Quickscrum

AGILE IMPLEMENTATION ISSUES FACED BY STARTUPS

(Prioritizing business value, monitoring projects & releasing the product)

COPYRIGHT © 2017 QuickScrum

Agile Implementation Issues (Prioritizing business value, monitoring projects & releasing the product) COPYRIGHT © 2017

All rights reserved.
No part of this publication may be reproduced, stored in a retrieval system
or transmitted in any form or by any means, mechanical, electronic, photocopying, recording or otherwise
without the prior written permission of the publisher.

For information, address Quickscrum SARL, 47, Rue Klock, 92110 - Clichy, France

Published by QuickScrum
France/Germany/India

INDEX

Summary

Many projects run into difficulties and fail to deliver the goals due to project management related issues, development concerns or inability to prioritize and deliver valuable productivity to clients in time. For start-ups and small to medium sized organizations dependent upon various types of funding, it is important to remain accountable to investors and produce financial gains over time to keep them interested to sustain the project.

Projects having a clearly defined scope and nature of deliverables are easier to execute and deliver as the team is able to focus precisely upon what the client needs and anticipates in terms of deliverables, and what it is supposed to deliver in terms of business value to the client. However, in real life scenarios, client-centric requirements often change forcing the team to rethink and re-plan its work, and set forth a new set of goals and objectives to fulfil the new vision. This causes disruption and loss of productivity.

Some of the commonest issues include:

- Inability of the team to understand and decide the nature of what business value (features and functionality) should be delivered to the client and when.
- How to track and systematically observe the project to ensure that the work process is properly in tune with and complies with the project vision.
- How to release completed work at regular intervals of time in a consistent manner in the market so capital can be raised in the form of ROIs.

The Agile framework offers reliable and productive project management processes for executing almost all kinds of projects - especially IT or software development related. It’s worthwhile to know how Agile processes can help to resolve these issues and why they’re better as compared to traditional approaches.

1. Preface

Agile processes like Scrum, Kanban and Extreme Programming "XP" offer a lot of scope for organizations to manage their development processes in an organized and productive manner to reduce waste, increase productivity and enhance their ROIs. For start-up companies, and small to medium sized businesses dependent upon investors and the market funding, it is very important to reduce waste (efforts, activities or results that don't result into product monetization), manage delivery deadlines and start generating a profit as soon as possible. Three common problems typically faced by companies are choosing the most important aspects of the project and when and how to develop them, how should important tasks be prioritized, how projects should be effectively managed to churn out the deliverables in time, and finally how should the deliverables be released in the market to generate a ROI in a consistent and reliable manner.

Not all entrepreneurs and CXOs have a technical background. Moreover, considering the present market scenario, most start-ups headed by young and enterprising people run into lots of problems owing to a lack of sound domain knowledge and work related experience. Younger folks usually don’t have enough time and working capital, and can’t afford to sustain resources for a long time to experiment with new concepts and ideas. For them it’s important to hit the nail directly upon its head and hammer in productivity in a correct manner and at the correct time in their projects to make them successful.

Without using a lot of technical jargon, this write-up tries to explain the basics of three important issues faced by start-ups as mentioned above, to help people new to or beginning with Agile to understand how Agile based value delivery processes can streamline work and handle projects in an effective manner.

2. Why do project fail - An introduction

Many a times, complex or long term projects exceed their budget, or deadlines, or even both, or fail due to some reason or the other. For investors and clients the ROI is the main point of concern. If break even points are not achieved early in the project it often leads to financially stressed conditions which tend to force managements to undertake drastic steps to curb additional expenditure, or simply stop the project if additional funds are not available to sustain it. One of the commonest factors which leads to non-productive projects is the inability of the team to deliver project value at regular intervals of time. Rather than indulging in post project failure discussions to retrospect the causes of a failed project, effective monitoring of projects can help teams and managements to streamline ongoing projects and make them successful.

3. Why projects should be monitored?

To monitor a project is to routinely gather information pertaining to all aspects of the project, and observe and record all activities occurring in it. Monitoring involves a systematic and purposeful observation of all ongoing processes in the project. It also includes giving meaningful feedback to the investors and project owners regarding the project status and how the milestones are likely to be achieved over time so informed decisions can be made by them.

Monitoring does not involve just staring at a computer monitor spewing out data from analytics tools and generating reports for the management and team members. The project manager or the process-in-charge needs to understand the metrics and forecast where the project is heading in accordance to the efforts put in by the team and how well the team is performing at the moment. Moreover, other constrains such as budget availability, short staffing, project

deadlines and other issues popping up in day-to-day activities also need to be considered as to how they're likely to affect the success and sustenance of the ongoing project.

Monitoring provides information as to what the status of a particular program, project or policy is at any moment, or is going to be over time, and how well the functioning of various processes in the project, including the resources allotted for it relate to targets and deliverables. Its focus should also be on optimum utilization of the resources made available for the project. The objective is to track the gap between what was originally planned and what is actually happening now.

Therefore, the primary reason why projects should be monitored is to:

- Get sound visibility into project execution.
- Determine what actions need to be taken to ascertain that project objectives and goals are successfully met.
- How project goals relate to team efforts, delivery schedules and quality of deliverables.
- Allow the team to educate and learn for itself from its past experiences and improve its productivity levels.
- Make the team accountable for the work it carries out by evaluating the performance metrics.
- Justify the capital invested by the stakeholders and investors.

3.1. What to monitor?

The nature of the project, its goals and objectives, and its deliverables primarily indicate what parameters should be ideally monitored and in what manner. It can be difficult to generalize the "what" aspect since it may differ from project to project. Generally, the key performance indicators (KPIs) used for monitoring the progress levels in a project are set up collectively by the client and project manager based upon their project related needs and deliverables. Some of the important aspects to monitor in a project can be:

Objectives and needs of the client

Is the client's project vision fulfilled and are the project's milestones successfully met?

Efficiency level of the team

Is the team working as per the development plan and are enough efforts put in to meet the deadlines?

Collaboration and communication levels

Is the team working together in a harmonious manner and pursuing the goals collectively with a single focus?

Business value and work monetization

Is the work delivered by the team monetizable?

Risk mitigation

What are the risks involved in the project? Can they be identified in time? Can they be resolved?

3.2. Levels of monitoring

A project should be ideally monitored at three main levels to get a clear insight regarding its current status so corrective measures can be taken well in time.

3.2.1. Activity level monitoring

It is done to ensure that each activity defined in the project schedule is carried out in a proper manner and within the time box allotted for it. It can be done by holding daily meetings with the team, or alternately the project manager can check the status of all tasks scheduled to be carried out for the day. Typically, the daily tasks to be carried out by team members are listed on paper and their completion status is checked at the end of the day to identify any pending work remaining on that particular day. This level of monitoring is basic to all project management processes since it makes the team accountable for the work assigned to it. It is to ascertain that the project proceeds as per plan and delivery deadlines can be met successfully.

