8-2017

Project Manager's Perception of Agile Methods Success

Ankit Sachdeva
Harrisburg University of Science and Technology

Project Manager's Perception of Agile Methods Success

Ankit Sachdeva

Harrisburg University

GRAD 699 2017

Abstract

Since it's inception in 1970, the Waterfall model has been widely accepted and proven to be quite effective in the software technology development. As years have passed, there has been a rise in the number of technology companies and such rapid expansion in the software industry has called for frequent changes and swift delivery. To cope up with the increasing expectations, companies started to adopt new development processes and as a result, businesses witnessed several different process and Agile development methodologies like Scrum, Extreme Programming, Test Driven Development, to name a few. Companies were able to effectively work with change requirements while working on successive iterations. This paper would throw light on the existing software development trends and whether agile methodologies are merely a fad or an absolute necessity for the software industry. This paper is a competitive study of three companies analyzing transition of projects before and after Agile measured through customer acceptance and other indicators. The paper would also drill down on the type of resources and involvement of Agile system demands and whether or not there is a possibility of Agile methodologies been overthrown by any potent successor in the near future. The paper would talk about different styles of leadership techniques adopted in Agile and which type would be best suited for any Agile project. Once singled out the core advantages of Agile, a plan will be created to shift from Waterfall to Agile and learn the core Agile concepts and what should be right way to transform a project into Agile.

Keywords:

Examples – Agile, Software Development, Sprint Backlog, Sprint Planning, Sprint Retrospective, Waterfall

Table of Contents

Introduction

A Software project has to go through a number of stages before a product is obtained. Every software development project starts with requirement gathering that is documented and analyzed. Then, depending on the size of the project, different tools and coding practices are used to obtain the basic software. After a few rounds of thorough testing and fixing the issues, it is ready for the client/ customer.

Different Software's have diverse complexities and every member has to work as a team in order to build a quality product. Hence, a process is of utmost importance. In 1970, Royce first introduced the Waterfall model.

The Waterfall method followed a sequential approach, i.e. only after the completion of one phase the subsequent phases would come into the picture. For waterfall projects, the first step is requirement gathering followed by design architecture, after which the development team would implement various logic and support software components to create an application that resembles the requirements stated in the SRS (Software Requirement Specification). Then this application would be tested and all distinct modules and features would get verified. If any bugs or issues are found they are fixed and finally the product is shipped.

Even though the Waterfall method brought structure to complete the software development process, is was rigid. The problem with the Waterfall approach was a lack of iterations. Due to

the sequential phases, changes couldn't be made after a certain stage. There were many new methodologies that were introduced- like the Spiral model by Boehm in 1988, the Rapid Application Development by Martin in 1991 and the Manifesto for Agile Software Development by several researchers in 2001.

Agile development process evolved a lot from the Waterfall model and emphasized more on team interaction and catering to change requests than following a strict development guideline. It also brought into light the importance of frequent communication with the client. Taking into consideration every point put forward by team members could further perfect agile development process after the completion of every iteration.

Even though the Waterfall method is not completely gone from the market and it is still practiced by a few companies, but its popularity is reducing. Agile on the other hand has been well accepted by different software development companies.

To instill the Agile method in practice businesses can take the help of one or the other prevailing Agile frameworks like Scrum, Kanban, Lean Software Development etc. The distinct edge that these Agile frameworks provide over Waterfall method is an effective solution to handle demanding projects with a high success rate.

The Agile manifesto is based on four key values and twelve principles that make rapidly evolving deliverables possible. Every deliverable or features are divided into iterations and each such iterations are targeted and shipped as small releases. Such practice creates transparency

and instills trust among the client while projecting a clear picture of the software application's future.

Once Agile development was adopted by different software development companies, it was soon realized that such development process not only provides structure and instills discipline within the team, but also helps the client get the best possible version of product with frequent feedback mechanisms.

This paper would outline the distinct advantages brought forward by Agile software development and how this is beneficial as a development practice compared with Waterfall.

Problem Statement and Justification

Every software development project has to follow a process in order to deliver a successful product. Since its inception in 1970, the Waterfall model has helped in providing structure to a number of different projects worldwide. In the current market, where fast pace software delivery is a necessity, most organizations have opted for Agile as the right technique to ensure fast paced delivery. This paper would try to find out the exact reasons why Agile methodology is so successful? It would also emphasize on the distinct impetus provided by Agile to any software development project and whether it's a trend that has caught the attention of the software company or is it absolutely essential. It would also try to find any flaws with Agile method and how likely would any other process overthrow it in the upcoming future.

When the Agile Manifesto was framed in February 2001, it inculcated some elements of the Waterfall model and added a few features of the spiral model to come up with a method that would address the problems of fast delivery. Many teams that still follow a Waterfall model might feel switching to Agile to be an unnecessary overhead, but they are mistaken. Most clients and product owners want to go for short releases. They want to drastically reduce the time when a software application remains in the production pipeline. So if the question is why should Agile be adopted, the answer would be to reduce the time taken by a product to go from a warehouse to the customers, and that too without any major glitches or blockers.

