INTERNATIONAL JOURNAL OF ACADEMIC RESEARCH IN BUSINESS & SOCIAL SCIENCES

Vol 9, Issue 12, (2019) E-ISSN: 2222-6990

Critical Analysis of Software Development Methodologies based on Project Risk Management

Kamran Javed[1], Asif Hussain Khan[2], Lubna Tubbassum[3]

[1]PhD Scholar, IoT Engineering, Hohai University, Nanjing, China, [2]PhD Scholar, Dpt. of CS, Nanjing University of Science & Technology, China, [3]Banking Service Manager, Allied Bank Ltd. Bahawalpur, Pakistan

Email: kamran.javed86@hotmail.com, asifkhan00992@gmail.com, lubna852008@hotmail.com

To Link this Article: http://dx.doi.org/10.6007/IJARBSS/v9-i12/6720 DOI:10.6007/IJARBSS/v9-i12/6720

Published Date: 24 December 2019

Abstract

Computing is always the topic of interest for the researchers as many of its aspects are inviting researchers to explore them. One such aspect is software development methodologies for developing the software which can work closely to the need of client and nature of the requirement. The one important but neglected aspect on which these methodologies can be analyzed is project risk management and ability of the certain methodologies to assess and mitigate the risks in the project of software development. Software development methodologies are continuously going through evolution process and based on different variables new methodologies are being introduced. From conventional and tradition software development methods to the modern development tools, methodologies are differentiated based on their characteristics. In this research, a critical analysis of the conventional and agile methodologies will be presented on the bases of risk assessment and mitigation.

Keywords: SDLC, Waterfall, Agile, Risk Analysis

Introduction

Today the number of software development methodologies in the industry of software engineering with their varying approaches towards the development of product and team management has emerged. The emergence of the models is where providing tendency to develop more optimized models, there it is also creating the numerous problems in the industry of software development(Bouquet et al., 2018). Each model presented in the best interest of elevated software development, possesses its own limitations and problems. The ease and also the problems in these methodologies are only realized and experienced by the software engineers while developing the software of different natures. It is merely possible that one methodology works for each project, as the requirements and nature of the software under development are different, so the traits of software development is natural to be varied in each project and software(Enríquez, Sánchez-Begínes, Domínguez-Mayo, García-García, & Escalona, 2019). These problems may be identified at the early stage of software development or may not be identified till the software delivery to the client. The robustness and reliability of the adopted software development model is ample in such cases, where failing to accumulate the risks in the project at early stage might bring up total or partial failure of the project which in turn will elevate the project cost, wastage of resources and deteriorating the efforts of the developers. Additionally, in the early models the client's input in the development process, the identification and analysis of the requirements, effective task breaking, periodic approvals and testing of the artifact and mitigation of risks was not accounted at all in the process(Eastman, 2018). In Late 1990, the development of adaptive software development models were introduced and the said problems start being addressed. The aim of this research is to provide the critical analysis of the software development models including early developed models and modern development tools based on risk management techniques used in each model. Moreover, the critical comparison of the degree of being adaptive, scale and scope of the each model, cost effectiveness and efforts of the developing team will also be presented. Along with this contribution, in this research the potential enhancements in the current software development models will also be suggested to make the model more robust and effective.

Consideration of SDLC in Organizational Sustainability

It has been thought that the previously available methodologies and partially currently available methodologies are not suitable and sustainable for the businesses and organizations. The first hand available SDLC methodologies are stuffed and making systems much faster, due to which the resource consumption is exponentially growing. In regard of the availability of the natural resources, they are very limited and thus making the SDLC methodologies less sustainable. Nevertheless, the software is driven by the business behind them and therefore remotely impacting the economic progression (Penzenstadler & Femmer, 2013). To introduce the sustainability in the software development industry it is necessary to justify the addition of the cost due to sustainability. For instance, the repute management of the company in the market that is basically is the duty of the business analysts, though, for software engineers developing the software, it has become necessary to understand the responsibility of the consequences occurs in the long term due to the design of the software. Here, the impact of software engineering and Software Development Life Cycle methodologies on the organizational sustainability will be discussed (Penzenstadler, 2013; Penzenstadler et al., 2014).

The characterization of the Sustainability in the software engineering area has its scope in multi dimensions such as domain dependant and domain independent is given below:

Characterization 1: In software development the sustainability is the capability to tolerate and undergo. Such as, how much software development is adhered to the standards and laws, how much the software is maintainable when the projects are of long span and what is its energy efficiency.

