getting the benefits that they hope for from it.” *Source: http://www.netobjectives.com/blogs/scrumbutters-scrumdamentalists-were-mad-hell-and-arent-going-take-it-anymore*

Hence, the approach taken to adopt scrum is quite important. Since there is no prescribed approach, studying and learning from scrum adoption experiences of teams comes in very handy. Hence this paper.

This paper describes the approach followed by the team and highlights some simple recommendations. It includes things which went well and things which went wrong. Some interviews with team members would also be part of this paper. The entire approach was for a team which had two scrum teams in different sites. So, the Distributed Agile concept is also covered in the paper.

3. The Approach

This approach is based on the basic **Scrum Principles** and the **Scaled Agile Framework** adopted by McAfee for Distributed projects. It is called **“TGCFO Approach”** as an enumeration of the 5 phases involved.

The basic Scrum principles are:

- Empirical Process Control
- Self-Organization
- Collaboration
- Value based Prioritization
- Time-boxing
- Iterative Development

(Source: http://www.scrumstudy.com/scrum-principles.asp)

The Scaled Agile Framework adopted by McAfee has some enhancements to the traditional agile framework. It has been designed to support geographically distributed Scrum teams working on a single project. Mainly, a new role of **Product Owner Proxy** has been introduced. This person would be responsible for interacting with the Product Owner and obtain requirements and clarifications. This would eliminate the dependency of the Scrum team on the Product Owner who would usually be in a remote site. **Scrum of Scrums** is another process which involves Scrum Masters, Product Owner and Product Owner Proxies interacting on a regular basis to discuss progress and issues.

The TGCFO Approach which is based on these two comprises of the following phases:

- **Train**
- **Gel**
- **Coach**
- **Follow**
- **Own**

This approach is based on the success story of one of the enterprise teams in McAfee which adopted Scrum successfully. The team had representation across two sites: India and US. Both teams were working on the same set of requirements. Both teams were new to Scrum and the approach is based on the experience of the India team in their journey of Scrum Adoption.

3.1 Train

To begin with, the **Scrum Team** should be formed before the team attends the training. The Product Owner, Scrum Master and Proxy **roles** should already be identified. This will enable the team and the individuals to visualize their roles throughout the training. They would have a better understanding of their role and it would help them execute better when they actually start implementing Scrum.

Training needs to be done before introducing something new to the team. The team needs to know at least the theoretical concepts of scrum before plunging into it. They need to know the basic scrum principles, scrum ceremonies, scrum artifacts, the members of scrum team, the advantages of scrum and the disadvantages as well.

The main shift that has to happen is with the mindset of people. In Scrum, the roles of Manager, Lead etc. diminish. It is the team which is fully empowered and responsible for success and failure. So, the shift from a hierarchical mindset to a horizontal structure mindset is not that easy. Here, training plays

a very important role in helping the team understand their roles and responsibilities, understand the process and also helps the management understand their roles.

It is recommended that the entire scrum team (rather the future scrum team) attend the **training** together. In this way, they get to know each other (if they don't know already) and they could experience the entire training together. It would serve as a great kick-start to their scrum journey. After all, it's people who make processes successful.

Choice of the **Trainer** is equally important. The trainer should be a **Certified Agile Coach** and a **Scrum Practitioner.** It would be great if the trainer could continue with the team as a coach in their learning phase. In that way, the trainer can practically demonstrate what was covered in the training sessions.

The content of the training would be equally important. It is recommended that **training content** be reviewed with concerned people in the company so that they can ensure that the training would also cover aspects which they might want to emphasize. The training should not be a mere presentation. It should involve live exercises for the team at every step, like a mock Release Planning, a mock Sprint Planning, demonstration of a Daily Standup; Exercises on how to write Stories and Tasks, Estimation exercise and so on. This would keep the training interesting and would also give the team a first shot at all the different aspects of Scrum.

3.1.1 In Our Experience

The market was becoming more and more dynamic. Competition was getting tougher. The security landscape was also changing very quickly. PC business was going flat; new devices were coming up in quick succession. Security threats were also coming up for the new devices. So, McAfee had to get new products to the market earlier, they had to deliver features in increments and do that consistently. Whenever something new came up, their priorities had to change to keep up with the trend. Scrum was a perfect fit to handle this situation.