Once a team opts for Agile development, the effectiveness of the process would be quite evident within a few iterations. The team might take some time to adjust with the new development process, but with proper discipline and dedication, the perfect Agile delivery system can be set up within a few sprints. Once the team gets accustomed some of the notable deliverables are:

- Sprint Items - As development take place in multiple iterations (sprints), the team prioritizes the sprint deliverables and assign story points on the basis of complexity to each and every task/ feature. Once all the story points are assigned, the sprint capacity of the complete team is determined. Sprint capacity determines the maximum items that can be delivered by the entire team within the ongoing/ current sprint. Then depending on the priority of the tasks, and the story points provided, the sprint items or deliverables are decided.
- Continuous delivery - Agile manifesto dictates that working software is the primary measure of progress. As Agile processes fall into place, the team would be able to demonstrate their deliverables to the client at the end of each sprint. Once the development of the sprint items is completed and tested, the team can organize a demo and showcase their progress. Each iteration would have a small demo which would contain all the demo-able items that would showcase the progress done by the team during that particular iteration.
- Frequent feedbacks - Prompt feedback is one of the aspects that make Agile different from the Waterfall model. At the end of each sprint, team members

can demonstrate their progress and ask for the client's feedback. If the client recommends any improvement, the request can be added as a new task that can be taken up in the subsequent sprint. The team members themselves can also provide feedback, and the product owner can decide whether the feedback would be beneficial or would in any way add value to the product.

Clients can even propose change requests and get feedback from the team regarding the estimates or the feasibility of its completion within the proposed timeline.

- Sprint retrospective - It is a meeting, which is attended by the team members, project manager and product owner, which is conducted at the end of each sprint. During this meeting, each team member points out how the current sprint went and divide their points into three categories namely- "What went well? ", "What should we stop doing? " and "What we should continue doing? ". All these points from each and every team member are listed so that the overall process and delivery structure can be further refined.

To better assess the advantage and effectiveness of Agile client requirements are studied and then they are asked about how the development team should provide deliverables. Clients who work with Agile teams were asked for feedback and they pointed out that Agile development helps them get a rough structure of the product being built from a very early stage.

They also stated the iterative development structure creates room for sudden changes requests that they can recommend to the development team. After analyzing its priority and impact

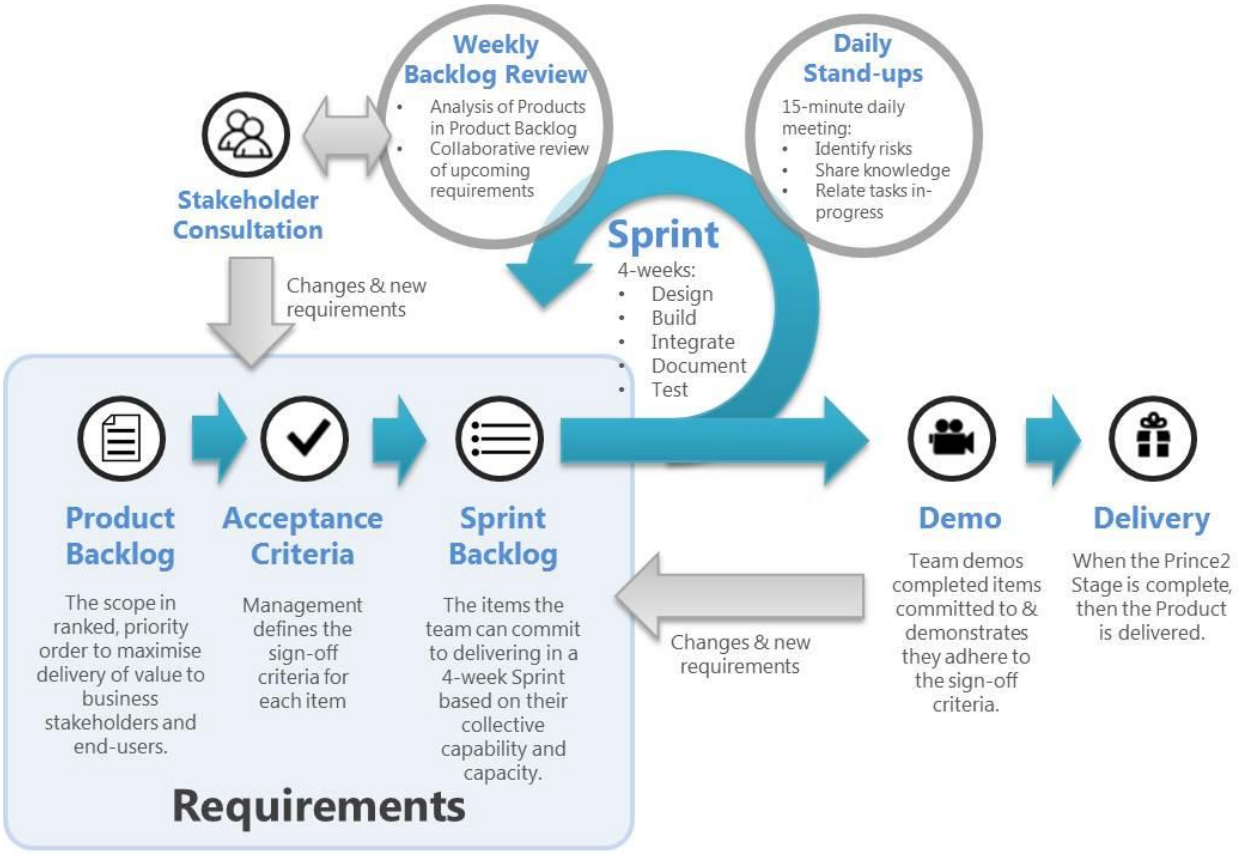
areas on the existing build, they come up with a plan and provide a tentative date for a demo release. Such releases help the clients to monitor the progress and judge whether the product is being developed in the intended manner or not. Also, such early demos are extremely helpful for the clients as they can showcase the development team's progress to the stakeholders or investors and can ensure an early start of ROI.

