e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

AGILE IN SOFTWARE QUALITY

Sejal G Agarwal*1, Piyush Jindal*2

*1St. Cloud State University, Minnesota.

*2BITS Pilani, Minnesota.

DOI : https://www.doi.org/10.56726/IRJMETS60178

ABSTRACT

Agile has become the mindset that has changed the way to deliver value to stakeholders. Software quality has been stressed during the entire development cycle that it takes half the time and effort. The goal of this paper is to find the impact of Agile on Software Quality and determine if there any particular techniques have been implemented to ensure the quality of the software. What are those different techniques that have been implemented and study the impact of those techniques? This paper focuses on Test Driven Development and Refactoring techniques and how that has impacted the quality.

Keywords: Software Quality, Agile Methodology, Agile Framework.

I. INTRODUCTION

Software quality is an exciting field in which the project team ensures the degree to which a system meets the requirement specification as well as the need and expectations of users or stakeholders. Lee, Ming-Chang. (2014) [6] in the British Council of applied science and technology said that software quality assurance is a formal process for evaluating and documenting the quality of the work products during each stage of the software development lifecycle. Alebebisat, Farhan & Alhalhouli, Zaid & Alshabatat, Tamara & Alrawashdeh, T.I. (2018) [8] in a review of literature on Software quality said that the software development industry considers quality a crucial factor in its development. The quality model defined in ISO/IEC 25010 [8] comprises the eight quality characteristics shown in the following figure [8]:

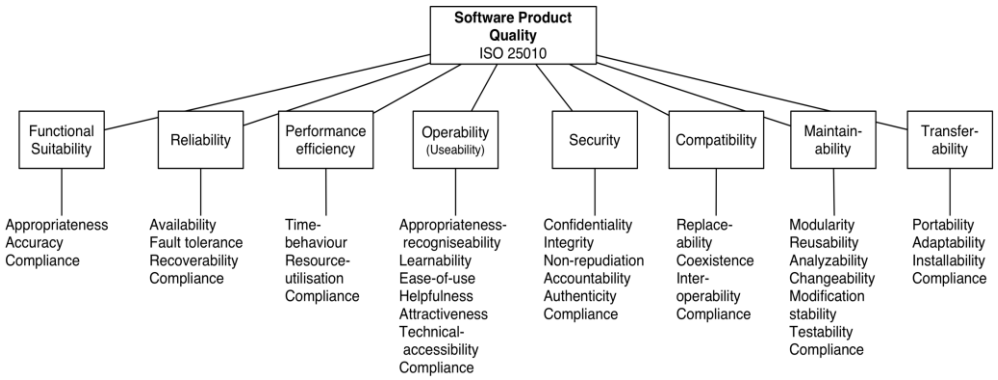

Figure 1: ISO 25010 Software Quality Characteristics

Agile was born when a group of software development groups joined hands and created Agile alliance in Feb 2001. This group has changed the foundation of software development and made it more practical as compared to process oriented. Agile alliance listed 12 basic principles [2] to guide teams. Due to the flexible and dynamic nature of Agile methods, it is being considered as best fit for fast growing software industry so using Agile methodology is a highly collaborative process where testers, developers, business analyst, and the project stakeholders actively work together on a just-in-time basis to understand the project and its domain. Agile is a framework and there are many methods that follow this framework as shown in given below figure [10].

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

Agile Methodology	Emphasis	Founder(s)
Extreme Programming (XP)	Efficiency, customer focus and feedback, and quality	Kent Beck
Scrum	Teaming, organizing work	Jeff Sutherland and Ken Schwaber
Feature-Driven Development	Iterative development of user-focused features	Jeff De Luca
Dynamic Systems Development Method (DSDM)	Structured approach to rapid development, collection of best practices	DSDM Consortium
Lean Software Development	Eliminate work that does not create customer value	Mary and Tom Poppendieck
Kanban Method	Visualize and manage workflow, just-in-time development	David J. Anderson
Crystal Family	People, communication, process rigor maps to product and organizational dynamics	Alistair Cockburn

Figure 2: Types of Agile Methodology

One of the principles in Agile Manifesto (Agile Alliance, 2001) states that Agile is all about meeting the customer changing requirements till the product deployment hence there is a definite need to understand the effectiveness of using Agile methodology in quality assurance. Brüggemann, Stefan. (2013) [4] in Agile Software Quality Assurance stated that Agile software development is performed in many projects inside and outside the space domain.

The technical heart of agile software development are iterations of two or three weeks in which all classical activities like requirements engineering, design, implementation, test, and review are performed and where working software is created. These short cycles lead to more transparency of the process and to early feedback gained by frequent user integration. Quality assurance (QA) benefits from these aspects by directly integrating with the team and by better measurement and analysis. Ongoing Verification and Validation is performed as a task of iterative acceptance through iteration reviews with customer involvement. Quality assurance activities now have to be adapted; in particular, they have to be shifted from a few major inspection points to the establishment of ongoing automated quality gates in the frame of continuous integration.