3.2.2. Project status reporting

Usually generated once a week, project status reports often contain a summary of all tasks completed successfully by the team in the week gone by against the actual activities planned for that week. The idea is to identify how much of work is completed in the project, how much of it is still pending, and whether the deliverables can be produced by the team considering its current rate of delivering work. It also helps to identify the causes of delays logged by the team and pinpoint any pitfalls hampering team productivity.

3.2.3. Milestone analysis

Analysis is done every few weeks to make sure whether milestones supporting the project vision are, or can be, achieved within the time duration allotted for its completion, and to find out how much of efforts are actually put in by the entire team over time to achieve the proposed milestones. When plotted on a graph, it helps to identify any deviations occurring in the proposed “line” of project completion versus time. If the deviation is severe or more and objectives are not met with time, project managers need to identify and understand the root causes and undertake corrective measures in time to correct the deviation so milestone deadlines can be met successfully.

3.3. Monitoring Agile projects

The success of an Agile project is not dependent only upon whether the process is properly implemented in the project or not. Three factors count – monitoring the progress, monitoring how Agile principles are followed by the team, monitoring how the business value is acquired.

3.3.1. Monitoring the project progress

There are three main metrics for monitoring the project’s progress.

3.3.1.1. Sprint Burndown

An Agile team organizes its work in time boxed durations called “sprints” which range from two weeks to a maximum of one month. The entire software is developed by the development team in these sprints. The sprint burndown metric tracks how the team has completed work throughout the sprint.

3.3.1.2. Release Burndown

While the sprint burndown metric focuses upon work completion at the iteration sprint level, the release burndown metric takes into consideration the status of various sprints covering the entire project. In Agile the product is released in stages – in the form of individual releases - by developing a set of features for each release. Business value in the project is acquired by releasing the product features in the market in the form of planned releases or versions ranging over a specific duration of time. The release burndown metric is useful for determining how much work is still left in the project and when the business value in the project can be realized over time. It is also useful for identifying any “scope creep” i.e. if more requirements are added to the original project and the scope of the project has increased significantly. To avoid scope creep from affecting the value of deliverables, the product owner always suggests important stories having high business values to the team for development purposes in the sprint.

3.3.1.3. Velocity

It is the amount of work, in the form of user stories that the team completes during a particular sprint. It is an important metric for forecasting how much work the team can handle in a sprint and the product owner uses it to decide how many stories should be selected for the sprint. It is a metric tracking the total work completed in the sprint versus sprint days. As the team executes sprints over time and takes up more work in the sprints, the metric improves and becomes more consistent over time.

3.3.2. Monitoring Agile implementation

In Scrum, an Agile based process, a special role is dedicated for overseeing the implementation of Agile principles in the project. The Scrum Master functions as a facilitator and ensures that the team follows the process in a proper manner. He/she ensures that the team does not face any impediments while working and also overlooks the various events or ceremonies involved in the process.

3.3.3. Monitoring the business value

A major difference between tradition project management methods and Agile is that the former focuses upon delivering software fulfilling just client based requirements while Agile believes in maximising the ROI through continuous delivery of shippable software and reducing waste. Re-planning of the deliverables can be easily done using Agile. Therefore, Agile measures outcomes over outputs and measures the Earned Business Value "EBV" metric. The metric also takes into consideration how efficiently and effective the team is performing while it delivers the business value in the project.

The business value can be calculated as:

$\mathbf{EBV_n}$ = Earned Business Value
= The total of all Business Value Indices for completed stories
= $\sum \mathbf{BVI_{(story)}}$ for all the stories completed in the first n sprints.

3.4. Types of issues faced during Agile implementation and how to resolve them

Based upon the open-mindedness of the team, its cultural and education levels, the degree of professionalism and the willingness to accept change management, development teams may find it moderately or very hard to accept Agile. Some common issues are mentioned herein.

3.4.1. Agile implementation issues

There can be many types of issues related to Agile implementation. However, a couple of common ones are discussed here.

3.4.1.1. Resisting change

This can be a very common issue in Agile implementation. Teams following traditional development methods are very much used to the staged processes and feel comfortable delivering work that is tested for bugs and regression in the next process stage. Developers focus only upon coding aspects and don't have to bother about optimizing the code or verifying their work whether it aligns with the project requirements or not. It can prove to be very difficult for them to adjust to the product incremental method typical to Agile processes in which the development team is not only accountable for its work, but is also required to focus upon testing the code and ascertaining that the work delivered by it fulfils the product vision and end user requirements.

Moreover, each team member is required to own the project and hold himself or herself accountable for his or her work. In addition, another aspect is that the team has to estimate how much work it is capable of accomplishing in a sprint and justify the reasons why it can only take up limited or less work for development at a time. Unlike in staged processes, Agile teams have to commit how much time it will take to deliver work and live up to that commitment.

The third aspect is that Agile promotes a working atmosphere in which each member in the team has to think like an entrepreneur and find innovative ways to work smartly and make active efforts to accomplish work in a shorter span of time. This can go hard with the mind set of teams used to traditional methods in which they are not required to think in any other manner except as developers or designers.

Development teams are generally reticent about change management and try to actively or passively oppose change when it is required to be introduced to improve the quality and quantity of deliverables. With Agile, the resentment is even more since the team members have to undergo a drastic mind-set change and learn about the new process.

Solution:
The management should try to introduce Agile after explaining the benefits of the new process and what the organization and the team stands to gain from Agile experience in the long run. It is important to convey the message that change is inevitable and Agile is to be implemented irrespective whether any team member decides to resign or discontinue with the process. Moreover, the management should also try to motivate the team and create an open environment supporting transparency and freedom where the team can frankly discuss its apprehensions and find a way to proceed with the change.

3.4.1.2. Understanding the process

Agile introduces fundamental changes in the way most people following traditional methods work. Teams have to open up their mind sets and think creatively, in an optimistic manner, and welcome the change. Moreover Agile processes like Scrum have ceremonies and artefacts which can be totally new concepts for most teams. Developers have to understand the product incremental model and its cycle, what continuous development and continuous integration is, devlops, and a host of new concepts to reckon with.

3.4.1.2.1. Estimating work

Developers are not very good at estimating work completion and deadlines. Generally, they hate a

situation where they're forced to commit a date or time and rigidly stick to it. This is exactly what Agile demands. You not only have to estimate work but make all efforts to deliver it in time with the required quality.