Characterization 2: concepts of the implementation of the SDLC techniques are denoted by Software engineering for the organizational sustainability. It facilitates in the sophistication of the domain dependent into submerse level and technical requirement for the software development and implementation of the system.

Characterization 3: Domain experts characterize the software System and surrounding application context and its sustainability. Heinberg axioms define the fact of the software system and organizational sustainability. For software system the constraints can be derived from the requirements by software engineers for the sustainability of the organizations.

Early Methods and Code & Fix

The software development methodologies were introduced in 1950s when needed for the effective development of the optimized software. The models proposed at that era were code driven methodologies of software development. The term code driven is not to be confused, as if the software development was not about the coding. The figure below will elaborate the term code driven methodology. This model was consists of only 3 stages, conceptual development, code and fix and release product.

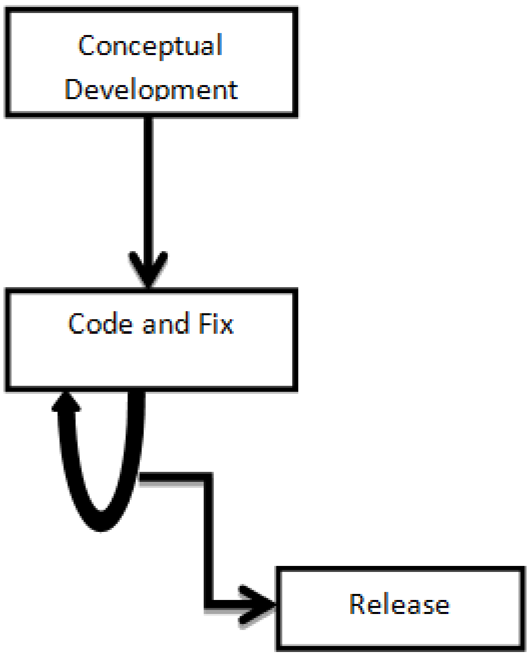

Figure 1 Code and Fix Model

The figure above illustrates that at the second stage of developing the software where the developers have to develop the code for the particular software is oriented towards the design and development. The developer team codes the software and tests it, then fixes the problems by rectifying the code and so on. In this era of software development no negotiation with clients was required, no requirements used to be accounted from the client or business side. The code and fix model was limited to the best of knowledge of developer and their understandings(Mohagheghi, Dehlen, & Neple, 2009).
The code and fix model received not very good reception and mocked due to its derided process of development cycle. However, it is most used model of all times, if taking in account the several small software applications developed each day. This model receives much criticism because it does not require that much requirement as of today to begin coding the software development. As shown in the figure above when something is developed it is then fixed and developed again until it starts working perfectly according to the initial requirement. This model cannot be implemented to the projects which are to be developed on the large scale, rather it will be good for small and routine projects for oneself use. The problem arises when the model is implemented to the project which is big, to be used on large scale and future modifications and extensions are expected. Rework on the coding and development is

a time consuming process and can also sabotage the system structure(Jiménez, Piattini, & Vizcaíno, 2009).
For instance, the project risk management is concerned; this model can put the whole project in the risk and might cause failure at the end not only in terms of budget but also team mismanagement and trust issues. This model does not support the configuration of management and software architectural plan. Though this model is good for making quick version of the software operations or may be just interface, but it is said that there is no more untidy and dangerous project development model does exist(Sohaib & Khan, 2010).

Water Fall Model

In the late 1960's as a result of evolution process another model was developed to meet the requirements of the software development and make the process of developing the software systematic. Waterfall model is one of the known and widely used development processes. **Royce** illustrated the phases of software development during his work on developing the software for spacecraft industry(van Casteren, 2017). **Royce** also elaborated the sequence of phases which when combined makes a software development process which is shown in the figure 1 below.
Regarding waterfall model, many researchers put forth number of ideas to improve the phases and design of the software development model, however, **Unhelkar**, in his studies indicated that in the waterfall model, every phase depends upon the results of the previous phase, which makes it sequentially dependent on the deliverables of the previous phase(Unhelkar, 2016).