When McAfee started investing in Scrum, the goal was to slowly move the company towards Agile Development. Our team members already had experience with Quarterly Delivery in the previous product. So, when work began on the new product (Business Platform Services), they were the natural choice for experimenting Scrum in our Business Unit.

The team was divided into multiple teams, each team working on a specific part of the product. There was another part of the team in United States. A **Scrum Master** was identified for each team and since the **Product Owner** (in our case, the Product Manager) was in a remote location (United States) and a **Product Owner Proxy** was identified.

A **Scrum Coach** was identified by our company for each Engineering site. Each of them were experienced Agile Practitioners with multiple years' experience. The coach from US was flown in to do the training. It was two day offsite training.

Before the training, Scrum Masters had a session with the Product Owner Proxy and they identified some real epics and stories to carry into the training. They also had discussions to come up with some challenges / questions that they could think of so that they could discuss during the training.

The agenda of the training was a mix of concepts and simulation. The first day was mostly on **concepts**: What is Scrum; The approach; Roles and Responsibilities; Scrum Ceremonies; What is Epic, Story, Task and so on. This was followed by some **practical simulation** which continued onto the second day. Some activities were like Breaking down an Epic into Stories, Story into Tasks, a 30-minute Release Planning; a 30-min Sprint planning and so on. Then,

the art of estimating using Story points was introduced. This was followed by a game of **Planning Poker.** And finally, the training ended with an extensive Q&A session.

It was a great fun-filled experience for the team. However, there was quite a bit of confusion in their minds. It was a totally new process and it required a totally different mindset. Everyone had apprehensions.

3.1.2 Comments

Here are some interesting comments from McAfee team members about the training:

Shyamsunder said “Scrum training was quite useful. The training helped in many doubts on scrum, agile getting clarified. At the end of two day training everyone not only got to know of various terminology but most of methodology, it’s advantageous over waterfall. It provided the solid foundation and platform to embrace scrum. It helped in getting clarity on roles, responsibility of scrum members.”

Shruthi said “Scrum training was really useful as we got a chance to understand agile methodology very well. It was very interactive, knowledge imparting, fun filled training. The training has acted as a strong foundation for the Scrum process being followed.”

3.2 Gel

This is a very important phase. It has nothing to do with the Scrum Methodology but this is the key to success. Scrum methodology relies heavily on people and their mindset. Scrum lays a lot of emphasis on collective thinking and decision making. The entire team is responsible for any success or failure. A cohesive team can work wonders whereas a loosely-knit team can cause disasters. So, it is very important to ensure the team remains together.

One key step towards achieving this is to **co-locate** the team. Co-locating the team will enable frequent verbal communication. Verbal communication is by far the best way of communicating as it is real-time and either parties get instant feedback. Verbal communication often helps resolve issues which might take more time and effort otherwise. Co-location also enables collective thinking which is obviously better than individual thinking. Another advantage of co-location is that entire team is always aware of what is happening and they are always in sync. This reduces the need for many discussions and eliminates many conflicts.

All members of the team should be seated together. In many companies, all of them are seated in a single room. In some companies, they are seated in a circle. In companies where there is a cubicle structure, they are located in adjacent cubicles.