3.4.1.2.2. Roles in Scrum

In contrast to traditional or classical methods, Scrum - an Agile process - doesn’t have and doesn’t need any manager, task manager or a team leader. The three roles are:

1) Product Owner
2) Scrum Master
3) Development team

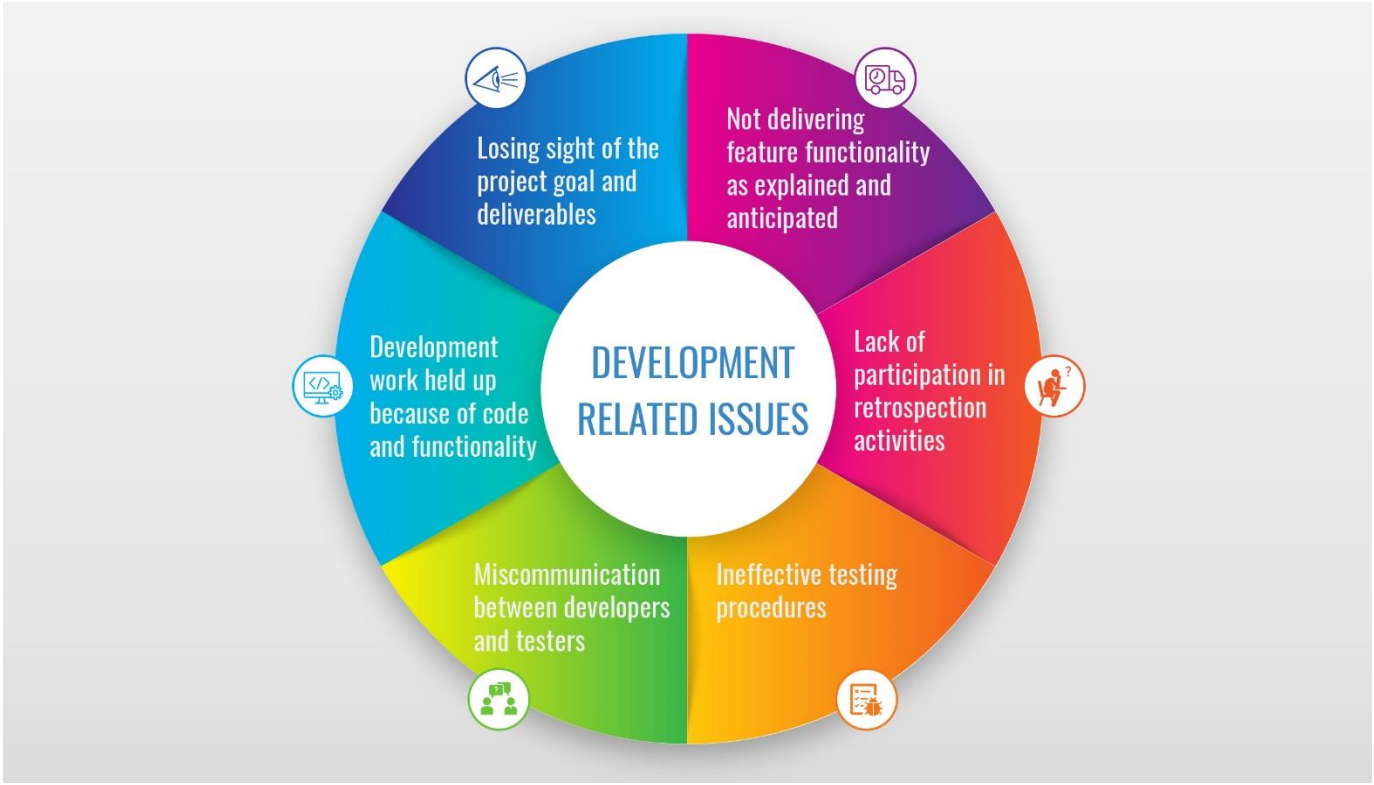
These three roles are coequal. Each role has certain responsibilities. Also each of these three roles have to fulfil their responsibilities by closely interacting and working together to finish a project successfully. Team members may find it difficult to understand and follow these roles.

3.4.1.2.3. Meetings and ceremonies

Scrum process includes several types of meetings or “ceremonies”. Agile teams are cross functional in nature. They take part in the ceremonies and make important decisions. Non - cross functional teams may find it hard to understand and accept what these ceremonies stand for and might resent taking part in them.

3.4.1.3. Development related issues

Development related issues are highly common with teams starting with Agile practices since they are not much familiar with the product incremental structure of the development process. Agile teams are cross functional and often developers also perform test cases to check code quality and reliability. At times, teams employ specialized testers and QA professionals to carry out quality tests. Whatever the case, if Agile principles are not followed in a proper manner it directly affects productivity and performance levels.



3.4.1.3.1. Losing sight of the project goal and deliverables

In Agile, the business value in the project is availed through staged and planned releases. Therefore, the development team needs to focus more upon delivery of value through it work rather than just coding feature requirements. In practice, teams often experience time constraints while meeting sprint deadlines. So over time they may lose their focus of delivering valuable work and start concentrating fully upon work completion. Teams stop thinking about what the product vision is and what goals the project needs to achieve to sustain itself. This leads to unfocused work efforts and disorientation – the team keeps on developing features and doing its work because it is supposed to, and not because the project goals and objectives should be achieved. Developers and designers stop being creative in their work. They stop innovating new and better ways of developing features and working functionality. As a result the velocity metric indicating the total work completed by the team in the sprint stops improving, thus conveying the fact that the team has stopped learning from its past experience and not speeding up work. This is contrary to Agile principles.

Solution:
The Scrum Master and project leads should constantly try to motivate the team members and hold healthy discussions so ideas can be exchanged regarding how the development process can be streamlined, made more easy and less time consuming using online tools and development aids if possible. Retrospection activities should be encouraged. Teams should be made more accountable by holding short meetings at regular intervals of time (all though this is not recommended by Agile – the principles state the team should take work ownership but in practice the team members actually don't which is the main problem) to obtain feedback regarding the work status. The management should also step in and try to create a healthy and conductive working environment to reduce the stress levels.

3.4.1.3.2. Not delivering feature functionality as explained and anticipated

The client expects a particular feature to work in a particular manner whenever he/she requests its development. Each task or feature requirement in Agile – known as a user story – can be explained in terms of what criteria needs to be fulfilled for it to be considered as properly developed and shippable. It helps to make the feature valuable and monetizable after its development. The team starts ignoring this criteria due to some or the other reason, or loses its sight while building it. As a result a feature is developed that is not in tune with what the client expects or needs.

Solution:
More stress should be given for the team to focus upon the acceptance criteria and the definition of done (DoD) which determines the value of the feature and its importance. Developers should be reminded again and again to create work in tune with these criterions and ensure they test the feature after they develop it.

