Suresh Gyan Vihar University,
Jaipur International Journal of Converging Technologies and Management (IJCTM)
Volume 1, Issue 2, 2015
ISSN: 2455-7528

Agile Adoption in Tech Startups Our Findings

Sagar Jain
Information Technology Department,
Sardar Patel Institute of Technology
Mumbai, India
sagar7993@gmail.com

Rohit Nair
Information Technology Department,
Sardar Patel Institute of Technology
Mumbai, India
rohitrnair1994@gmail.com

***Abstract*—There has been a rise in the adoption of agile methodology within the software development community. Organizations both large scale and small scale are opting to use agile methodology realizing the benefit it offers over traditional models. However when it comes to a small scale mobile centric tech startup, their needs and goals may be entirely different from large corporations. The research in this paper presents a comparison of existing process models, their advantages, limitations in the context of a small scale mobile centric tech startup and suggests improvements to the rigorous Agile methodology which is currently followed. The findings of this paper are presented in the form of a case study of a similar startup where they have interned and adopted the suggested methodology.**

Keywords—Agile Methodology, Lean Startup, Critical Path, small scale mobile centric tech startup.

I. INTRODUCTION

In recent years, there has been a spike in entrepreneurship. Startups are recognizing the importance of technology to reach out to more customers. In keeping with the importance of mobility in today's world, many startups are focusing on developing applications to support their business, for mobile platforms. The work dynamics and development processes followed in a startup could be different from those in a large corporation, because the needs and objectives of a startup can be vastly different from their ultimate vision. For a startup whose main focus is on technology, following a rigid model such as traditional waterfall model is very difficult.

Many startups are adopting agile methodology for their software development purposes, because it is more adaptable to their constantly changing needs. Agile methodology has been shown to be successful for large companies through various case studies and research papers**[1]**. Different developers may have their own understanding/interpretation of the agile model. In the absence of an experienced agile developer who has a clear understanding of the philosophy behind agile, the entire development process may become disorderly.

For a small scale mobile centric tech startup it may not always be possible to adhere to a rigorous agile model**[2]**. In such a case certain improvements have to be made to mould the development process to suit the needs of the organization.

Agile methodology is defined as one that follows all the Agile principles, for the purposes of this paper.

A startup involving a team of 10 or fewer members - including the business side - focused on developing mobile applications is considered to be a small scale tech startup for this paper. With a rise in the number of startups focusing on providing services via mobile, they are a good focus group to base this research on.

The improvements suggested in this paper is based on the development approach which was actually followed by the authors for a startup. These improvements cater to both the business and technical needs of the startup.

Section II describes the metrics which are commonly used by startups to judge the development effort. Section III describes the agile model and its benefits and limitations. Section IV then describes Lean Startup methodology and its limitations. Section V describes the case study which acts as the basis of the proposed system and the proposed system itself. Section VI is the conclusion, followed by the acknowledgement in section VII.

Abbreviations:
JBGE - Just barely good enough
MVP - Minimal Viable Product
CP - Critical Path
USP – Unique Selling Point

II. METRICS

The metrics usually used to determine the project's success vary from organization to organization. Figure 1 below, shows the iron triangle**[3]** which represents the commonly used metrics and the tradeoffs between each metric which ultimately defines the quality of the project.

Figure 1 – Iron Triangle

The metrics used here are the same as that used by Yau and Murphy**[2]**:

Suresh Gyan Vihar University,
Jaipur International Journal of Converging Technologies and Management (IJCTM)
Volume 1, Issue 2, 2015
ISSN: 2455-7528

Time - The time taken from start of the project to a release of the product or service.
Cost - The total cost incurred by the startup, including cost of hiring engineers, hardware, software costs etc. **Scope** - The number of features and extensions of the product. **Quality** - Quality factors involve both internal (testability, maintainability) as well as external (usability, reliability)

III. AGILE

In order to avoid the pitfalls of the traditional waterfall model, a solution was proposed which came to be known as Agile Model. The agile model acknowledged "the need for an alternative to documentation driven, heavyweight software development processes" **[4]**. According to Cockburn and Highsmith, Agile software development does not necessarily introduce new practices but is the "recognition of people as the primary drivers of project success, coupled with an intense focus on effectiveness and maneuverability" **[5].** The development process in agile starts with the most basic set of deliverables, followed by planning, implementing and testing the next set of features in subsequent iterations thus
increasing the agility of the development team