The paper primarily talks about Agile projects strictly for the software development sector. Even though similar methodology is adopted in other fields, the paper won't dig deep into such details. This paper will not focus on the significance of Waterfall model or how it actually came into picture. Neither would it showcase the advantages it brought forward for static projects, or for projects with fixed requirements.

While talking about effective and efficient Agile, a lot depends on the behavior of the client, professional relation amongst team members and the company. Even though such human factors contribute a lot towards the success of an Agile team, this paper does not include these parameters into consideration.

The iterative nature of Agile ensures fast feedbacks and increases the overall efficiency of the project. It teaches team bonding and instills ownership in each and every team member thus uplifting the overall quality of the project. So considering all the benefits that Agile has to offer, efficient and successful projects can surely be delivered with Agile. And given all its advantages, it is highly unlikely that any other development process would overthrow it in the future. In

order to provide a brief pictorial representation of Agile methodology, let's look at the distinct input provided and what deliverables are to be expected.



This above picture by Zen Ex Machina clearly shows the various steps of Agile development. So the basic Input for Agile development is

- Product/ Sprint Backlog - These backlog consist of the various stories and features that need to be delivered in the current sprint or iteration.

While the deliverables are:

- Product Demo - Demo occurs after the end of each iteration and a working version of the product is obtained at each Demo.
- Release - When significant modules' development is complete, releases are planned.

Literature Review – Analysis of Related Work

The Dynamic Relationships Management Journal took a deep dive into Agile to show how the process changes and the way teams interact and work. It clearly outlined the benefits of the methodology and showed how a process can increase efficiency and impact the working style of any individual in a project.

In Waterfall projects, project managers are solely responsible for creating and allocating tasks, whereas in Agile, the team members show immense amount of self-organizing skill and take pride in owning the product as a whole. They conduct routine standups to keep up with the proceedings of each team member and also perform retrospective meetings at the end of each sprint to ponder on their flaws and mistakes and how the process can be further improved as a whole. The author also pointed out how Agile can be beneficial for the client. Such flexible development process allows the client to introduce or request for change even when the required document and functional specifications have been laid out. Thanks to iterative development, deployments can be done at the end of each sprint which would provide the clients with a partially demo of the product. On the clients' part, they can leverage the demo to convince their stakeholders of the product's ROI early in the development phase.

While in "Iterations in software development processes: A comparison of Agile and Waterfall software development projects", the author's analyzed the interactions and their durations in

each software development phase. They used state transition tables and performed qualitative and quantitative analysis to arrive at a conclusion. They found out that during the design phase, Agile method emphasized on the progression of the iterations while in Waterfall technique, the focus was provided on the efforts taken during each iteration. The Waterfall model took every precaution to ensure a stable design so that they can avoid any unnecessary iteration later in the development stage. This in turn increased the effort and time taken to complete the design phase for a Waterfall project.

In the case of Agile method, the outcome was quite different. Rather than stretching the design phase and delaying the subsequent phases, the Agile method introduced Quality Control later in the development phase so that they can track down any issues and bugs that came up due to flaws in the design. Later, the teams took into account all the bugs present in the system and were able to pinpoint the most vulnerable part of the application. Such brute-force strategy to track down issues by introducing QA team later in the development phase helped them to minimize defects in future versions and also provide a sturdy build in every release, efficiently.

Even though Agile teams are driven and motivated enough to take ownership of the project's deliverables, they do need a manager to assist them in times of releases and handle requirement grooming and delegating tasks amongst team members. But a project manager's approach and vision to manage the team vary from person to person. According to "Leadership of Information Systems Development Projects", leaders can be broadly classified into two groups: Directive Leaders and Empowering Leaders.