3.4.1.3.3. Lack of participation in retrospection activities

Agile stresses very heavily upon the "inspect" and "adapt" principles which form the core of all Agile processes. The team has to learn constantly from its past mistakes and ensure they're not repeated again in the future. Also, new and better ways of doing work should be discovered to increase productivity levels and deliver more valuable work. All this is made possible through the retrospection activities in Agile. In Scrum a special ceremony known as the sprint retrospective meeting is held at the end of each sprint to support this principle. The team stops participating in these activities as a result of which further improvement in work in not availed and the velocity index stops improving. Further growth stops in the project.

Solution:
Retrospection activities should be made mandatory and team members encouraged to participate in them. If required, team members can be told to discuss new ways of improvement and prepare a list or a plan stating what new activities they propose to do with proper call-to-actions.

3.4.1.3.4. Ineffective testing procedures

Testing procedures - especially unit tests - form an integral part of the development process and should be carried out preferably in the sprint itself when the particular feature is being developed. Testing ensures that the definition-of-done is properly fulfilled by the team, there are no bugs in the feature and that the

feature is valuable to end users. Many times teams don't carry out these tests since they may be time consuming, or the team lacks the necessary resources or skill sets to perform them. The work delivered in such cases may contain bugs and lead to regression. The project owners may incur technical debt in the future.

Solution:
Testing activities should be made mandatory and teams should start considering testing procedures as an integral part of development process. They should be accounted for while the team estimates work during sprint planning sessions held just before a new sprint is planned.

3.4.1.3.5. Miscommunication between developers and testers

Effective communications are required for doing quality work. When team members discuss things and exchange ideas, thoughts become clear, and the entire team comes to know about the nature of deliverables and how they should be built. In Agile, testing should be a part of the development process and testers should work closely with developers to resolve test related issues and problems. When testers stop communicating with developers for test issues, or developers stop responding to testing queries, it leads to a situation where partially or improperly tested features are released in the market which can reduce the project value in the long run.

Solution:
The product owner or the project lead should motivate team members to collaborate and exchange ideas. At the same time developers and testers should be made more accountable regarding the quality of deliverables – especially fulfilment of the DoD - to ensure only fully tested and valuable features are developed at all times.

3.4.1.3.6. Development work held up because of code and functionality

Big or complex project development can have a lot of dependencies as far as code functionalities go. Often a piece of code created by one developer functions as a source of input for the task to be developed by another developer. In such cases, the second developer has to wait for the first developer to complete his or her work and create the particular functionality before the later one can start with his or her work. During the second half of the sprint planning session when the team members are selecting and taking up their development tasks, they should also discuss about code dependency issues and organize their work accordingly. When team members don't analyse their tasks and don't discuss the dependencies, it leads to a situation where one developer is loaded with urgent work while the other one has to wait for the first one to complete. Time is wasted and productivity at the team level is greatly reduced. It decreases team velocity.

Solution:
The developers and team members should work more closely and analyse dependencies before they take up their tasks. If dependencies still exist, each developer should plan his or her work with regards to the work to be carried out by the developer who's going to be affected by the particular dependency.

3.4.1.4. Communication and collaboration related issues

Communication levels play a vital role in a project's success. Right from project inception to delivering goals and objectives, the team relies upon timely and effective communication to develop product features,

share ideas and resolve technical issues. With remote or distributed teams working in the project, communication between project managers and teams can often prove to be difficult owing to different time zones and cultural differences. With in-house teams, effective communication lays the foundation of a shared product vision and harmonious working environment in which the team can take informed decisions by discussing, analysing and working together.

3.4.1.4.1. Communication

The sixth principle in the Agile Manifesto states:

"The most efficient and effective method of conveying information to and within a development team is face-to-face conversation."

In Agile, it is crucial to have effective communications between developers and the client, end users, and the stakeholders. The team develops software as per the product vision explained by the client. In turn, the client verifies work delivered by the team at the end of each sprint. Team members need to communicate with each other to resolve development issues. At every stage communication is vital and when it stops, it could raise some issues.

Some of the causes could be:
- Miscommunication of requirements
- Different time zones
- Failure to focus and listen properly
- Culture differences
- Attitude and ego related issues
- Gender bias
- Inadequate knowledge

3.4.1.4.2. Collaboration

Collaboration amongst team members can be challenging since developers are used to their individual methods for creating code and working functionality, and may find it very hard to follow some one else's coding methods. Agile processes favour continuous development. It is common for the team to use GitHub or Git to share, update and version the code before final changes are uploaded for deployment. It is recommended the team follows certain coding standards to make the code readable to others. To work effectively, the team needs to constantly collaborate and share ideas and suggestions to make things work.

Some of the common issues faced can be:
- Priority conflicts
- Insufficient alignment amongst team members
- Limited automated software development processes - lack of Continuous Delivery
- Coordination challenges
- Unpredictable software delivery
- No visibility over status, progress and results

4. What is a product release?

When a new product, or a set of new product features providing some value to customers or end users is launched in the market, the launch activity is described as a product release. A product release helps to create a value stream

for business owners. Its function is to bring in capital for the business by generating a profit. Product based companies – especially those following a SaaS model - have to depend heavily upon releases to grow and sustain their growth over time.

Organizations can grow only if the products they manufacture are released at the correct time and in the correct manner. If a product is released prematurely before it can be properly tested and made shippable, it could result in end users experiencing bugs or regression. This can lead to bad user experience and even affect brand reputation. On the other hand if a product is released too late it could prevent the organization from availing the business value in time leading to stressful financial conditions for the project owners. Projects may be forced to close down if they can't sustain themselves over time. Therefore releasing the product in the right manner at the right time is very important.

Both traditional Waterfall processes and Agile methods facilitate product release. However the manner and frequency of the releases vary considerably depending upon which method or process is employed by the team to manage the software project.

5. Releasing projects the traditional way

Traditional processes are best suited when the scope and time of the project are fixed. However, in practise, stakeholders may request changes in existing features and functionality as and when market conditions change. This can delay the development process and extend the deadline since the team may be forced to take up unplanned design and/or functionality related work which it has not anticipated and accounted for at the time of project inception. In Waterfall processes as the project completion and the release date are fixed at the onset, the development team is often kept under pressure to deliver work – both planned work as well as feature changes related - within the stipulated deadline. Teams may find it difficult to cope up with the added development activity and start cutting corners to complete work in time. As a result, the release may contain bugs and have faulty functionality which reduce user experience and even lead to bad brand image.

Owing to the staged process flow, Waterfall processes are irreversible in nature. The product can be released only after the project completes. Moreover, since only a single version of the product can be designed at a time in the project, you can’t have multiple product releases supporting upgrade versions for the same product in the same

project. It's also difficult to address risk mitigation since defects found in later stages cannot be reworked upon and corrected by reverting back to prior stages. It can be also difficult to adapt to changes in the product design once the project is documented and the development process starts. Therefore the business value of the release cannot be increased or enhanced once the project deliverables are decided.