A. Benefits of Agile(with respect to a startup)

1. Daily meetings can ensure smooth communication between team members and decreases time and development costs.
2. Pre-sprint planning helps the team narrow down their todo list and focus on the immediate goal. In startups,where the final product is not fully defined it is easy for developers to fall into the trap of developing too many features.
3. Encourages test-driven development and suggests that the developers write acceptance test for their code before they implement the features.
4. Quality of software is also improved not only because of the decrease in bugs and faults in the software but also because of improved maintainability of the software.

B. Limitations of Agile

1. Setting a strict sprint for deliverables is difficult since the sprint may be set by someone who is not aware of the challenges which could arise in development.
2. Many a times startups are trying to break into markets with existing players in which case it is necessary for them to be creative in their product to define their USP. This get limited when sprints are considered.
3. Startups with limited resources may not have a certified scrum master to conduct the meetings and to provide a direction in the project. In such a case it becomes difficult to conduct these meetings.
4. A small scale startup has a limited number of developers, a list of features that is probably being changed and refined constantly and a limited amount of time and money. In this scenario it is possible that test cases written for some feature may become obsolete.
5. Extreme Programming which places special emphasis on pair programming takes a lot of resources which may not always be available in a small scale startup.

IV. LEAN STARTUP

Lean Startup, a term coined by Eric Ries**[6][8]**, is a process model that "builds on many previous management and product development ideas, including lean manufacturing, design thinking, customer development, and agile development". The central idea behind lean methodology is to simply eliminate each and every step which is deemed wasteful. This ensures that only those steps/processes are carried out which add value to the development process. Lean startup methodology aims to help a startup develop a minimum viable product (MVP) which can be used to demonstrate the startup's idea during investor meetings.

A. Limitations of Lean Startups

There are certain key problems associated with lean methodology:
1. The focus is on developing a minimum viable product there are chances that the developers may develop features which are ultimately scrapped from the product.
2. Another problem that could occur is that the lean startup methodology is not scalable in case the startup wants to expand its development activities.

V. CASE STUDY

The authors interned at Fixy a startup which provides handyman services to customers. At the time of employment, there were 5 individuals including the founders. As the founders were from a non technical background, it gave the authors a unique experience on dealing with the business side individuals as well. During the first week, the developers followed a combination of agile and lean methodology. However due to the lack of a proper project manager who could guide the interns, the development process proceeded very slowly and in a haphazard manner.

At this time, the lead authors put in the model suggested in this paper. This helped streamline the development process while still providing for adaptability in case requirements have to be changed suddenly. The suggested model is shown below.

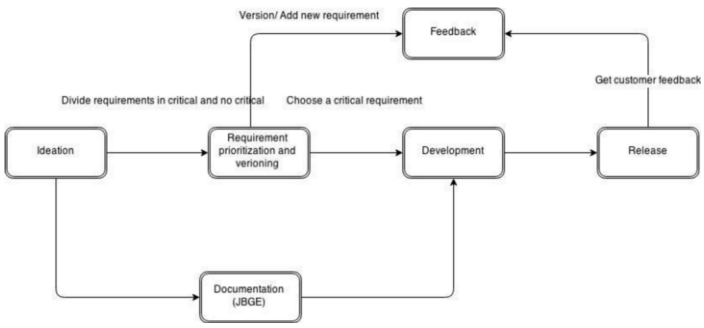

Figure 2 –Flow of Proposed Model

Suresh Gyan Vihar University,
Jaipur International Journal of Converging Technologies and Management (IJCTM)
Volume 1, Issue 2, 2015
ISSN: 2455-7528

A. Ideation

First step of the software development process involves brainstorming sessions and other activities which are designed to generate ideas on what features can be added to the product apart from the main idea of the founders. Ideation is usually carried out by startups before they hire the engineers required; the authors highly recommend that this be carried out with the engineers too, since it is possible that they could bring out new and innovative which would help the startup gain an edge over the existing competitors. The meeting involves all the members of the startup. The activities carried out during this phase are:

1. Mindmap

This step involves the creation of a map consisting of all features which could be implemented in the application/project. Care must be taken to ensure that the suggested features are not impractical or too large for a startup especially one working within the confines of the mobile platforms. The mindmap must be optimized repeatedly and the optimized mindmap obtained should be taken as the basis for the project.

