Agile Plus – Comprehensive model for Software Development

Amit Juyal  Umesh Kumar Tiwari  Lata Nautiyal  Shashidhar G. Koolagudi
Assistant Professor  Assistant Professor  Assistant Professor  Professor
Graphic Era University,  Graphic Era University,  Graphic Era University,  Graphic Era University,
Uttarakhand- India  Uttarakhand- India  Uttarakhand- India  Uttarakhand- India
{amitjuyal26,umeshtiwari22,lata.nautiyal1903,koolagudi}@gmail.com

Abstract

In today’s changing environment the main priority of a customer is software that executes 100% requirements of customer. In addition there should be some scope so that existing software must adopt future requirements too, that is, it should be flexible enough that can accommodate future requirements, within budget and timely delivered. Traditional Agile models seem flexible but practically it’s hard to achieve their documented guidelines. Here in this paper I am proposing a new, flexible, and customer-oriented Agile based model, Agile Plus Plus, which includes the capability of traditional Agile models as well as methods to provide fast and instant deliverable. I am also explaining types of requirements of the client or of end user.

Keywords
Agile, Agile Plus Plus, Traditional Software Model

1. Introduction

Agile software development is a software development approach which states that there should be regular interaction with customer in each phase of software life cycle. In Agile, customer satisfaction is the highest priority for success of project. In Agile, the process of development of software is done with changing requirements of user and regular communications between user and developer[1].

2. Problem Definition

Agile model is mainly focus on quick delivery of iteration on regular basis .There is low level of testing is done in this model, less emphasis on non-functional requirements, low level of design, and coding is done without testing ,contractual difficulties are another significant issue relating to agile software development methods .Traditional contracts have been based on conventional models. In which customer gives all requirements at the starting of the project, buyer of the software may state exactly what is required, and if all the specified functions are not provided, then the contract is not met. How does this fit with an agile approach in which we try to provide fixed length iterations with variable functionality (which is determined based on current requirements, etc.)?

3. Agile Based Software Development

In Agile Based development customer regularly interacts with developer in the beginning of iteration (working code) and finishing of every iteration as shown in figure 1. As the project start, customer gives some requirements to developer, developing team (generally of seven to eight people) start with these requirements and develop working code and deliver to customer. Customer use it and response to developer with feedback and some additional functionality. Feedbacks give some other requirements and ideas to improve the functionality of a system. These requirements are added to the existing syste[3]m and deliver next iteration to the developer.

The main focus of Agile is on customer’s requirements. So there is need in Agile to identify correct and necessary requirements. Requirements can be of three types [4]

Normal Requirements:-
How the system will behave after deliver final product what are the objectives and goals of final product. All these specifications, which is given by a customer during meeting with development teams, are normal requirements.
Expected requirements:-
These are the fundamental requirements of the system that is not directly stated by customer but their absence can be the reasons of dissatisfaction of customer for example ease of human machine interaction, and ease of software installation.

Exiting requirements:-
These are the feature that customer never tells to the developer but their presence make the customer very satisfy.

In some cases there may be a chance when customer cannot express their actual requirements but they can know their requirements when they use working code (iteration) or demo release by developers. This problem may cause when customer is not able to give requirement in advance. Agile mainly focus on requirement of customers and misunderstanding or misinterpreted may cause failure of project or system, late in delivery or unreliable.

For the success of project there should be efficient communication or interaction is required between customer and developer because agile method works with unstable requirement to develop small iterations of final product.

Communication between developer and customer is mainly classified in two categories
1. When both developer and customer are in same location.
2. When both are in different sites.
Obviously first one has highest priority. When both developer and customer are in same location then interaction become easy because communication is done through face to face and on immediate feedback. Fact to face communication helps both to the customer and developer. In developer’s case, face to face communication helps in identifying the actual requirements of user because there are no technical limitations between both. Developer can see the expression of face of the customer this will identify those requirements which are not directly give by customer. In Customer’s case, face to face communication make ease for customer to tell whatever in his/her mind to developer because there is no technical limitations for example in case of email requirement are more precise. Immediate feedback from customers helps in corrections and modifications in existing systems. But the main problem in this is that it is very difficult to the customer to remain present all the time in on site because of various personal reasons of customer. When both are in different locations interactions is done through various methods such as through video conferencing, audio (telephone), and email. In this type of communication customer can directly communicate to the developer using various technology while both are at different places. The main problem in this is that if there is some failure of technology between them, than the communication can be interrupted this may cause late delivery of project, over budget of project. Another problem in this, if the developer cannot see the expression of the client then it is hard to the developing team to decide what are the actual requirements of the user if the interaction is done through telephone or email. But the main advantage in this is that customer does not need to physically present at location of developing team, this may save the time of customer.