II. SOFTWARE DEVELOPMENT METHODOLOGIES & COMPARISON

	Software Process Model	Advantages	Disadvantages
Plan Driven	• Waterfall • Incremental Development • Iterative development • Spiral Development • Prototype Model • Rapid Application Development	• Suitable for large systems and teams. • Handles highly critical systems effectively. • Appropriate for stable development environment. • Require experienced personnel at the beginning. • Success achieved through structure and order.	• Longer length in each iteration or increment. • Cannot accommodate changes any time. • Lack of user involvement throughout the life cycle of the product. • Costly for the dynamic development environment. • Assume that, future changes will not occur.
Agile	• Scrum model • Extreme Programming (XP) • Dynamic System Development Method • Kanban • Feature Driven Development	• Suitable for small to medium systems and teams. • Can accommodate changes at any time. • Effective for the dynamic development environment. • Required expert agile personnel throughout the life cycle. • Success achieved through freedom and chaos.	• Not suitable for large systems (except FDD). • Shorter length in each iteration. • Can accommodate changes at any time. • Costly for the stable development environment. • Assume that, frequent future changes will occur.

Figure 3: Advantages and Disadvantages

Many Software development methodologies have been examined by managers to choose the one which works best for the project that is being worked on. Two methodologies are Waterfall and Agile. The waterfall method is the linear approach to software development and all the stages are executed in sequence from Requirement definition till delivery of the product whereas Agile method is an iterative approach to development emphasizing the rapid delivery of an application in highest value component first. Lu Bauer in Agile

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

Methodology and System Analysis [1] has identified the agile methodological instruments regarding the software development lifecycle, from the waterfall model to the agile model. The author also explained the importance and idea of using agile methods in software development projects and inclination toward the adoption and development of agile methods. Given below figure [9] was part of the hybrid study which shows the difference of plan vs agile models and their advantage and disadvantages.

The differentiator of agile from other methodologies to software development is the focus on team collaboration and self-organization. The agile manifesto emphasizes on responding to change as compared to following the plan but always raises the question of the software quality. Huo, Ming & Verner, June & Zhu, Liming & Ali Babar, Muhammad. (2004) [4] compared the waterfall model with agile processes to show how agile methods achieve software quality under time pressure and in an unstable requirements environment, i.e. author analyze agile software quality assurance. Authors present a detailed waterfall model showing its software quality support processes. Authors then show the quality practices that agile methods have integrated into their processes. This allows them to answer the question "Can agile methods ensure quality even though they develop software faster and can handle unstable requirements?"

III. REQUIREMENT CHANGES & QUALITY

Quality as per ISO 9000 [10] is ensuring that the stated needs that are specified by the customer or stakeholders have been met entirely. Software experts think that real stakeholder requirements come up during the creation of the system. During the development of the part of the whole system, stakeholders interact with the part of the system that is being created, they can revise the requirements by adding or removing or modifying some of the existing ones. This can't be achieved using the traditional approach and requires the new development approach to adopt for the change. Agile methods can accommodate this new approach as they comprise of iteration and incrementality. The iterative and incrementality leads to the adoption of the changing requirements and active participation of stakeholders. Agile also spread decision making throughout the software development life cycle. During various studies that are conducted in 2001 and 2002, it was clearly shown that 80% failure was due to the use of waterfall models and upfront requirement and approximately 60% of the requirement was never used or rarely used after development.

Quality of the whole system depends upon the task of completing the actual stakeholders' requirements. Agile provides the right framework which provides the successful delivery of a complete system which is the number one criterion for quality assurance. Institute of Electrical and Electronic Engineers (IEEE) defines quality as "the degree to which a system, component or process meets customer or user needs or expectations."

As business is moving toward the approach of an agile framework, Quality assurance activities now have to be adapted; in particular, they have to be shifted from few major inspection points to the establishment of ongoing automated quality gates in the frame of continuous integration. QA personnel now can proactively assure product quality with respect to coding and quality standards. Brüggemann, Stefan. (2013) [5] performed the agile software development iteration of two or three weeks in many projects inside and outside the space domain. These short cycles lead to more transparency of the process and to early feedback gained by frequent user integration. Quality assurance (QA) benefits from these aspects by directly integrating with the team and by better measurement and analysis. Ongoing Verification and Validation is performed as a task of iterative acceptance through iteration reviews with customer involvement.

Extreme Programming (XP) makes the quality as the highest priority of the Software development life cycle using unit testing defined as part of the acceptance testing which is part of the stories that have been created and functional testing, which is achieved by continuous involvement of the stakeholders while creating the stories for the requirement.