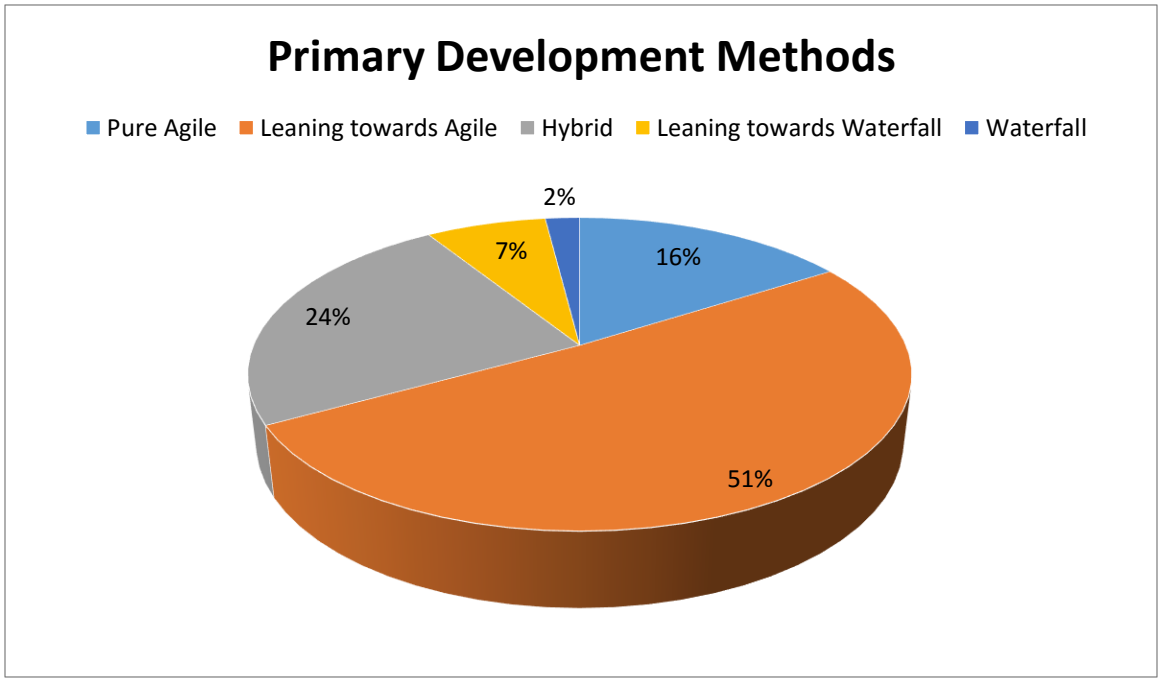
Directive Leaders make decisions, provide individual input and manage team boundaries. While, Empowering Leaders consult team members and take major decisions as a team. The study suggested Empowering Leadership could do wonders when the team consists of highly experienced individuals while Directive Leadership is required for a fresh development team. However, as the level of expertise and domain compatibility is not the same for all team members, it's best to opt for a middle ground. And for this reason, Shared Leadership is the best bet (Hoch and Dulebohn 2013). In Shared Leadership, monopoly of leading the team members is not restricted to one individual or hierarchy as multiple team members take up leadership roles. As Agile manifesto states, in an Agile development team, individuals and interactions should be preferred over processes and tools. So going by the description, Shared Leadership is the best fit for Agile development, as it maintains team integrity and inspires the team to work towards a common goal.

In spite of its many advantages, clients can even sight increased cost as one of the downsides of Agile (Gibbs, 2006). As Agile development would welcome change requests with open arms, such practice over a course of time would certainly extend the deadline leading to increasing cost of product development. But this very problem can be dodged or can be used to act in the client's favor using Agile development.

As mentioned in " Agile Project Management – a future approach to the management of projects", client can put Agile iterations to good use and make the best use of the application build (version) generated in the previous iteration or sprint. They can leverage the previous or

last stable build from previous sprints to demonstrate stakeholders and investors of their current progress and also show the analysis of the rate at which the application is evolving at each iteration. Such report not only justifies the need of Agile but also establishes its sheer necessity in every software development project.

According to an online study conducted by TechBeacon, our current software development is far away from being completely Agile.



According to the study, merely 16% of all development projects are completely Agile, while 51% of them are gradually leaning towards Agile. Even though nearly half of all software development teams are trying or at least considering being Agile, the transition won't be easy or happen overnight.

One of the foremost reasons why complete shift to Agile hasn't been possible is the phobia of change. Every project which is following Waterfall or some other methodology fear that transition from their existing methodology which seemed to be working and giving decent result to Agile can be tough or in worst cases devastating. As the team and the company as a whole has been well settled with their pre-existing development process, trying out any new development process (Agile) might disturb the flow and the rhythm that their team and clients are accustomed to.

For big companies, such shift is even tougher. As they have to lie down documents and compile their process documentation from scratch. That's not all, they have to educate their team members of the Agile methodologies and train them to how to manage their tasks and the different responsibilities they need to undertake. And then, one of the biggest hurdles for them is being able to convince the client of the distinct advantages of Agile and how they would be benefited.

These are some of the reasons why 7% of the development teams are still leaning towards Waterfall and 2% are still holding on to pure Waterfall.