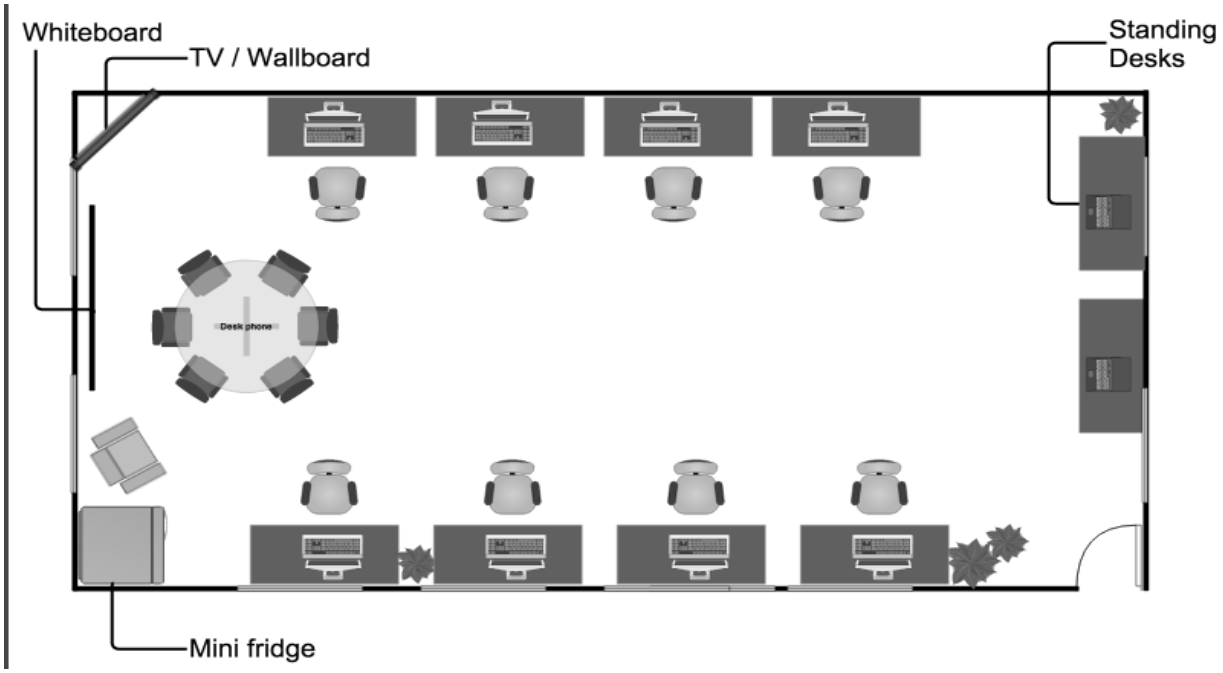

Fig 3.1: A sample seating arrangement for a Scrum Team (Source: http://www.velocitycounts.com/2014/01/building-best-agile-team-workspace/)

Seating the team together can facilitate lot of **informal discussions** and help build **strong relationship** between the team members. Initially, there might be conflicts but gradually, the team will start to learn to work together. This is very important in building a successful scrum team. This would also enable the team to participate in various scrum ceremonies.

Another way of ensuring the team **gels** well is to organize team building activities and outings. These would help people understand each other and that would go a long way in ensuring greater collaboration within the team.

3.2.1 In our experience

Prior to Scrum adoption, our team had been seated separately, Development on one floor and QA on the other. Bringing the entire team to a common location was tough considering space constraints. But the management put in a lot of effort to seat us all together.

There were lot of apprehensions initially about seating Development and QA together and there was fear of constant conflict. But, the experimentation began. Initially, there was not much communication; people were still moving along with their old team mates. But, as scrum ceremonies started and the project gathered steam, they started interacting more.

The other thing that helped was team activities. In the past, the teams went separately for team outings. Now, they were made to go out together, be it a team outing or some team celebration or lunch. This helped build some bonding between team members and it also helped them communicate better.

3.2.2 Comments

Some comments from McAfee team members about team co-location:

Shruthi said “Positives: Lots of time is saved. Now we don’t have to send out mails and wait for responses, we can just walk down to scrum team member’s desk and get doubts cleared. Standups are consuming lesser time as we can gather quickly. Negatives: Sometimes we (both Dev and QA) end up consuming each other’s time when it is really not required. “

Asha said “In Beginning when the team was co-located I felt uncomfortable, felt like QA cannot share openly to DEV. But later on felt it was good sitting together as could get clarifications without any delay from DEV and vice versa.”

3.3 Coach

Creating the Scrum environment, training and ensuring team collaboration is the first step. The next step would naturally be to start following Scrum practices. But, it is easier said than done. The team would have to change their mindset and that would take time. So, it would be apt to have the Agile Coach to stay with the team for some time.