2. Key Processes Areas

A key process area in this model indicates all the requirements which are absolutely essential in the app in question. Identifying the key process areas is an important step in this model. While trying to identify the key process area, one must ask themselves, "Is this a feature for which the user will launch the app?"

3. Wireframe

This stage involves creating wireframe sketches for the application. This provides the developers with a clear view of how the application UI would be like. It is extremely important that this step be carried out since it serves as a prototype UI for initial feedback, and further development.

4. UI/UX

With the advent of design philosophies like Material Design coming into the picture in recent times, UI has gained importance. This is especially the case when it comes to mobile applications where it is of utmost importance to provide the user with an amazing experience along with the desired functionality. It may happen that the developers are not involved in the UI/UX step since it involves working on Photoshop and other designing softwares.

5. Uniformity Constraints

It is necessary to ensure that mobile apps have the same look, feel and provide the same user experience across all major platforms(IOS, Android, Windows). In order to ensure this, it is necessary for developers to discuss the issues which may arise during development and how to tackle them.

6. Review (End of Day)

This step involves reviewing whatever has been discussed in the meeting.

B. Requirement prioritization and versioning

The second phase of this model involves adding the key process areas only, to the job pipeline. In addition to this, the authors also found it helpful to set a threshold for the number of requirements in the pipeline. In case the threshold is violated, then non key process areas can be added to the pipeline.

1. Pipeline

The pipeline as shown below has 4 main stages:

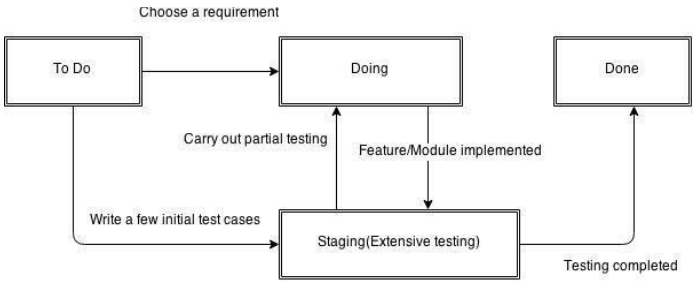

Figure 3 – Pipeline for requirement tracking

1.1. To Do

This is the first stage of the pipeline which describes all the requirements which have yet to be implemented. The requirements can be described either in terms of user stories or technical specifications, as is preferred by the startup. At Fixy, user stories were used to make engineers think of the user's perspective, when developing the application.

1.2. Staging

This stage is where all the testing of the modules are carried out. In a startup, it is impractical to write detailed test cases for each and every requirement. This is because some of these requirements may change based on the interpretation of the stakeholders involved such as the developer, the customer and the business side of the startup. Having a dedicated staging area ensures that developers will think about the test cases that need to be passed after development rather than failing at a later stage. Hence some rudimentary test cases at the beginning, which could be expanded on during the development itself or when the module has been completely implemented.

1.3. Doing

Engineers choose a requirement from the pipeline, and the requirement moves into the doing stage. It is important to note that since only critical requirements are added to the pipeline initially a minimum viable product with all the critical features will be completed as soon as they are implemented.

1.4. Done

Suresh Gyan Vihar University,
Jaipur International Journal of Converging Technologies and Management (IJCTM)
Volume 1, Issue 2, 2015
ISSN: 2455-7528

This is the last stage of a module. Any module which has been completely tested is added to this stage.

2. Requirement Versioning

Another improvement applied to this phase is to ensure that the requirements are versioned instead of being completely changed. Many a times the developers work on a requirement for a long time only to realize that the requirement has changed completely. This means that they have to develop the entire codebase from scratch. In this scenario it is especially helpful if the requirements are simply versioned as much as possible in order to increase reusability. Additionally it boosts the developer's morale.