Thus, you can be sure of releasing your product on time using traditional methods only if you're absolutely sure about the project's scope, know exactly what you're going to design and develop in your project and whether you can estimate your team's velocity (the speed or rate at which the development team can complete work in a given duration of time) correctly.

6. How projects are released using Agile

The success of a product release depends primarily upon what functionality the features offer in the release and how useful they are to end users. The business value in a project can be availed only if the releases are properly designed and planned. To make a release valuable, Agile tends to focus upon the following aspects:

- Can the business value be delivered in the form of a releasable product?
- Is the goal of building a reliable product with the desired quality met?
- Can the project successfully deliver the product releases on time?

6.1. Planning the releases

Since development is carried out in product incremental cycles in Agile, the release of a product depends primarily upon the development team's velocity metric which indicates how much of work the team can handle or get done per iteration cycle. Given the speed at which the team can deliver shippable product features, it can be planned how many sprints are required to develop a particular set of features. The features set is made available to the end users in the form of a release. The goal of a release is to deliver working software to end users as quickly as possible.

For explanation, suppose a product has 9 features. If the team has a capacity of developing 1 feature per iteration cycle or sprint, and each sprint lasts for 2 weeks, a release containing a set of 3 features can be released in the market every 6 weeks. Since the product contains 9 features, if 3 features are planned for each release, it would take 3 releases to launch the entire product.

- As the team becomes more conversant with the product with each development cycle, it tries to improve upon its velocity metric and speed up development work. More work can be taken up by the team per sprint which can reduce the total number of releases required to launch the product.

- Since features are launched in sets in releases, it becomes possible to receive end user feedback at the end of each release rather than receiving the feedback after the entire product is deployed. Product owners can plan further rework to enhance the business value of the features even while the product is being developed. The rework can be taken up by the team as a "feature enhancement" task which can include all suggestions provided by end users to make the feature functionality more useful and powerful.

- Business value is delivered to the client at the end of each release. Therefore, it becomes easier to achieve the breakeven point and fulfil the marketing milestones.

6.2. Designing the sprints

A sprint is a predefined duration in days (generally lasting 1 or 2 weeks or more but not extending 1 month) during which the team develops valuable shippable software and delivers it to the client. Each sprint is planned in a special Agile event known as the sprint planning meeting. During sprint planning sessions, the product owner (a person responsible for delivering the value in the project) selects a few valuable and prioritized product development tasks (user stories) and presents them to the team for development purposes. The team decides how much work (number of tasks) it can complete in the development cycle (sprint) and creates a sprint backlog (a list of finalized features and working functionality) to be completed during the upcoming sprint.

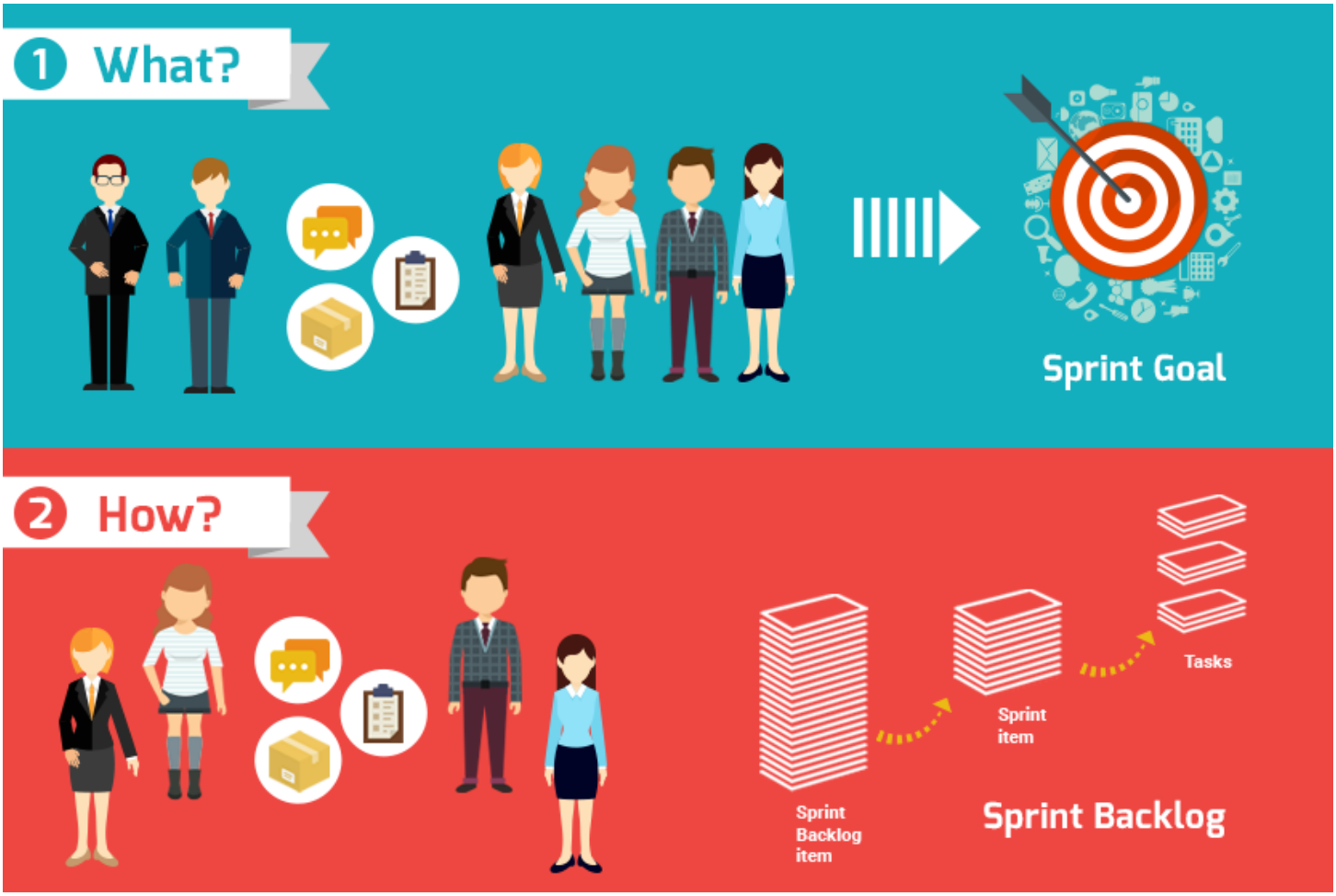
The team understands how each feature is to be developed and what criteria (acceptance criteria) should be fulfilled so that the particular feature can be considered as successfully completed (fulfil the Definition of Done). If the team has any doubts or needs any clarifications the product owner helps the team to understand the deliverables and what end users expect out of the proposed functionality.