Literature Reviews

S.No	Author	Title
1	Aljaž Stare	Agile Project Management – A Future Approach To The Management Of Projects?
2	Marian STOICA	Analyzing Agile Development – from Waterfall Style to Scrumban
3	Naga Sri Morampudi	Evaluating Strengths and Weaknesses of Agile Scrum Framework using Knowledge Management
4	Veeresh Thummadi Kalle Lyytinen Nicholas Berente	Iterations in software development processes: A comparison of agile and waterfall software development projects
5	Peng Xu Yide Shen	Leading Agile Teams: An Exploratory Study of Leadership Styles in Agile Software Development
6	M. Steven Palmquist Mary Ann Lapham Suzanne Miller Timothy Chick Ipek Ozkaya	Parallel Worlds: Agile and Waterfall Differences and Similarities
7	Sondra Ashmore	The Impact of Process on Virtual Teams: A Comparative

		Analysis of Waterfall and Agile Software Development Teams
8	Karlheinz Kautz Thomas Heide Johansen Andreas Uldahl	The Perceived Impact Of The Agile Development And Project Management Method Scrum On Information Systems And Software Development Productivity
9	Hoch, J. E. Dulebohn	Shared Leadership in Enterprise Resource Planning and Human Resource Management System Implementation
10	Gibbs, R. D.	Outsourcing and the IBM rational unified process

After going through the above-mentioned papers, the advantages of Agile development were quite evident. Not only does it encourage team bonding and taking ownership of the product, but also heavily rely on iterations so that they can deliver modules or parts of the application in quick successions. Inclusions of QA prevent defect slippage and also help in defect prevention at later stage. Agile methodologies instill a lot of trust among the client and also help them work in close coupling with the development team.

From all the literature reviews and after consulting IT professionals who work in Agile projects, the distinct advantages provided by Agile development was clear. But in spite of such evidences, there is quite a lot of work required to make Agile development the mainstream or default development strategy for software development.

Proposed Solution Approach

Agile projects are definitely more streamlined and have higher success rates than Waterfall development. In order to increase the chances of a successful product switching to Agile would be the only logical option.

Main Results Expected –

Once shifting to Agile is complete, the team would soon be able to witness the following results

- Increase in team's efficiency - The main reason for the increased efficiency is due to the fact that, after each release, the team can product a potentially shippable version to the client, no matter how small. So the team, clients and even the stakeholder would be able to judge whether their vision matches with the product being developed.

  Also, dividing the complete requirements into multiple sprint items creates a breathing room for change requests that further sharpens the application's capability.

- Enhanced product quality - With Agile, testing occurs more frequently and the QA team works in tandem with the developers. This results in early detection of issues and eliminates any blockers when the application approaches final release. Also, every team members apart from acting their part pitch in their

ideas and try to get periodic feedback from the clients to improve the overall quality of the product.

- Quick turn-around time - Right from requirement gathering and design to testing and product demo, everything occurs in iterations. This approach helps the Agile team to prioritize the deliverables properly while catering to the change requests proposed by the client. Such systematic process coupled with frequent interaction amongst the team members and clients, reduces the time required to ship the product into production for UAT (User Acceptance Testing) and finally to the intended target audience.

How the results will be produced? –

The above results are produced through

- Properly planned sprints - When the product specifications are provided by the client, the development team breaks down each and every requirement in small tasks and features. Based on the team size and their availability (available working days), the sprint capacity is calculated. The sprint capacity actually tells the team, how many features can be picked in a sprint (typically 1 - 2 weeks long). Once sprint capacity is calculated, story points are assigned to each and every feature based on the complexity of each task and how much time it would consume. Then these tasks and features are prioritized and assigned to different team members.

In this way, properly planned sprint reduces the chance of slippage and also keeps all the team members aware of the sprint deliverables. All this estimation is provided in sprint planning meeting and every team member is a part of it. They can raise any concern they might have regarding the requirement or its complexity. Such practice creates transparency and organizes the complete development process.

- Frequent Client feedback - Agile doesn't specify the frequency at which the client feedback calls should be scheduled. Client calls can be scheduled daily, especially during the requirement-gathering phase when there are a lot of grey areas or at least once a week, to let them know of the development team's progress.

How to tackle this issue? –

For today's fast paced development requirements, sticking to Agile methodology is the only logical option. So, in order to make software development efficient and add quality to the product, transition should occur from Waterfall to Agile. Some of the feasible steps to tackle this issue are

- Initiative to switch - The first step would be to initiate a switch to Agile. Any team member or project manager can themselves go through various Agile documents to understand what advantages Agile development has in store for the team as a whole. They should also take initiative to chalk down the project areas that would be benefited by this transition.

- Client approval - Before taking any major step or allocating any resource or planning for this transition, it is necessary to let the client know about the transition. They should be communicated of the switch to Agile from Waterfall, and the specific reasons why this transition is required. In order to get this approval, the project manager and other senior members of the team should be well prepared with the advantages of Agile and the disadvantages of Waterfall, that actually called for this change. In addition to that they must also be ready to answer any question, query or concern that the client might have.
- Spreading the word - Once client approval is received the next step would be to inform the team members about the transition so that they too can prepare themselves for the transition. During this phase, the very basics of Agile development should be imparted to the team members and anyone interested within the organization so that they too can initiate the implementation of Agile methodologies in their respective projects.
- Educating the team - This is the most crucial and effective of all methods. The organization can invite faculties and conduct sessions to over the basics of Agile. Such sessions can be held periodically for a few hours every day until every team member gets accustomed to the process. Internal sessions within the team can also be a very good way to further strengthen the basics of Agile. Referring Agile manifesto and other Agile development books or documents is recommended.
- Sprint 0 - Sprint 0, also known as inception sprint is when the product backlog is created. Once the team is up to speed with Agile, the first step would be to go

through the client requirements and design a product backlog. Then a basic environment is set up and high-level application architecture is laid down. So, sprint 0 signifies the starting of Agile development where all the features are broken down in stories, tasks and features and are assigned priorities, while a small piece of code is written to get a basic overview of the overall application architecture.