4. Suggestions for Agile method
1. Some Planning is necessary:- As we know that agile is mainly depends upon requirements so some planning is necessary to identify how the project built.
2. Testing should be done earlier will help in finding problem earlier.
3. There should be some cost estimation tool or model
4. There must be finite number of iterations
5. Requirements should be freeze at some level so that development of software goes to next step of development.

6. There should be some non functional requirements

7. There should be plan or solution for future requirement (performance, reliability, exception handling and functionality)

5. Differences between Agile and traditional models.

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Agile</th>
<th>Conventional models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Objective</td>
<td>Satisfying the changing requirement of customer by frequently releasing iterations(working code)</td>
<td>Satisfying the customer by providing complete software through long software life cycle.</td>
</tr>
<tr>
<td>Involvement of customer</td>
<td>Very high, regular interaction between customer and developer.</td>
<td>Low, rarely in once in a month or more.</td>
</tr>
<tr>
<td>Working environment</td>
<td>Very fast , mainly focus on releasing small working codes with changing requirements</td>
<td>Mainly focus on deliver final project</td>
</tr>
<tr>
<td>Documentation</td>
<td>Rarely</td>
<td>Very high</td>
</tr>
<tr>
<td>Testing</td>
<td>Only unit test</td>
<td>Complete testing in developer site and customer site.</td>
</tr>
<tr>
<td>Communications</td>
<td>Face to face, video conferencing, audio and email but on regular basis</td>
<td>Rarely communicate</td>
</tr>
<tr>
<td>Team</td>
<td>Generally small teams of 7-8 people.</td>
<td>Variable size depends upon project.</td>
</tr>
</tbody>
</table>

6. Proposed Agile Plus Plus Development Model (Agile with some additional feature)

**Agile Plus Plus** is new method to develop software more frequently and error free. It is an iterative software development model that emphasizes on not only extremely short life cycle but also testing with each and every step. In Agile Plus Plus requirements are gather and based on requirements iterations are deliver on regular intervals. But in Agile Plus Plus each iteration must go through following phases.

Agile methodology fastest way to develop software that mean listen, code whatever listen and deliver it .The software is developed without too much design and plan. Agile is best fit for small projects but it is not too much suitable for large and complex projects. Because in large and complex project if requirements are keep changing then it is very difficult to modify earlier iterations.

Agile Plus Plus is a software development model. It is an extension of Agile model which aim is “Code it and fix it “where as in Agile plus Plus “Code it with design, testing and fix it with maintenance” . Agile Plus Plus similar to Agile where quick delivery of working software with changing requirements is done. In Agile Plus Plus working software is a complete software i.e. requirements are transforms into the software after going through different phases of software development as shown in the figure 2. In this figure, customer gives requirements to the developing team. These requirements go through different phases of Agile Plus Plus development model as shown in the figure and working product is deliver to the customer. Customer uses it and reply with feedback and additional requirement to the developing team. In Agile Plus Plus a proper planning (Which is not in Agile) is done before actually implement the product. Cost Estimation tool is used to decide the overall budget. Testing is a very important phase in Agile Plus Plus. All the phases of Agile Plus Plus is describe follow.

7. Phases of Agile Plus Plus

**Requirement with non functional requirements:-**

In Agile Plus Plus requirements are taken from the customer regularly via face to face communication, video conferencing, e-mail, and audio mail etc. Then requirements are prioritized; only essential requirements are first implemented. Essential requirements are those requirements which are necessary for the correct functionality of the system. To prioritize requirements helps in saving time and cost because developer has to focus only to implement only essential features. Now in Agile Plus Plus this time can be consumed to adds non-functional requirements with customer’s requirements. Non functional requirements provide reliability, security, portability and other
additional features that satisfies customer. Because customer got the working code with additional features without paying extra amount. By adding non functional requirements, makes customer more satisfaction.

Figure 2:- Agile Plus Plus Software Development Model

**I Phase:-Analysis with cost estimation:-**

At the time of requirement gathering anything which comes in the mind of customer becomes requirements of customer. But it may be possible that some requirement is not feasible, some are irrelevant and some are not possible due to budget of project, time or due to any other technical reason. So at this phase feasibility study is done which helps to find correct and high priority requirements. If the developing team starts the work without any analysis of requirement than there may be a chance of project failure because it may be possible that they implement requirements which are not very important for the necessary functionality of the project.

In Agile requirements are changes rapidly so it is very difficult to estimate the cost of the product. To overcome this problem “A Cost Estimation Model” is used to estimate the cost of the project with changing requirements.

The main idea of Agile is “Code it and Fix it”. There no provision for software project planning. Which is an important phase in Agile plus plus. The purpose of Planning is to establish reasonable plans for performing the software engineering and for managing the software project. Software Project Planning involves developing estimates for the work to be performed, establishing the necessary commitments, and defining the plan to perform the work.