The team members i.e. the developers, designers, testers, technical writers, data base administrators etc. than take up the proposed tasks from the sprint backlog based upon their levels of expertise and start decomposing the work (tasks) in a manner such that each task can be developed independently (provided they don't have any dependencies).

Working software is than developed by the team during the sprint.

The objective of a sprint planning session is to:

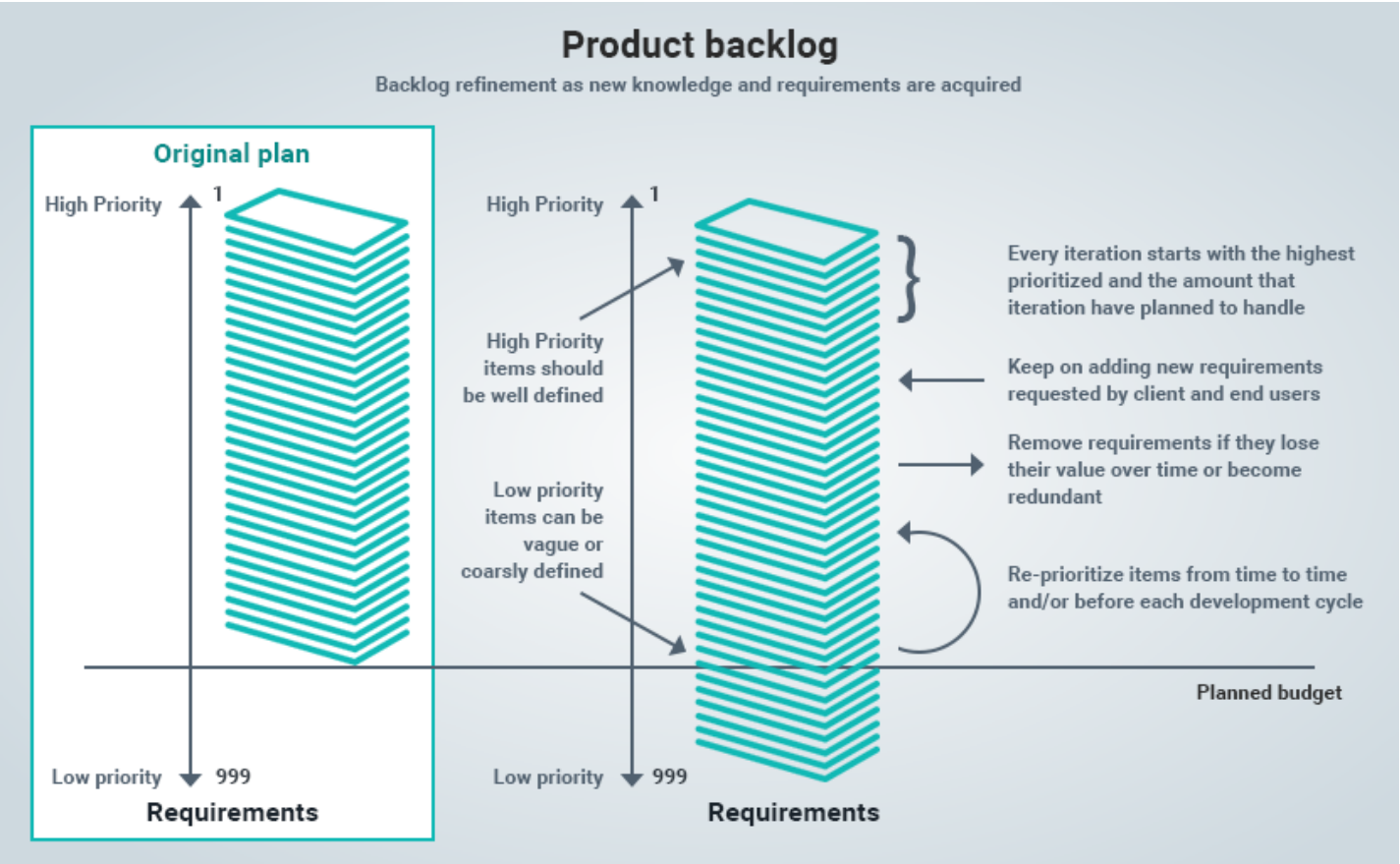
- Establish a clearly defined goal for the sprint.
- Choose valuable and important tasks or user stories that support the sprint goal.
- Break or decompose work (user stories) into specific development tasks.
- Create a sprint backlog.



6.3. Prioritizing the product backlog

Broadly speaking, the product backlog is a list of all activities (feature development, testing, documentation and deployment) needed to build the product. Each project has a product vision that explains what the final product should be like i.e. what kind of features the product should have, what type of functionality the features should offer, how end users shall use the features etc. People involved with the project viz. stakeholders, end users, technical team and the product owner suggest feature requirements and functionality that can fulfil this product vision.

The product backlog, therefore, is a "wish" list of all functionality desired in the product and keeps on growing in size over time as more and more requirements keep on adding in the list. Agile focuses upon the development of most valuable and important features first followed by less important ones so that maximum value in the project can be tapped and delivered in the form of working software to the client at the end of each product incremental cycle - the sprint. Hence it is required to arrange the product backlog such that valuable and most important tasks or development activity is contained at the top whereas less important requirements are placed in the middle. The process of prioritizing the product backlog as per the business value of the stories contained in it is also known as “backlog refinement” or “backlog grooming”.



7. Product monetization in Agile versus traditional methods

TRADITIONAL	AGILE
1. Single release in the project.	1. Multiple releases.
2. User feedback available at the end of the project.	2. Feedback available at the end of each release in the project.
3. Cannot inspect and adapt to end user requirements.	3. Easily adapt to end user requirements by planning feature enhancement tasks in sprints.
4. Avail profit only once when the project complete.	4. Acquire financial benefits in stages after each release. You don't need to wait for project completion.
5. Monetization value of the project is determined and fixed before project starts.	5. Value of the project can be increased by planning additional product features in upcoming sprints and releases.