In order to successfully implement Agile, a well-designed plan is very important. But before designing a plan, we questioned a few IT professionals who are already working in an Agile project, to shed some light over the Agile practices.

- **Question** - Generally, the length of Agile sprints varies from one to four weeks. According to you what is the favorable duration of a sprint?

  -> At our project we stick to two-week sprints. Such duration is neither too long or neither too short. On keeping the sprint short (one week), the deliverables need to be broken down into very small features and task that sometimes aren't feasible. Also, with one-week sprints we have to conduct sprint-planning meeting four times a month. For this reason we stick to two week sprints.

- **Question** - What is the best way to accept change requests?

  ->Change requests are bound to come at some point of time and when they do, you need to clearly communicate to the client that it would be taken up in the upcoming sprints. So, when any change request arrives at the middle of a sprint,

we keep it in the product backlog and depending on the priority of the change request to address it in the subsequent sprint.

- **Question** - Agile calls for frequent feedback. How frequent should these feedback calls be?

  -> In our project we try to stick to two client calls in a week. So in a sprint (considering that it is a two week sprint), we have four client calls + one sprint retrospective call. Hence, five calls. In every week, we have a general technical discussion that targets any open questions or queries from the development team, and a demo call that is used to showcase our progress. While the retrospective call focuses on the practices that we follow and if any improvements are needed in the existing process needs.

- **Question** - Should the testing team be involved right from the design stage?

  -> Absolutely, engaging QA early in the development phase brings clarity in the project and in some conditions have also helped clear hurdles that might have come up eventually. During the design phase, QA team members question the feasibility of each and every feature and provide suggestions how erroneous scenarios can be averted.

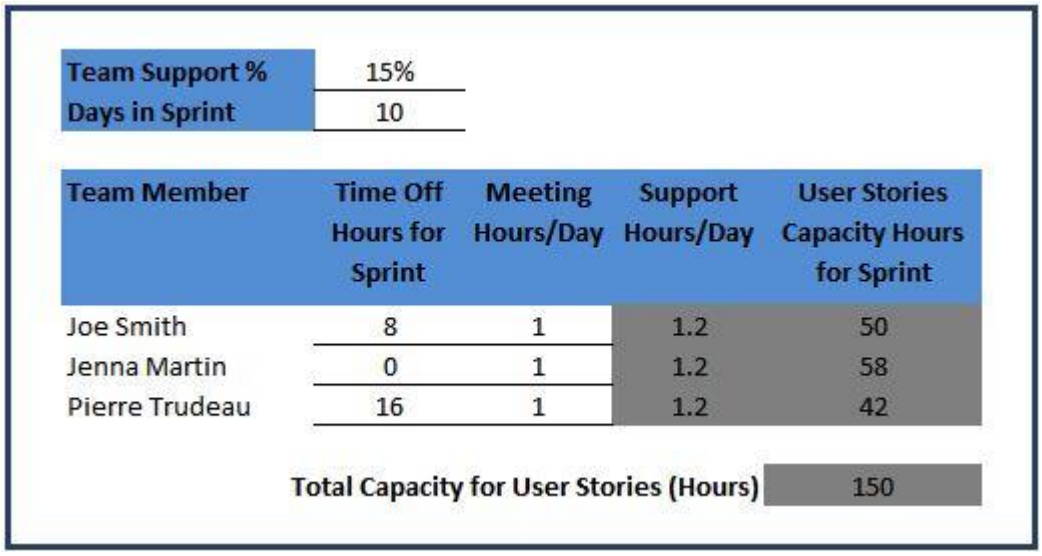
- **Question** - Story points are assigned to different user stories or features based on their complexities. How is it done?

  -> Story points are assigned in the sprint-planning meeting. Before providing story points, we set a benchmark for a task having complexity "1". For example, a task where a developer has to use a logo and display it on a web application,

has a complexity "1". Now, we compare each feature with this base story and provide story points. Anything which is a bit more complex than the base story is given a story point "2", while a much more complex task is given a story point "5". And the story point follows a Fibonacci sequence, i.e. 1, 2, 3, 5, 8....

After going through different Agile manuscripts and asking different questions to IT professions the following process has been finalized to successfully mark a transition to Agile.

- Creating Product backlog – When the entire team is at par with the concept of Agile, the first thing that need to be targeted in the product backlog. Team members would go through the entire SRS document and create short stories, tasks and features. If any feature is too big, an epic needs to be created. The purpose of the product backlog is to divide all the release deliverables into small chunks so that all these items can be divided among the team members and organized on the basis of priority.
- Calculating capacity of the team – The sprint capacity has to be planned at the starting of each sprint. The sprint capacity would give a rough estimate of the amount of working hours each team members has and on the basis of this, sprint items can be assigned to them. If any team member is on leave the working hours would reduce, so will the sprint capacity of the team. In this way calculating sprint capacity minimizes the chances of slippage. This neat table from Agile Reflections, clearly projects how sprint capacity is calculated.