The teams would be accustomed to a different way of thinking before adopting Scrum. They would follow the **Big Bang Approach (Some explanation on Big Bang Approach at: http://www.slideshare.net/zumbara2009/big-bang-or-bit-by-bitdoc)** where given a set of requirements, they would sit together for days and brainstorm and plan how to implement the requirements. They would come up with the detailed design upfront and start planning the development effort. They would have fixed a release date and they would start deducing milestones moving backwards.

With Scrum, the mindset has to change. Everyone has to start thinking in a **bit-by-bit** fashion. They should do a very high-level design of the solution using the requirements. Then, they should prioritize the requirements and decide on what to work in the first sprint. Detailed design of the first sprint requirements should be planned and accommodated in the first sprint along with development based on the design. On completion of the first sprint, they should start thinking about requirements of the next sprint and so on. This requires the team to be able to logically divide the existing requirements and prioritize among them and decide on the implementation plan. They should start working with a mindset of going in deep for only the upcoming sprint’s requirements and think of the other requirements when the corresponding sprint is coming up. The mindset should be to ship a Potentially Shippable Increment to the customer (if not really ship, the Quality should be at that level).To adopt this iterative thinking approach, the team needs to have a good idea of the overall solution and its requirements. They should be able to discuss efficiently and come up with a detailed execution plan.

This change cannot happen without an expert’s constant guidance and feedback. Here, the presence of an **Agile Coach** becomes crucial. The Agile Coach should be actively involved with the team, at least in these initial few sprints. The coach should be part of all scrum ceremonies in this period. The Coach should handhold the team through each ceremony in the initial sprints and later, continue to be actively involved after few initial sprints. This serves one main purpose: The coach can observe the way the team is adopting the methodology and give early and effective feedback. This helps in correcting their mistakes in the learning phase itself so that they do not end up adopting a **“Scum-but”** methodology. In communicating the feedback, the coach should be careful not to over communicate / under communicate. At times, this could de-motivate the team. In this aspect, the Scrum Master can lend a hand by helping the coach communicate effectively to the team and also work as a proxy-coach and help the team start following the processes. The patience, expertise and people skills of the coach come in handy here.

3.3.1 In our Experience

Once our training was complete, a coach was appointed for our team. He was an external agile practitioner and had many years of experience in Scrum and coaching.

To begin with, he started attending the planning meetings and the daily scrum meetings. For the first few days, he was just a keen observer, talking only when any questions were directed at him. After those initial days, he started giving feedback to **Product Owner Proxy** and **Scrum Master** about what was not going well and how we could improve.

Our earlier daily standup meetings used to be prolonged. While giving updates, some people used to get stuck in the issues they were facing and that would trigger a long discussion. Sometimes, team members who were not part of the discussion would see that as a waste of their time. Another risk was that the focus of the standup would get lost and people might forget to bring up essential points in the standup. The coach suggested that standup should be focused and not last more than a few minutes. The agenda should be to only discuss **"What was done yesterday", "What is planned for today" and "What the impediments are".** All other detailed discussions should be taken up post standup with only relevant people.

Another instance was during the Sprint planning meeting. Since the product was new to the team and there was no product owner or Architect presence locally. The coach observed during the first couple of Sprint Planning meetings that there was lack of technical decision making in the team. As a result, most of the planning meetings ended with quite a lot of ambiguity. The coach suggested to have a local technical point of contact to help things proceed more gracefully.

3.3.2 Comments

Some comments from McAfee team members:

Manish said "Coach gave us good tips to leverage on the scrum practices and its importance. His admonitions during Scrum meetings help us cope with the learning curve quickly."

Shyamsunder said "Coach attended few meetings as an observer, he noted down how team is responding to scrum adoption. He shared observation which helped us to rectify the issues with daily standup, attention to process, streamline story writing, address gaps between Dev and QA and make the process more effective. "

3.4 Follow

This is the actual implementation phase of Scrum. The team would have gone through the training phase and would also have the guidance of a coach. Now, it is time for some action. The key in this phase is to follow the scrum ceremonies and scrum principles religiously. There might be some amount of tweak in the processes to suit the dynamics of the team but the basic essence of Scrum should remain.