8. Prioritizing product features - An introduction

In layman's terms, a product can be considered as successful only when huge number of people use it and the owner can earn a good profit by selling it in the market. The "power" of a product can be judged by the kind of features it supports and how useful those features are to end users. Therefore, it is important to capture the requirements of end

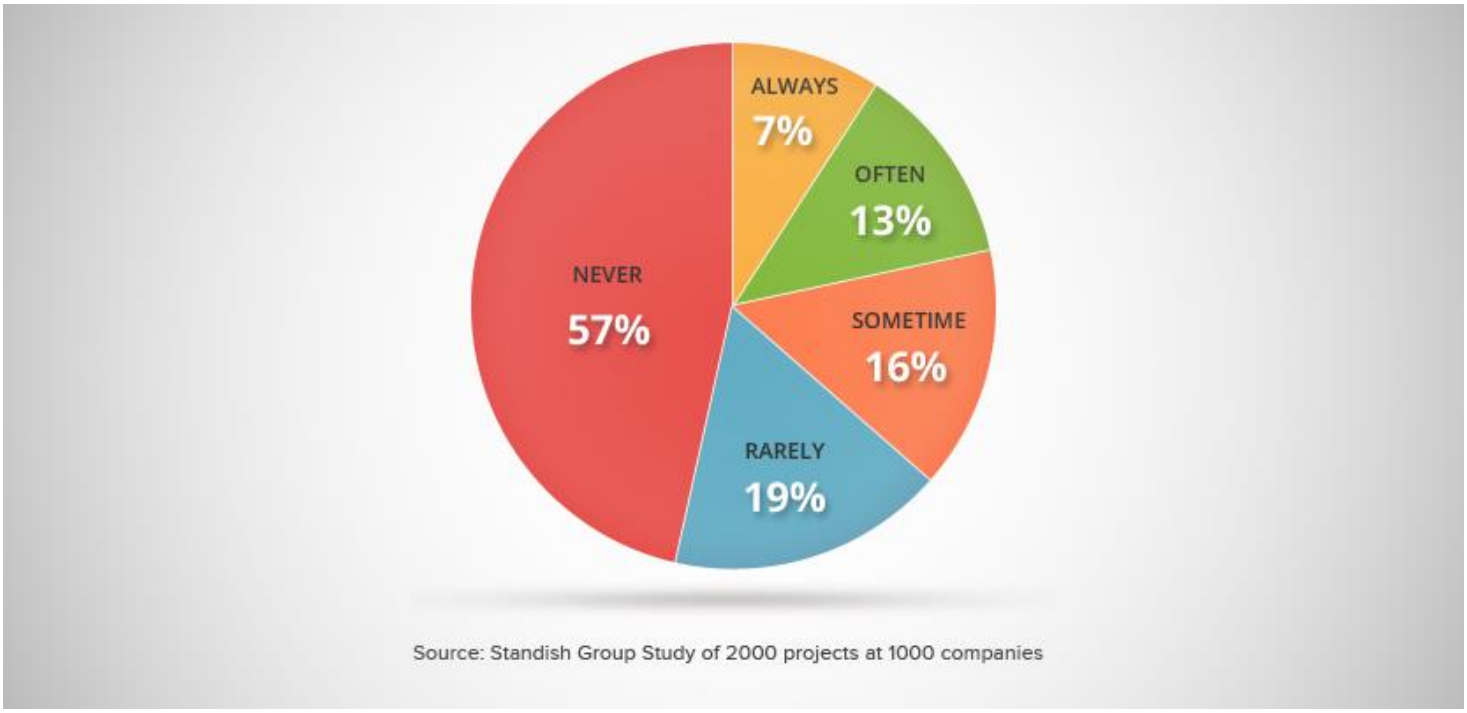
users in a systematic and organized manner in your product, and determine how important they are. The business value in Agile is indicated by how much important a feature, or a “user story” is from the client or end user's perspective.

9. End users don't use all features all the time

Users don't always use each and every feature offered by a tool or automated process. As per observations:

- 57% of the features are never used
- 19% are rarely used
- 16% are used sometimes
- 13% of the features are used quite often
- **7% of the features are used on a regular basis**

Agile focuses more upon the development of these frequently used 7% of the features which carry more business value.



10. Prioritizing product features the traditional way

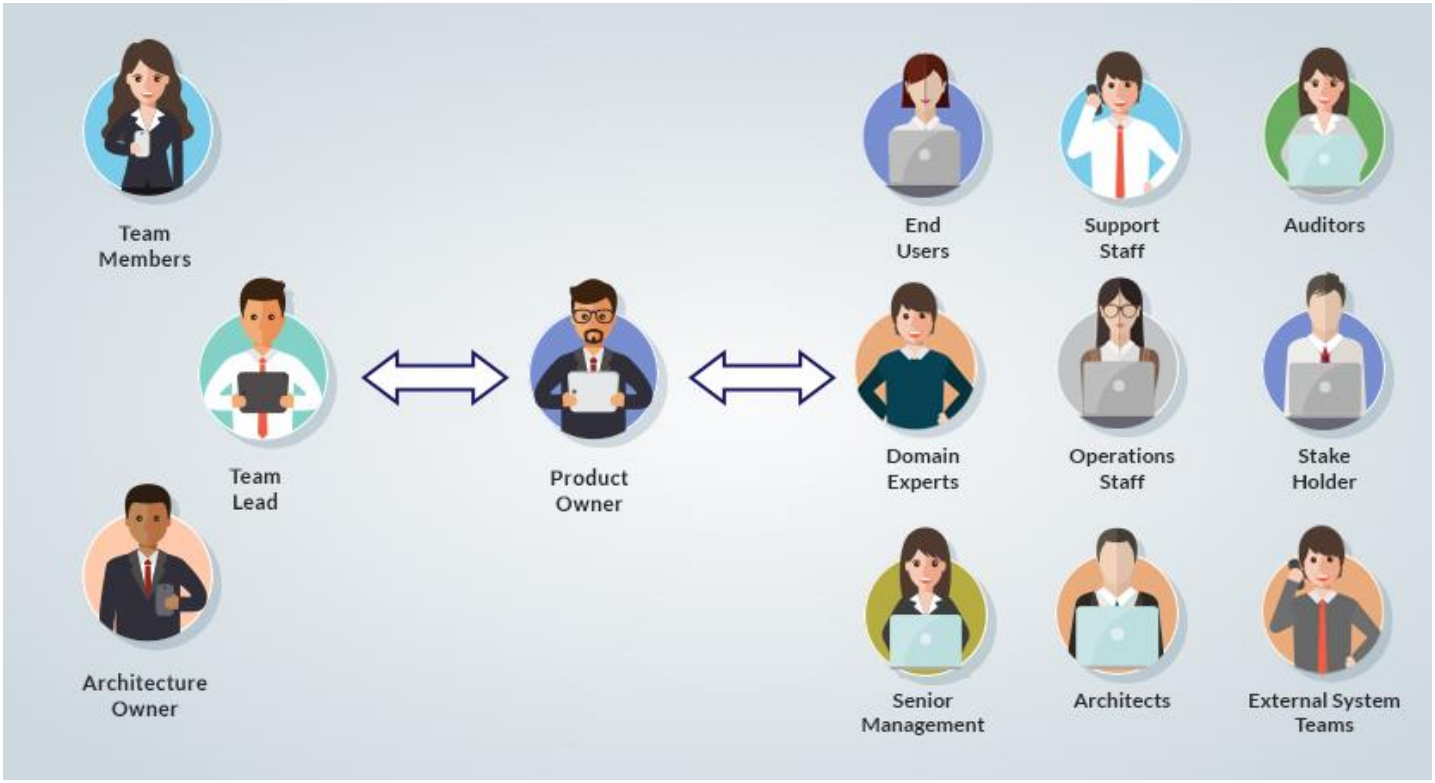
Traditional development methods follow a sequential (non-iterative) design process involving conception, initiation, analysis, design, construction, testing, production / implementation and maintenance stages. Requirements are gathered from the client, project owners, stakeholders, end users and other sources. These requirements are documented in the requirements phase at the start of the project. Once defined and documented, the requirements cannot be changed in subsequent stages. Therefore, in Waterfall methods once end user requirements are captured in the project new requirements can't be added later on. Moreover once documented, the requirements cannot be changed or updated in subsequent stages even if stakeholders or end users want modifications in existing functionality. So if you're following Waterfall methods you can capture end user requirements only once in your project when it starts.