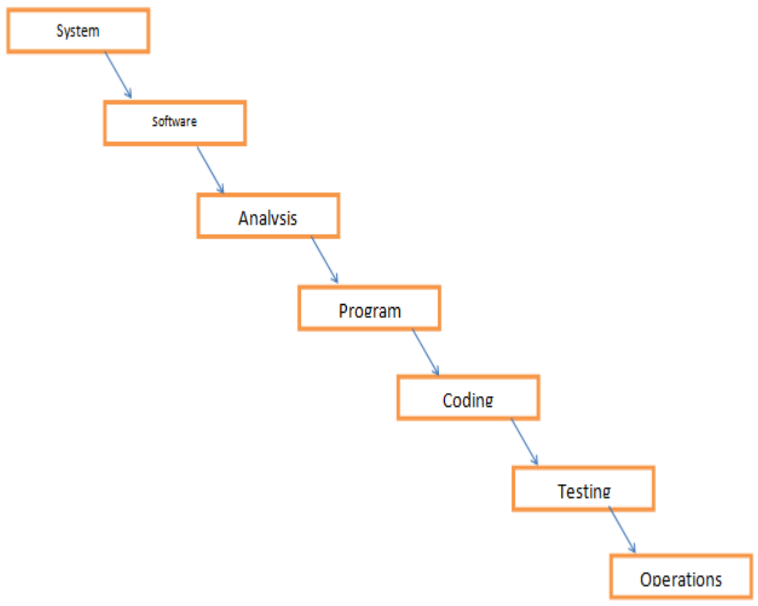

Figure 2 Water Fall SDLC

This means that without receiving the results from the last phase this model cannot move further. This holds back the whole model, for instance, if the program design is still to be discussed, negotiated and approved the developer team cannot start with the coding. Royce also discussed the possibility of the iteration between each phase, which means that ever successive step have the iterative relationship with the preceding phase. This is how the model can work progressively without freezing it on single phase. Each phase will work

parallel and the requirements from the clients can be adjusted during the iterations (Amjad et al., 2017).
However, there are some issues indicated by the researchers and the developers. These issues may also be contributed from other stake holders of the project. This model emphasizes on the initial and basic requirements of the project to be discussed and frozen in very early stage of development. In case the requirements of the project are not well informed to the developer team and not negotiated or understood among clients and the developers then the waterfall model will not be an appropriate method of software development(Gnatz, Marschall, Popp, Rausch, & Schwerin, 2002). Some other researcher also discussed in their researches that this model is costly and it requires so much efforts. This conviction is confirmed when certain things are bring to considerations like difficulty in initiating iteration and achieve it, problems in accepting changes and implanting them and documentation approvals in each phase and other problems which may rise at the end of the project. This might bring the project to a situation where many useless features may become part of the project compromising the basic requirements of the client(Petersen, Wohlin, & Baca, 2009). This in turn risk the project to be a failure and the risk involves in it regarding project characteristics and management may impact the performance of the business and also member of the developer team.

Doctrines and Rehearses of Agile Methodology
i) Principles of Agile Methods:
In software development life cycle among all other methodologies Agile Methodology stands alone due its distinct features. This methodology is based on the best experiences of modeling and documenting the software systems. Different values and principles worked together for developing the projects of software development. In the fast changing environment Agile methodology is more flexible and fits itself well.
According to researchers all the agile methodologies bear approximately identical characteristics in terms of project management and risk mitigation in software development. All agile methodologies emphasize on the interactions between client and developer team and among developer team as well. The communication among the stakeholders have non-trivial importance in the development of the software, as it gives the feeling of the ownership and mitigate the risk of project failure due to non-compliance of the final product with the client requirements. Moreover, the mediating artifacts in software development are tends to be resource intensive in the conventional methodologies. If the developer team does not make itself to cope up with the resources exhaustion then the whole project put to the risk of failure. But in Agile methodology the reduction of resource intensive mediating artifacts makes the project safer till the completion and mitigates the risks of project failure(Cohen, Lindvall, & Costa, 2003).In his study it is also mentioned that the one on one communication is mandatory in the Agile software methodology which means that teams are to be work in close locations. It means that the teams can facilitates each other in making decisions and start implementing it with immediate effect rather than waiting for the correspondence on the issue. It is also mentioned that dynamic prioritization and feature planning are put to the continuous iteration cycles in the methodologies with Agile approaches. In another study Highsmith and Cockburn (2001) mentioned that Agile development needs close customer partnership in order to effectively developed the software and reduce the risk of the project failure.