There are various ceremonies in Scrum which need to be followed by the team. There might be some sort of resistance within the team, they might question the need for some processes, and there might be some reluctance. But, it's good to stick around for some time to start seeing the results. The very people who felt the processes were meaningless might start seeing the value once they start following them and also understand the intention behind them. The coach and scrum master would come in very handy in achieving this. They need to ensure the team remains focused, motivated and together.

Following are some important ceremonies:

- **Release Planning:** The team runs through the requirements that come from the **Product Owner**. They prioritize those requirements and also look at the dependencies with other requirements. They also do a high-level capacity planning analyzing the timelines and resource availability. Based on this data,

they fit the requirements into sprint buckets, just to give a rough picture of what they can achieve within the release. Then, they identify the Risks, Dependencies and Assumptions and design mitigation plan for each. This is followed by planning for the first sprint.

This is the general procedure but some important points need emphasis :

- The Release Planning should ideally begin with the Product Owner briefing the team(s) on the requirements, the customer perspective and priority. This would give the team clarity on what they need to implement and why. This clarity would have a positive effect on their estimates and deliverables. A much better approach could be to have this discussion a couple of days before Release planning so that team members could raise questions and get clarification.

- If the teams are spread across sites, it would be a good practice to have sync-up calls before and after planning sessions. This could be a good starter for the planning sessions. If not the entire team, at least the Scrum masters of the teams and the Product Owner could sync-up.

- The team should also create/update their **Definition of Done** criteria for a story, for a sprint and also for the release. Based on these requirements, they should add stories that would help in achieving the criteria (like Performance Testing, Automation and so on)

- The Scrum Master should follow up on the Assumptions, Risks and Dependencies and mitigate them so that the team does not face any hindrance at the time of execution.

- **Sprint Planning:** This begins with the assumption that the team is already aware of what requirements need to be worked on. The team starts with a detailed capacity planning, considering the planned time off for each member and arriving at a total capacity number. Then, they start picking up each requirement and breakdown into simpler stories if required. Acceptance Criteria is added to each Story and an estimate is made. The stories are then prioritized. Based on the estimate, priority and the team's velocity, the team analyses whether all planned stories can be achieved within the sprint. After adjusting the stories into the sprint bucket, each story is broken down into simple tasks. An effort is added to each task and owner assigned. All this is done keeping the team and individual's capacity in mind.

  Some specifics to be aware of:

  - It is always good to enter a Sprint Planning with a well-groomed set of stories. Requirements should already be broken into Stories, acceptance criteria added and estimated. This would make the Sprint planning activity simpler and more effective.

  - Any ambiguity with respect to the stories for the sprint should be clarified before the Sprint planning.

  - It would be good if the team can discuss each story in some detail, consider the Definition of Done (DoD) criteria and then come up with tasks and effort estimates. This ensures that the team has thought through all aspects around implementation of the story and as a result, the story estimation becomes more accurate. They can breakdown the story into more meaningful and granular tasks and ensure that the DoD

criteria is met. If the team does not discuss in detail, they might end up under-estimating or over-estimating the story and might also miss some important tasks. This might result in incomplete implementation of stories and might also affect milestones as stories might get carried over each sprint.

- In case of distributed teams, it would be good to have sync-up calls before and after the planning meetings. At least the Scrum Masters and Product Owners should attend.

- Ensure the stories and tasks are granular so that they can be tracked easily.

- **Daily Standup:** The basic idea of this is to have all team members at the same level of understanding and also to accelerate identification and resolution of issues. The team meets every day at a pre-defined time, each member updates on **What was done, What would be done** and **Impediments.** They are also expected to update their status/efforts (in the tracking tool if they use one).

  Some tips :

  - Standup should be quick. People should not deviate from the 3-point agenda and should be very concise. Standup should not drag on and turn into some other discussion.

  - The basic intention of each team member should be to update the full team and not just the Scrum Master.

  - Any impediments brought up by the team members should be noted by the Scrum Master and should be worked upon.

  - Any points that might need further discussion could be taken up in a separate discussion post-standup. Only relevant people need to be included.