11. How features are prioritized using Agile

It's not very difficult to capture client-centric requirements in Agile:

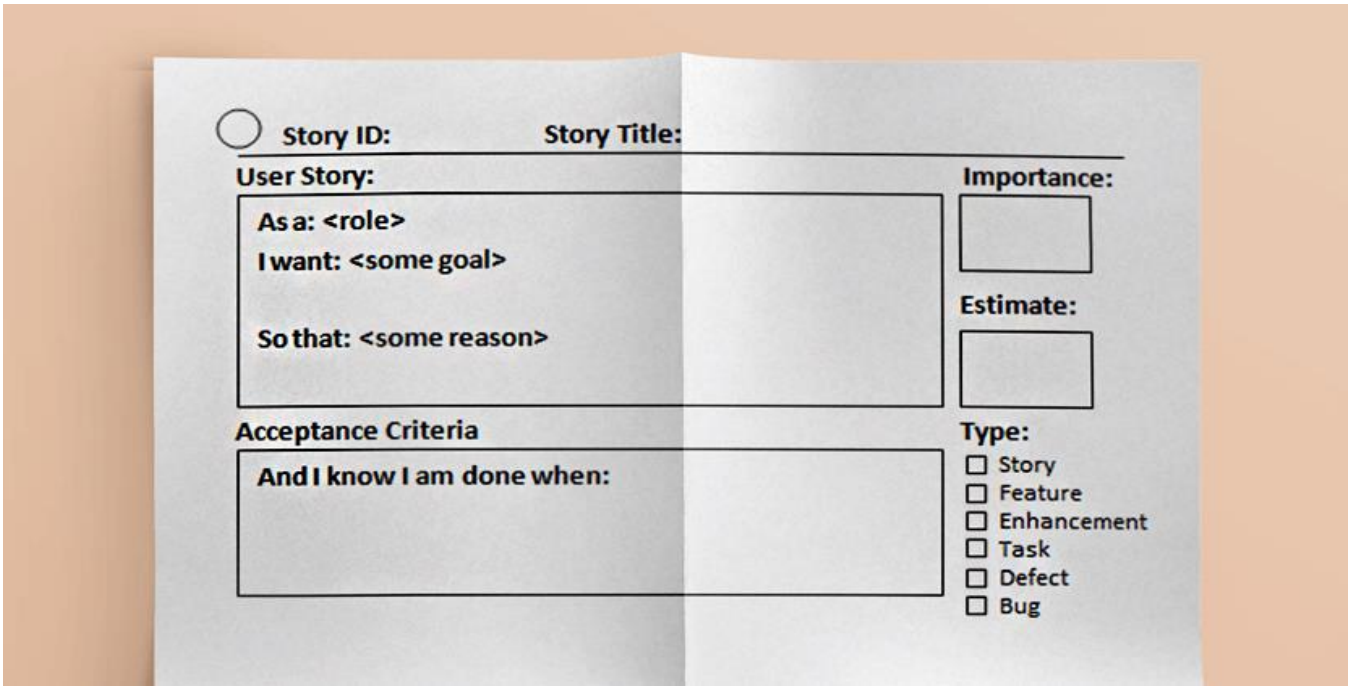
11.1. Get valuable ideas and feedback from client, end users and various other sources

Based upon the product vision shared by the client, the Agile team invites the stakeholders, technical team, end users, marketing and sales team, and management personnel to brainstorming sessions with an objective to identify market requirements and spot end user needs. Data is also gathered from various other channels such as usability testing, product usage data, emails, forums and more if possible.



11.2. List the ideas in the form of user stories in the product backlog

Ideas and suggestions collected from people and other sources are thought over and short listed to include only valuable suggestions which have a certain importance in terms of how useful they are to people using the product. The requirements are documented using a "As an , I want to , So that " format to specify the role (Who wants the feature or functionality), proposed activity (to do what) and the result or objective (to achieve what). This particular form of documenting the client/end user requirements is known as “user story” creation. A common list of all features and functionality required to build the product is prepared. This list is known as the “product backlog”.



11.3. Detail each user story and prioritize its importance

It can be difficult to specify the technical aspects and the particulars of each and every user story at the time of its creation. The reasons could be many. At times teams might feel a particular features is important and should be included in the backlog but may need further participation from end users to get the required details. In such cases, a user story is created for the particular feature and included in the backlog but its details may be kept blank. Later on when proper details are made available to the team, it updates the particular story by filling up the blanks with relevant info.

Agile focuses upon timed and sustained delivery of business value at the end of each product incremental cycle. Important stories having high business values are placed at the top of the product backlog while less important ones are arranged below them. Stories in the bottom have the least value. At the beginning of each new development cycle, few valuable stories are collected from the top of the backlog and made ready for the upcoming sprint. Stories are always collected from the top to ensure that maximum value is delivered to the client from each sprint.

12. Requirement gathering in traditional method versus Agile

TRADITIONAL	AGILE
1. Very hard to change requirements once defined and documented.	1. Existing features can be redeveloped/upgraded to add working functionality.
2. Product requirements are gathered only when the project starts.	2. Client, stakeholders and end users can demand new features and functionality at any time – Even late in the development cycle.
3. Very difficult for teams to adapt to new technologies since the scope of work and development time is fixed.	3. Teams can adapt to new technologies, techniques and tools as and when required if existing technology cannot deliver the required work quality.
4. The business value of the project is ascertained and fixed at the time of project inception.	4. New features can be added to the product in subsequent releases to enhance the business value of the project.
5. Business value can be delivered only when the entire project is completed and deployed.	5. Business value is delivered on a consistent basis to the client at regular intervals of time through the sprint iterations.
6. All feature requirements stated in the project have to be developed.	6. The requirements list (product backlog) can keep on increasing over time in the project but only selected and valuable features are developed in the project. Other less important features need not be developed.