“People and interaction over process and tools" is one of the four agile manifestoes [2] value. Stakeholder interaction is most important for the success of software development. A successful relationship with stakeholder goes a long way as compared to process or terms and conditions defined within the contract. Other value given in agile manifesto [2] states that "Responding to change over following the plan", which means that development team should accept the changes in requirement and changes the validation and verification as per the experiences that have been gained from the change in requirement.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

Agile teams don't maintain the traditional documentation practices hence doesn't have the records of documentation. These teams use a review meeting to showcase what has been done as part of the current iteration. This reduces the time spent on creating heavy documentation which is hardly being used or read in the future. The team which practice traditional practices requires document for each and every requirement in the form of word/excel or presentation format. In the current world, it can be of any form including videos, stories, whiteboard discussion, meeting recordings.

IV. TEST DRIVEN DEVELOPMENT

Extreme Programming is an agile software development framework designed to improve the quality of software and its ability to properly adapt to the changing needs of the customer or client. Yasvi, Maleeha. (2019) [15] in their study of Review on Extreme Programming-XP stated that Extreme programming is an iterative software development methodology which aims to produce higher quality software and helps in providing an optimal solution. Extreme Programming differs from other software development methodologies as it focuses more on adaptability and responsiveness to the changing customer requirements. By using extreme programming as a software development methodology, better results have been obtained in software development. Test-driven development (TDD) is testing first connect of Extreme programming (XP), which is one of agile methodology. TDD always emphasized on test-first approach and this is what makes it different from traditional approaches where testing has been done later in the cycle after development has been completed. Figure 4 shows the difference between the traditional approach vs TDD approach [13].

Yenduri, Sumanth & Perkins, Arlene. (2006) [15] in their paper conducted an experimental study over two groups of students comprising of undergraduate students (seniors) who develop software using the conventional way of performing unit testing after development and also by extracting test cases before implementation as in Agile Programming. Both groups developed the same software using an incremental and iterative approach. The results showed that the software had a smaller number of faults when developed using Agile Programming. Also, the quality of the software was better and the productivity increased.

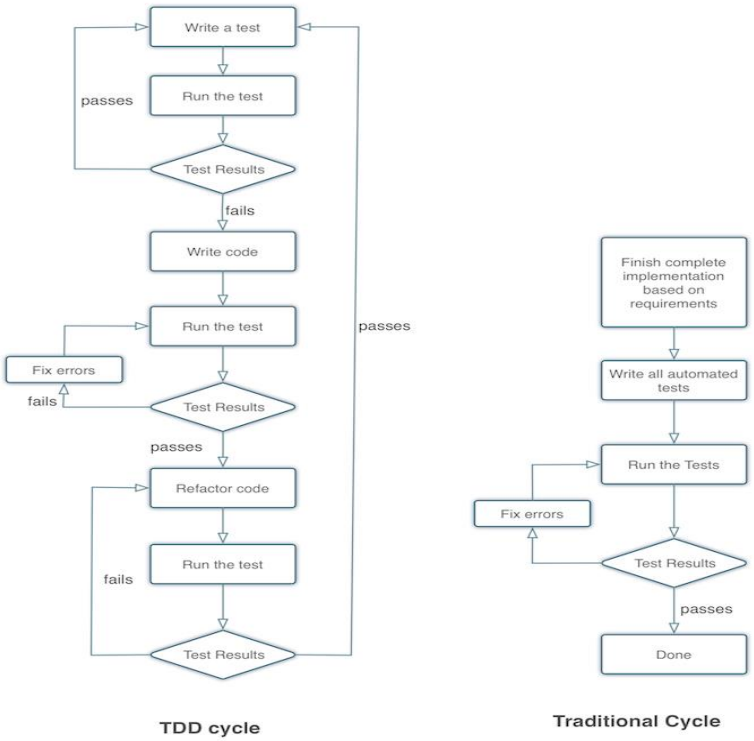

Figure 4: Traditional Cycle vs TDD Cycle

Table 1 [15] clearly states that the number of written cases were more whilst decrease in faults detected by SQA during unit testing, integration testing, and acceptance testing. TDD also shows the increase in productivity of the team as the number of hours spent was less than the traditional approach.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:07/July-2024 Impact Factor- 7.868 www.irjmets.com

Table 1: Metrics Recorded after the experiment

	TDD	Traditional Approach
Number of test cases written	629	211
Total Person Hour Spent	928	1245

V. REFACTORING IN AGILE

As per traditional followers, Agile methodology follows quick and often performed without a fixed plan which can lead to bad code, wrong architecture, design mistakes, anomalies in software but organization or teams that follow the agile methodology usually follow.