- **Backlog Grooming:** This is very important. During the sprint, the team needs to meet at least once a week and groom the stories planned for the future sprints. They should breakdown the story into simpler stories if required add acceptance criteria and an estimate. This would help the team get clarifications on questions and would ensure that there is no ambiguity at the time of sprint planning.

  Some key things to keep in mind:

  - Team should ensure grooming happens on a regular basis

  - Whatever question comes up during the grooming, Scrum Master should follow up on those and get clarity.

  - It would be good to also identify Risks and Dependencies during grooming so that they could be mitigated before those stories are taken up in future sprints.

- **Sprint Review:** The team presents their work to the wider audience at the end of the sprint. They seek feedback from Product Owner and other key stake holders on the completeness of the story. If the Product Owner and Stakeholders feel that all the required criteria are met, the story could be closed. If not, the story could either move into the team's backlog or a new story could be created to

implement the feedback.

Some important points:

- Only Completed stories should be demonstrated.
- Each story demo should begin with an overview of story followed by the acceptance criteria and then the actual demo.
- Scrum Master should take note of the feedback and ensure it gets incorporated in future.
- Incomplete stories should either be moved back into the backlog or should move into the next sprint.

- **Sprint Retrospective:** This is more of an introspection opportunity for the team on completion of a sprint. They usually discuss **What went well, What needs Improvement** and **What are the steps to be taken.** Each team member should list top two things which went well and top two things which need improvement. Then they vote for the top two things to be improved and identify the plan of action.
  The retrospective is a very important step towards continuous improvement. It gives each member a chance to express their concerns and also enables the team to introspect. Not only this, it is also a platform to celebrate and recognize the achievements of team members. People feel valued in the team if they get recognition amongst their peers and leadership for an extraordinary effect or some creative idea.
  Some important points to note:
  - A Scrum master should take care of tracking the steps identified for improvement.
  - The items identified in the previous retrospective should be reviewed and the progress analyzed.
  - Retrospective should not just be a complaining session, the team should also celebrate the success they saw during the sprint.
  - In case of distributed teams, sharing the retrospective notes would be important so that the offshore team also gets an opportunity to see them.

The Coach plays an important role in this phase. In the initial phase, the coach would be involved in each ceremony, guiding the team through each ceremony. But as the phase progresses, the coach's involvement should decrease gradually. Towards the end of the phase, the coach's involvement should be minimal.

3.4.1 In our Experience

The team had very interesting experiences during this phase.

To begin with, there was a lot of confusion. They did not know what to do in each ceremony. Though they had some theoretical knowledge from the training, practical implementation was a different ball game. The coach came in very handy at this stage. He gave valuable suggestions at each stage, sometimes in the meeting itself and sometimes to the Scrum Master. The team on the other hand did not show much interest to start with. They complained that there were too many meetings. There was not much focus in the meetings too.

As sprints passed by, with constant guidance from the coach, the team started realizing the importance of the ceremonies. But now, there was another issue. The team was interested in the ceremonies but they did not know how to handle them in a smart way. This resulted in prolonged meetings and wastage of time.

Towards the end of the phase, the team learned the art of smart scrum ceremonies. The daily scrums were short and precise, grooming sessions were regular, sprint planning sessions reduced from a day to a couple of hours and the retrospectives were more open and constructive.

3.4.2 Comments

Some comments from McAfee team members:

Shyamsunder said “Initially we were not sure how team would embrace and work per scrum but to our pleasant surprise everyone responded very well to work as per new methodology. Team(s) started demonstrating ownership and started delivering more stories, more productive. Our opinion changed when we saw some team(s) adhering to the process strictly. One of best thing was everyone agreeing to “Sprint done criteria” and going extra length to achieve it. “

Asha said “We started good, but initially we were little lagging in planning, but slowly we picked up in that. Planning was one of the areas which we picked up slowly”

3.5 Own

This is the final stage of scrum adoption. By this stage, the team would have sufficient knowledge about the entire process and would more or less operate without the assistance of the coach.