Team Support %	15%
Days in Sprint	10

Team Member	Time Off Hours for Sprint	Meeting Hours/Day	Support Hours/Day	User Stories Capacity Hours for Sprint
Joe Smith	8	1	1.2	50
Jenna Martin	0	1	1.2	58
Pierre Trudeau	16	1	1.2	42
			Total Capacity for User Stories (Hours)	150

- Sprint planning and assigning story points – In order to reap the benefits of sprint planning meeting, presence of all team members is mandatory. In this meeting items from the product backlog would be taken up and discussed, and the complexity would be decided. The above questionnaire highlights the fact that the complexity of the task/ feature would decide its story point, which would be relative to the complexity of the base story. Every team member would share their view and in this process story points would be assigned to every sprint items.
- Sprint backlog – After sprint planning, once story points are assigned, all the user stories would be prioritized and added in the sprint backlog. The number of items in the sprint backlog would be referred in each sprint and items from this backlog would be picked depending on the sprint capacity.

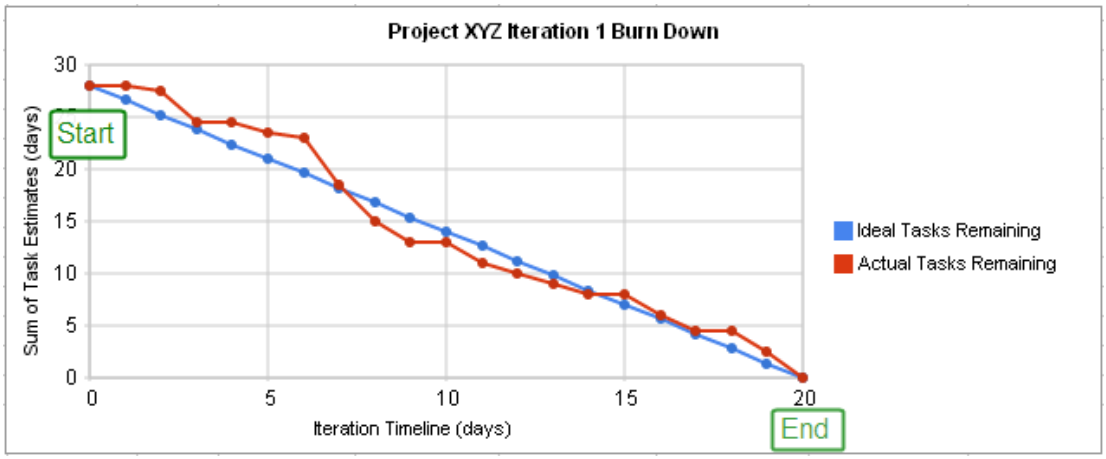
  Any new change requests that come at the middle of the sprint won't be part of the sprint backlog and would be move to the product backlog. In the next sprint-

planning meeting, these items would be prioritized and assigned to the upcoming sprint.

- Daily scrum – Daily scrum meeting should also be conducted so that each and every team member are fully aware of the items being delivered and get a clear picture about the items each team members are working on. In this meeting team members would talk about the items they worked on yesterday and what they are planning to do today. The meeting should be brief and should not exceed more than 15 minutes. Any technical discussion or queries are not the agenda of this meeting.
- Burn down Chart - A burn down chart is a graphical representation of work left to do versus time. Such a chart would not only help the project manager to keep track of the proceedings in the project, but also bring immense clarity for the client.

  There would be two types of burn down charts

  - Sprint burn down chart – It would be created every sprint, and would show the slippages and the pace at which sprint items are delivered.
  - Release burn down chart – It would be created after every release.



Proposed Work Plan

Items	Duration (in Days)	Start Date	End Date
Searching relevant literature	2	03/02/2017	03/03/2017
Reviewing papers and documents	3	03/06/2017	03/08/2017
Market research	8	03/07/2017	03/16/2017
Fabricating questionnaire	1	03/17/2016	03/17/2017
Collecting data	3	03/20/2017	03/22/2017
Creating Problem statement	1	03/23/2017	03/23/2017
Documenting literature review	3	03/27/2017	03/29/2017
Data analysis for proposed solution	3	03/30/2017	04/03/2017
Editing and document revisions	2	04/04/2017	04/05/2017

References

[1] Stare, A. (2013, May). AGILE PROJECT MANAGEMENT – A FUTURE APPROACH TO THE MANAGEMENT OF PROJECTS? Retrieved from http://www.sam-d.si/Upload/Articles/DRMJv02n01a04.pdf

[2] Thummadi, V., Lyytinen, K., & Berente, N. (n.d.). Iterations in software development processes: A comparison of agile and waterfall software development projects. Retrieved March 23, 2017, from https://goo.gl/MDgEQ1

[3] Faraj, S., and Sambamurthy, V. 2006. "Leadership of Information Systems Development Projects," IEEE Transactions on Engineering Management (53:2), pp. 238-249.