Refactoring is one of the key techniques as part of extreme programming and other agile methodology where the developer will change the internal structure of the code without impacting or changing the external functionality of the code. Its sole purpose is to improve the code quality by reducing the cost and maintenance effort of software. It also has an effect on quality attributes like understandability, productivity, flexibility, reusability. Refactoring leads to the complete picture of the product for the team which is beneficial for team understanding and validating the scenarios for integration testing. It also increases the code readability for the team. Code refactoring makes the code more organized which leads to low maintenance of the product, one of the quality attribute.

Kumari, Noble & Saha, Anju. (2014) [14] in their study stated that software quality is an important issue in the development of a successful software application. Many methods have been applied to improve software quality. Refactoring is one of those methods. In their paper, they applied fourteen refactoring methods and noticed that they effect randomly on different software quality attributes. They have concluded that refactoring improves the quality of software but developers need to look for the particular refactoring method for the desired quality attribute.

VI. CONCLUSION

Agile has led to the delivery of software delivery in smaller cycles with continuous feedback in place. This has been mistaken with that teams who follow agile doesn't follow the plan which leads to less documentation and incomplete execution of the work. During this research, first we have looked at different quality attributes and then different types of agile methodology then shared the different type of software development methodology and comparison between them. With continuous changes in requirement needs, there is always changing what stakeholder need which leads to the change in how can we evaluate the quality of the software as functional and non-functional requirements are derived using the requirements which are continuously changing and making the change in how can we evaluate the quality. Agile helps with the changing requirement and ensures that we have accommodated all the new requirements as we proceed further with delivering in small chunks. There have also been concerns about poor documentation as there is no particular work or excel document created as part of agile. Agile focus on interaction over documentation but in the current world, documentation has been changed to different ways and it doesn't have to be just word or excel document, we can use anything like video, stories or white boarding or brainstorming session to be the document. Extreme programming (XP) has Test-driven development (TDD), which is completely focused on testing first and one studies has been shared that shows how TDD increases the quality of the software as compared to traditional cycle then we have discussed about refactoring, which is also part of extreme programming, that has also improved the quality of the code and has increased many non-functional attributes of software quality. This proves that agile methodology has no negative impact on software quality, whereas there are different techniques that have been introduced to ensure the higher quality of the software. There are many experimental studies that have been done in the past, which proves that TDD and Refactoring have positively impacted the attributes of software quality. This research concludes that agile doesn’t compromise with software quality and there are many techniques implemented to even achieve greater quality.

VII. REFERENCES

[1] Agile Methodology and Software analysis –

http://www.umsl.edu/~sauterv/analysis/Agile%20Methodology%20and%20System%20Analysis.htm

[2] Agile Manifesto and 12 Principles –

https://moodle2016-17.ua.es/moodle/pluginfile.php/80324/mod_resource/content/2/agile-manifesto.pdf and http://agilemanifesto.org/principles.html

[3] Software Quality and Agile Methods- Ming Huo, June Verner, Liming Zhu, Muhammad Ali Babar National ICT Australia Ltd. and University of New South Wales, Australia

[4] Brüggemann, Stefan. (2013). Agile Software Quality Assurance.

[5] R. Agrawal, D. Singh, and A. Sharma, "Prioritizing and optimizing risk factors in agile software development," 2016 Ninth International Conference on Contemporary Computing (IC3), Noida, 2016, pp. 1-7. doi: 10.1109/IC3.2016.7880232 URL:

http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7880232&isnumber=7880185

[6] Lee, Ming-Chang. (2014). Software quality factors and software quality metrics to enhance software quality assurance. British Journal of Applied science & Technology. 4. 3069-3095.

[7] Alebebisat, Farhan & Alhalhouli, Zaid & Alshabatat, Tamara & Alrawashdeh, T.I. (2018). Review of Literature on Software Quality. 8. 32-42.

[8] https://iso25000.com/index.php/en/iso-25000-standards/iso-25010

[9] Rahim, Md Shamsur & Chowdhury, AZM & Nandi, Dip & Rahman, Mashiour & Hakim, Shahadatul. (2018). ScrumFall: A Hybrid Software Process Model. International Journal of Information Technology and Computer Science. 10. 41-48. 10.5815/ijitcs.2018.12.06.

[10] Introduction to agile methods by Kristi Runyan, Sondra Ashmore Ph.D. Chapter 3

[11] https://medium.com/innodev/agile-development-for-dummies-dd161da253c7

[12] http://derekbarber.ca/blog/2012/03/27/why-test-driven-development/

[13] Yenduri, Sumanth & Perkins, Arlene. (2006). Impact of Using Test-Driven Development: A Case Study. 126-129

[14] Kumari, Noble & Saha, Anju. (2014). Effect of Refactoring on Software Quality. Computer Science & Information Technology. 4. 37-46. 10.5121/csit.2014.4505.

[15] Yasvi, Maleeha. (2019). Review On Extreme Programming-XP.