With this, the team should start following the scrum processes religiously. There should be no requirement of a coach. The Scrum Master should be in a position to guide the team when they hit any obstacles. The team should also be in a position to think independently and come up with solutions. There might be occasional failures but those would be the stepping stones for their future success.

Another key aspect of this phase would be for the teams to continuously **inspect and adapt**. Each team’s requirements would be different and so, they would need to tweak the processes accordingly. This knowledge would come from the learnings, mainly from retrospectives.

The team in this phase would enjoy their processes, come up with new ideas and resolve issues as they encounter. They should make their own decisions and be accountable.

Key things to remember:

- Ensure the team has a free hand in making decisions.
- The management should remove any fear of failure that the team might have. This comes with encouragement and supporting through failures.
- The team should continually inspect and adapt.

3.5.1 In Our Experience

The team is almost in this phase. They have started enjoying the processes. They voluntarily come up with ideas to improve processes. They think on their own and come up with process improvements when they encounter issues.

The fear of failure has reduced to a great extent, if not eliminated. They are ready to take calculated risks. They take feedback from retrospectives very seriously and act upon them in the future iterations.

There is still lot of scope to improve. The theme here is to **inspect and adapt.**

3.5.2 Comments

Some comments from McAfee team members:

Manish said “Initial phase was a learning exercise, but we learnt and adapted from our mistakes. We as a team and as individuals were transformed to plan and execute and strategize better in the project and scrum activities. The mantra that we still carry along - “Inspect and Adapt”.

Shruthi said “Defects were detected in early phases of development. Discussing design of various components with Dev even before the actual development started has helped the QA get in depth knowledge of the component. Discussing the test case design in Scrum meetings has helped us look into the test cases in various angles, get collective inputs from Dev as well as QA and hence resulted in fool proof testing.

Sometimes, some meetings end up being redundant; I think we have scope for improvement there. “

4. Results

The results were not totally rosy. There were some things that went well whereas there are things that the team needs to improve upon:

- Very important, a set of unknown people came together and within a couple of quarters, they knit together as a single team.
- A group which could not take any decision in the preliminary stages became almost self-reliant in a couple of quarters.
- A team which hated processes transformed into team constantly thinking about process improvements.
- The team which had least predictability in terms of throughput developed into a team with considerably good predictability.
- The team reached a stage where they were able to deliver around an average velocity. There were some sprints where they achieved a couple of points above the average whereas some others were a couple of points below the average. The below chart from Version One tool depicts the trend.

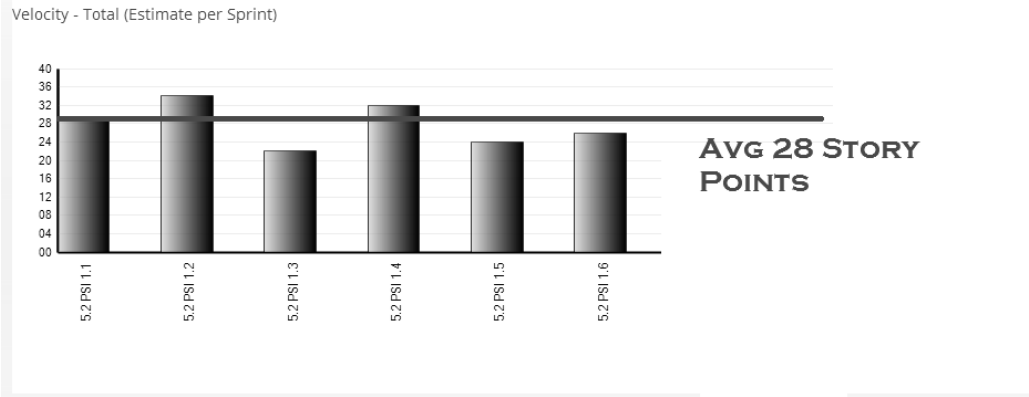

Fig 4.1: Velocity Chart from Version One depicting Velocity across multiple sprints

- The team was able to predict to considerable accuracy what it could achieve within a sprint.
- The defects started decreasing as the team started communicating and collaborating.