3. Knowledge Transfer

In order to ensure proper knowledge transfer it is necessary to record the assignees, start date and end date of the requirement, current status, possible over-dues for requirements assigned to a developer.

C. Documentation

1. In-House Documentation

Developers find documentation to be a very cumbersome and useless process. However completely avoiding documentation as allowed in lean startup methodology is not only a bad practice but also leads to major issues for future development. Hence it is necessary to maintain Just Barely Good Enough(JBGE) documentation for in-house development purposes. The definition of JBGE could vary from organization to organization.

2. Outsourcing

Sometimes startups may outsource some of their work to other startups/external developers etc. Here the documentation becomes the primary form of communication between the two organizations. Each one of them will have their own views on the documentation and those views may not be on the same page. At Fixy, the founders were of the firm opinion that all documentation has to be comprehensive and in depth, so that there are no chances of any miscommunication. The authors found that in the long run such a strategy proved to be very effective as it increased accountability among all the parties concerned.

D. Development

This is the phase where the actual coding for implementing the requirements is carried out. Agile methodologies like Scrum place special emphasis on scrum meetings which are to be carried out every day in order to update everyone of what is happening, what needs to be done. These meetings are headed by a scrum master who is a team leader. Scrum meetings allow the development team to "socialize the team members knowledge", and have a deep cultural transcendence**[6]**.
The same concept was initially applied at Fixy where the authors were expected to give an idea of the progress made at the end of the day. However it was observed that some critical features which were difficult to implement took more than a day. This meant that the meetings were not productive and this led to frustration among the developers.
On the suggestion of the authors, the founders agreed to the following guidelines for meetings so as to allow for more development time.

1. Daily meetings (not done on days of progress meeting) The daily meetings are to be carried out twice in the day. However the advantage here is that this meeting is required to be attended by only those who are interested. Other developers can carry unlike in Scrum meetings where it is mandatory that everyone attend.

1.1 Requirements and dependencies

This session is to discuss dependencies or requirements which a developer has. It could be another module that the developer is awaiting or a sub function. In any case, the developers can meet at a pre specified time and discuss their requirements.

1.2 Code Review

This is optional and can be carried out provided there is an urgent need for it and sufficient resources are available. This will help ensure that the developers are not blind to silly logical errors which could be caught by a co-developer.

2.Progress meetings

The progress meetings are carried out once every three days, wherein everyone in the startup meets up and provides updates regarding –

1. Work done

2. Work to be done

3. Urgent issues

This meeting is to keep the founder abreast of the progress in terms of user stories. This meeting is extended when the buffered timeline for one sprint ends. New milestones are set up for the next sprint, along with a new buffered timeline. The authors would like to place special emphasis on the concept of a buffered timeline. Usually agile methodologies define a sprint whose time period could vary between two weeks to a month. However it is possible that some modules which had to be completed in a sprint need an additional day's work to get completed. In this scenario, a buffer time of 2 days would be extremely helpful to the developer.
One other case which needs to be considered when developing an app is dealing with dependencies. In case a developer is working on a module which is completely dependent on another module, then it becomes difficult to manage since one doesn't want either developer to sit idle or feel over pressured. In this case two contingencies could be put in place:

2.1. Critical module

Suresh Gyan Vihar University,
Jaipur International Journal of Converging Technologies and Management (IJCTM)
Volume 1, Issue 2, 2015
ISSN: 2455-7528

Both the developers should try to do pair programming which will ensure that the dependency gets implemented faster thus serving both of them. But it is possible not all developers will be comfortable with pair programming. The option is to choose another critical module which doesn't have any unimplemented dependencies.

2.2. Non-critical module

If the developer is developing a non-critical module, then he/she could simply choose a different task from the pipeline after reverting the earlier requirement to the To Do state.

E. Release

This phase is reached when all the critical modules have been implemented. It is a form of beta release which helps the developers gain an idea of anything that may be wrong or disconcerting to the user.

F. Feedback

The feedback that is received during release of the product is used as a direction for future development. Since all critical path modules have been implemented, a meeting with all the members of the startup is held in order to discuss the way forward in terms of development and to give a direction for the development of non-critical modules. This meeting would also involve a discussion of the feedback received from the users from the release phase.