The software planning process includes steps to estimate the size of the software work products and the resources needed, produce a schedule, identify and assess software risks, and negotiate commitments. Iterating through these steps may be necessary to establish the plan for the software project (i.e., the software development plan). Software estimates are documented for use in planning and tracking the software project.

**II Phase:-Design with verification:**

Design is a meaningful pictorial representation of something that is to be built. A good design gives the right direction for the development of software project. Software design is the process which translates requirements into the blueprint for developing software. In A++ quick design is done to develop right product. In A++ Design is done with verification for develop right product.

**III Phase:-Coding with testing**

After the design, it is translated into code. Developers take the flowcharts, UML diagrams, and the other design documents and translate them into code, component by component, module by module or object by object. Each component is unit-tested on its own and a code review should be done. Usually the integration is done at the end of the coding phase or at the beginning of the test phase. This is often the time for surprises, unforeseen events and regression, as things may not fit or work together as planned, and the team may have to go back to the design phase to make the appropriate changes.

Depending on how tough the schedule was, the project may get in trouble right at this point, as the plan was to integrate the components to form a perfectly working system. But now there is a delay due to the integration issues. The risk of a big surprise at the end can be reduced by introducing different milestones during the coding phase, at which point in time the system is
integrated and needs to provide a certain level of functionality. These milestones should ideally be followed by a set of tests to verify that the functionality works correctly.

After the developers have declared that they are done with the coding phase, what is often also called DCUT (Design Code Unit Test), the different units and components are put together into an integrated system. This is usually the time where the project goes south due to many surprises and integration issues. This could be because developers have made mistakes during coding, or that a component A behaves slightly differently than another component B expected it to and now they simply don’t work together.

It is usually helpful if a smaller number of testers start with some basic test cases and, as soon as those can be successfully executed, the majority of the test team should join the phase. The goal of the test phase is to identify bugs in the software before it is released to the end user.

With traditional projects, the developer to tester ratio is more in the area of 1:1–1:3 [14], with agile projects it is more in the area of 1:3–1:5 [15], due to the fact that more testing is required by the development teams and the traditional test teams are now more focused on regression testing and verification of additional platforms.

IV Phase:-Planning and Tracking Test Progress
The first step in planning the test is to get a good overview of what the features and functions of the solutions are. Based on this list of features and functions, an overall test plan needs to be developed for the overall project. This test plan should contain all the different test phases for the project. Usually a project has the following steps in testing:

**Integration & function verification test:**
This is usually the first test phase after the developers say they are done with coding and unit test. The goal of this phase is to ensure that the functions work as expected and that the overall solution is installable and fits together. This test phase usually also contains any required accessibility testing.

**System verification test:**
Here the complete solution is tested as a whole in complex customer-like environments under load. When software comes in the user’s environment, it will get interaction with the four environmental components:
1. Human: End-User
2. Hardware: On which it is going to install or implement, and through which it will get input and output.
3. Software: Operating system and other software.
4. External File systems.

The goal is to verify that the solution functions for several days or weeks without issues such as running out of memory.

**Performance verification test:**
The goal is to verify that the solution meets the performance requirements, especially with regard to throughput and scalability.

**Acceptance test:**
Solution development projects for a particular customer often end with an acceptance test phase during that the customer is validating if the solution meets his or her specifications and exceptions, before he or she agrees to close the project. These different test phases are then usually planned separately, and test scenarios are developed for each of them. But it is important that there is an overarching test plan covering all the phases that ensures that all areas are covered and on the other hand also ensures that there are no duplicate test cases. There are often specialized test teams with skills in each of these test areas. The team then estimates the effort it takes to complete these scenarios. This estimate should also include the time it may take to reapply new builds and retry the test after defects were fixed.

Often the test teams don’t directly assign efforts in person days or person weeks, but translated them into abstract units like points.

The progress of a test can be tracked and managed using a *s-s-curve*. The start of a test is usually slow until the basic problems are resolved, like, for example, installation or configuration problems that may block several test team members from carrying out their functional test cases.

The test teams reach their highest productivity as soon as these problems are resolved and all the different test cases can be executed in parallel. Towards the end of the test, the number of blocking defects becomes
smaller and there is only a small number of remaining test cases that are still failing. Therefore the s-curve is flattening out.

The test teams usually differentiate between Actual and attempted test cases, meaning the team started to test the respective test case, and completed test cases, which are those that finished without remaining bugs.

7. Conclusion

Agile Pus plus is a software development model that provides a right direction towards the software development. This model mainly focuses on customer satisfaction via releasing regular iteration on regular intervals. Modelling is a skill that all developers must gain to be effective. Agile Modelling (AM) defines a collection of values, principles, and practices which describe how to streamline your modelling and documentation efforts. Modelling can easily become an effective and high-value activity if you choose to make it so; unfortunately many organizations choose to make it a bureaucratic and documentation-centric activity which most developers find intolerable.

10. References