[4] Hoch, J. E., and Dulebohn, J. H. 2013. "Shared Leadership in Enterprise Resource Planning and Human Resource Management System Implementation," Human Resource Management, (23:1), pp. 114–125.

[5] Manifesto for Agile Software Development. (n.d.). Retrieved April 01, 2017, from http://agilemanifesto.org/

[6] Gibbs, R. D. (2006). Outsourcing and the IBM rational unified process. Upper Saddle River: IBM Press.

[7] Jeremiah, J. (2017, April 07). Agile vs. waterfall: Survey shows agile is now the norm. Retrieved April 11, 2017, from https://techbeacon.com/survey-agile-new-norm

[8] Posts about Sprint Planning on Agile Reflections. (n.d.). Retrieved April 19, 2017, from https://agilereflections.com/tag/sprint-planning/page/2/

[9] Burn down chart. (2017, April 14). Retrieved April 19, 2017, from https://en.wikipedia.org/wiki/Burn_down_chart

[10] Hodgson, P. B. (2013, April 22). PRINCE2 processes vs Agile development methodologies. Retrieved April 20, 2017, from https://zenexmachina.wordpress.com/2013/04/03/prince2-processes-vs-agile-development-methodologies/

Appendices

Survey Questionnaire

Survey Questionnaire

Perception of Project Manager's towards Agile

By: **Ankit Sachdeva**
For **GRAD 699 90 2017**

Table of Contents

Part 1

Background Information

Name:

Company:

i. How extensive have you worked in software development? Please mention number of years.

_____

ii. How extensive have you worked with Agile methods or its practices? Please mention number of years.

_____

iii. Mention the number of projects you have used Agile methods or its practices? Please enter number of projects. (e.g. 3/10)

_____

Part 2

Agile Acceptance

Please select the correct rating from the scale of 1 – 10 (1 being lowest and 10 being highest) and answer below questions best to your perception.

i. When using Agile, it enables me to accomplish my tasks quickly

1	2	3	4	5	6	7	8	9	10

ii. When using Agile, it improves the quality of work I do

1	2	3	4	5	6	7	8	9	10

iii. When using Agile, it enables flexibility in my tasks

1	2	3	4	5	6	7	8	9	10

iv. When using Agile, it enhances my effectiveness on the job

1	2	3	4	5	6	7	8	9	10

v. When using Agile, it gives me greater control over my work

1	2	3	4	5	6	7	8	9	10

vi. When using Agile, it is compatible with all aspects of my work

1	2	3	4	5	6	7	8	9	10

vii. When using Agile, it fits well with the way I like to work

1	2	3	4	5	6	7	8	9	10

Part 3

Cultural Acceptance

Please select the correct rating from the scale of 1 – 10 (1 being lowest and 10 being highest) and answer below questions best to your perception.

i. My team members have a strong sense of identification and commitment to the team

1	2	3	4	5	6	7	8	9	10

ii. My team members have the willingness to learn and change

1	2	3	4	5	6	7	8	9	10

iii. My team members have strong interpersonal and communication skills

1	2	3	4	5	6	7	8	9	10

iv. My team members are technically competent

1	2	3	4	5	6	7	8	9	10

v. My team members have collaborative attitude

1	2	3	4	5	6	7	8	9	10

Part 4

Impact of Agile Methods

Please select the correct rating from the scale of 1 – 10 (1 being lowest and 10 being highest) and answer below questions best to your perception.

i. Using Agile has enhanced the functionality of applications that we build

1	2	3	4	5	6	7	8	9	10

ii. Using Agile has decreased the number of errors in the systems/software products we build

1	2	3	4	5	6	7	8	9	10

iii. Using Agile has improved the quality of the systems/software products we build

1	2	3	4	5	6	7	8	9	10

iv. Using Agile has made me/my team more conscious of software quality

1	2	3	4	5	6	7	8	9	10

v. Using Agile has greatly speeded up our development of new applications

1	2	3	4	5	6	7	8	9	10

vi. Using Agile has definitely made me/my team more productive

1	2	3	4	5	6	7	8	9	10

vii. Using Agile has significantly reduced the time we spend in software/systems development

1	2	3	4	5	6	7	8	9	10

viii. Our customer/user(s) have been satisfied with the usability of our products since we started using agile methods

1	2	3	4	5	6	7	8	9	10

ix. Our customer/user(s) have been satisfied with the functionality of our products since we started using agile methods

1	2	3	4	5	6	7	8	9	10

x. Overall, our customer/user(s) are satisfied with us since we started using agile methods

1	2	3	4	5	6	7	8	9	10

xi. To what extent does the use of agile methods allow you to better predict the effort required for software development?

1	2	3	4	5	6	7	8	9	10

xii. To what extent does the use of agile methods allow you to better predict the quality of software that you develop?

1	2	3	4	5	6	7	8	9	10

xiii. To what extent does the use of agile methods allow you to better predict the delivery of software you develop?

1	2	3	4	5	6	7	8	9	10

Part 5

Comments

i. **Please use this page to provide us with any additional thoughts you may have regarding the use of agile methods/practices in your work environment. These comments may help us better understand your answers, or may provide us with future questions that need to be addressed in research. Again, we would like to thank you for your participation in this survey.**