G. Benefits of this model

There are several benefits of this proposed model.

1. In this model, minimal time is spent on documentation. The documentation written is "Just barely good enough".
2. The daily meetings which are a part of agile, have been cut down ensuring that developers have freedom to continue working while keeping the founders aware of the developments
3. In the ideation meeting everyone can discuss all possible ideas and ensure uniformity for the application.
4. As opposed to a rigorous agile model, the duration of sprints in this model is not iron clad.
5. Versioning requirements helps boost employee morale and ensure greater reusability.
6. The focus of all the engineers is on the key process areas first. This helps in developing MVP as soon as possible 7. The proposed model is scalable as the startup expands.
8. This model ensures proper record-keeping, and eliminates blame-game at a later stage if deadlines are missed in a sprint.
9. It also facilitates a smoother transfer of knowledge between the team members while maintaining transparency.

VI. RESULT

The table below, sheds light on the results obtained by following the prescribed methodology.

Parameter	Expected	Observed
Development Time per sprint	2 weeks	10 days
Cost to develop(Overall)	Rs. 75,000Rs. 10000	Rs. 50,000 – Rs. 75000

On following the prescribed methodology, the founders were able to identify additional resources required at an early stage (during ideation). The identified additional resources were as follows:

Parameter	Expected	Observed
UI Designer	0	3
Database developer	0	1

One of the co-founders, Nabeel Merchant said, "The time taken was a function of the changes and modifications that were required in the Back-End, a key insight learnt only during the development process. The development methodology followed by the interns helped us to understand how a software should be developed."

VII. CONCLUSION

A balance needs to be achieved between quality, time and cost for a small scale startup. In a small scale mobile centric tech startup at Fixy, the authors have observed that it was very difficult to be able to achieve the full advantages of using an agile model due to the paucity of resources, and small team size. The authors have observed how agile fails to provide an affordable solution for all problems faced by a small mobile centric company like Fixy, and have suggested some improvements to the process. It is to be noted here, that this proposed model has been followed by the authors while working at Fixy, and has been found to be a simpler solution than implementing a rigorous agile methodology. The authors would also like to point out, that while this method worked well for Fixy, it may or may not be suitable for other startups. Each small scale startup has its own requirements and goals, and adjustments must be made to this proposed model to suit the companies own needs, before adopting it.

VIII. ACKNOWLEDGEMENT

The authors would like to thank Prof. RupaliSawant for mentoring them, and Prof. RadhaShankarmani for providing her invaluable inputs regarding the working of agile processes. The authors would also like to thank the co-founders of Fixy, Nabeel Merchant and Tufayl Merchant for providing a conducive environment where the authors could provide suggestions to the development process that should be followed in their startup.

X. REFERENCES

[1] P. Krill, "Agile software development is now mainstream," InfoWorld,2010, http://www.infoworld.com/d/developer-world/agilesoftwaredevelopment-now-mainstream-190.
[2] Alex Yau and Christian Murphy's a Rigorous Agile

Methodology the Best Development Strategy for Small Scale Tech Startups?" http://repository.upenn.edu/cis_reports/980/
[3]. A R. Atkinson, "Project Management: Cost, Time And Quality, Two Best Guesses and a Phenomenon, It's Time to Accept Other Success Criteria", International Journal of Project Management, 1999, vol. 17 no.6, pp. 337-42
[4] D. Cohen, M. Lindvall, P. Costa. "An Introduction to Agile Methods." Advances in Computers vol 62, 2004[5] A. Cockburn and J. Highsmith "Agile software development: The business of innovation" Computer, vol 34 no.9 2001. pp.122
[5] E, Ries. "The Lean Startup, How Today's Entrepreneurs UseContinuous Innovation to Create Radically Successful Businesses" Crown Business. 2011. Print http://knowledge.wharton.upenn.edu/article/eric-ries-onthe-lean-startup/
[6]. SCRUM: An extension pattern language for hyperproductive software developmenthttp://jeffsutherland.org/scrum/scrum_plop.pdf