  Till the team adopted Scrum, the trend was to get lot of defects. With the adoption of Scrum, the defects reduced tremendously. The reduction could be attributed to the various quality measures like Code Review and Unit Testing which were done as part of the Story Definition-of-Done criteria.

  Following are two defect charts showing the defect count in a project prior to implementation of Scrum and in a project with Scrum. Both projects were executed by the same team and were of similar duration and scope.

		Priority					
		P1	P2	P3	P4	P5	Total
Severity	1	10	2	.	.	.	12
	2	8	74	10	.	.	92
	3	5	49	441	6	.	501
	4	.	22	104	110	9	245
	5	.	1	8	4	31	44
	Total	23	148	563	120	40	894

Fig 4.2: Defect chart for a project prior to Scrum Implementation

		Priority					
		P1	P2	P3	P4	P5	Total
Severity	1	4	1	.	.	.	5
	2	9	30	8	.	.	47
	3	2	15	74	2	.	93
	4	1	3	6	2	.	12
	5	.	.	.	.	1	1
	Total	16	49	88	4	1	158

Fig 4.3: Defect chart for a project with Scrum Implementation

- The management gained confidence on the capability of the team as sprints passed by.
- The Dev-QA divide started diminishing. Development was ready to assist QA team when required and QA team was ready to make improvements to make life easier for the Developers.
- The ceremonies started turning into sessions of learning and enjoyment.
- The team started quite a few quality processes as part of the Definition-of-Done. Those helped to achieve better quality :
    - The team achieved more than 95% code coverage through Unit Tests. Total of more than 7000 unit tests were written.
    - The team could remove more than 60% defects through two-level code review process. More than 1000 defects were filed in Code Collaborator tool including documentation, coding standards and coding / logic issues.

There are some concerns that are yet to be resolved:

- At times, the team feels **stressed.** Immediately as a sprint comes to an end, the next sprint would be in the queue. This sometimes frustrates the team members. They get a feeling that they have no room for relaxation between sprints. They might not enter the next sprint with a fresh relaxed mindset. Eventually, they get stressed.
- There are times when there are hard timelines. The product management once sensed some opportunities in the market to sell the product. There was a Partner launch program with which they wanted to launch the pilot version of the product. Hence, the team had to put in lot of extra effort to meet the timelines. That should not be the case in scrum but there were some business compulsions. This demanded the team to work over weekends.

5. Conclusion

To summarize, scrum adoption is not an easy task. It is a total transformation of the team, in both process as well as mindset. So, it has to be done in a well-planned methodical manner. Here are some key points to remember:

- **Management commitment** and support is required
- **Training** is a must.
- **Scrum Master** is key person. Choose the correct one.
- **Coach** would prove to be the guiding factor. Very essential in the transformation.
- Focus on the **mindset** of people as well. **Team building** is key to the success of any Scrum adoption.
- Some hand-holding is necessary in the initial stages.
- Follow the **ceremonies**. They will start adding value gradually.
- Reduce the intervention as the team matures.

- Allow the team to take **calculated risks**.
- Remove the **fear of failure** from the minds of team members.
- **Trust** the team.

References and Acknowledgements

References

- The Scrum Primer, A Lightweight Guide to the Theory and Practice of Scrum (v2.0)
- Pete Deemer, Gabrielle Benefield, Craig Larman and Bas Vodde
- Various resources on the internet. Exact source mentioned throughout the paper wherever applicable.

Acknowledgements

This paper would not be complete without thanking everyone who has been directly / indirectly involved:

- First of all, the scrum team at McAfee which is the central point of the paper. Without them, this paper would not have been possible.
- All my colleagues for their constant support and guidance.
- Mr. Nagendra Kumar, Quality Director at McAfee whose constant encouragement and guidance helped me complete this paper.
- Mrs. Jayashree, my ex-colleague and a Quality professional for her contribution towards reviewing and formatting the paper.
- Mrs. Jayashree Venugopala, a Lean Six Sigma Certified Marketing professional for her inputs and edits.
- My paper reviewers Leesa Hicks and Sue Bartlett for their valuable feedback.
- McAfee, my company for encouraging my effort and sponsoring my participation in the conference.
- My family for their constant support.