ii) Iterative property of Agile

Introducing the iterations in the software development whereas makes it effective and efficient to meet the criterion and requirements of the client and contributes in the success of the project there it also makes the development process of the product a little more complicated and time consuming. It is because each iteration in Agile development is self-contained many activities like requirement analysis, design, implementation, testing and deployment. These activities are to be undergo in each iteration to make sure that the project is on the right track according to the wish of the client (Williams, 2010). Each release of the iteration is the updated version of the changes adapted in the preceding release, adapted according to the changing requirement of the client or technological shift during the development process. Iteration is time boxed and its length is determined as one to four weeks decided in agreement phase.

iii) Facilitating tools and techniques for Agile Development

Emejom, Burgess, Pepper, and Adkins (2019) in their recent studies discussed the software development through Agile methodology for successfully managing the project for IR 4.0. Discussing the Agile development in the same context, it is facilitated by the tools such as Model driven architecture of software development in which models are taken as the center of attention and vital artifact. Significant parts of the software can be developed in the earlier stages using automated generation to speed up the development process and also middle-ware code of the final product can be developed to ensure its functionality and feasibility. In another, but relatively old study emphasizes on the fact that risk of the project failure can be somehow mitigated using Agile development process, where the different teams working on the same project works together at the same place where correspondence and communication among them is easy and on time. Moreover the along with such integrated environment the iterative and systematic upgrade in the code using refactoring and automated test suits can be helpful in managing the project so as to minimize the cost of the error detection and debugging those errors even in the later phases of the Agile process.

Agile Methodology Example

i) Scrum& Project Management

Mitigation of risks in the earlier stages of the development process is of non-trivial importance. Many software development methodologies introduce the risk management at earlier or later phases of the process. Risk mitigation and management actually refers to the minimization of likelihood and influences of the adverse events on the success of the project i.e. to eradicate the chances of occurrence of such events which can be responsible for the project failure. In Agile development methodologies the iterative nature of the methods discreetly and indirectly does the work for mitigating the risks. It is suggested by some researchers that risk management in Agile development must be observed at all times. The risks involves in the project development must be addressed as the part of routine stand-ups like in meetings of iteration planning, planning of release and meeting of review of the project (Ahmed & Mohammed, 2019; Buganová & Šimíčková, 2019; Tavares, da Silva, & de Souza, 2019a, 2019b). However the studies proposed the structured approach for managing the risks mainly contributed by Michele Sliger (2019) in his study of PMBOK practices in Agile development processes. This approach includes i) identification of risk, ii) Analysis of risk, iii) Response Planning for risks and iv) Monitoring and controlling of Risk. This approach ensures that risk identification must be done at each iteration to address any new risk arises during the development and this exercise is repeated by the whole team. Moreover, the potential failures and losses must be determined by the team based on the experience of team members, their intuitions and their knowledge about the project. In addition to this, whole team must contributes and participate to identify the alternatives and options in order to mitigate the risks. Once identified and proper responsive strategy is developed the risk should not be left unattended and proper monitoring of the risk must be carried out to control the project failures. In another study by Melegati, Goldman, Kon, and Wang (2019) and earlier by Paetsch, Eberlein, and Maurer (2003) suggested the same process of monitoring and controlling the project and managing the risks. The suggested approach in these studies tells about the scrum technique i.e. sprints, backlog of the product and daily scrum (refers to the daily meeting of team member for about 15 minutes). These three techniques are discussed in great details in the cited studies.

ii) Agile Issues

Timely identification of the errors and flaws regarding architectural design can save the end product from being a failure, though this problem is the trivial point however neglecting this issue can utterly put the integrity of the design on stake. This is because the late identification of the flaws and errors in the design costs too much to rectify at that stage(Turk, Robert, & Rumpe, 2005; Odike & Nnaekwe, 2018). It is also mentioned that the unceremonious assessment of the agile methodologies may not be appropriate or adequate for the safety critical systems establishment.

Comparative Analysis based on the Risk Management and Project Characteristics

i) Maven Projects

In maven or archetypal projects the coding is at the center of activities and in software development the more conducive approaches are agile. Agile practices and principles are broadly used in the small projects where the maximum period of the project is 6 months and 5 programmers are involved. Pilot projects are also potentially benefited from the agility of the development process and the same is true for the experimental projects (Unhelkar, 2016). According to some researchers more attention is paid to the structure of the software, its architecture and planning in great details especially in large systems when waterfall model is observed as it is more predictable. However, the project characteristics of Agile and Waterfall models are discussed in subsequent section of this study.

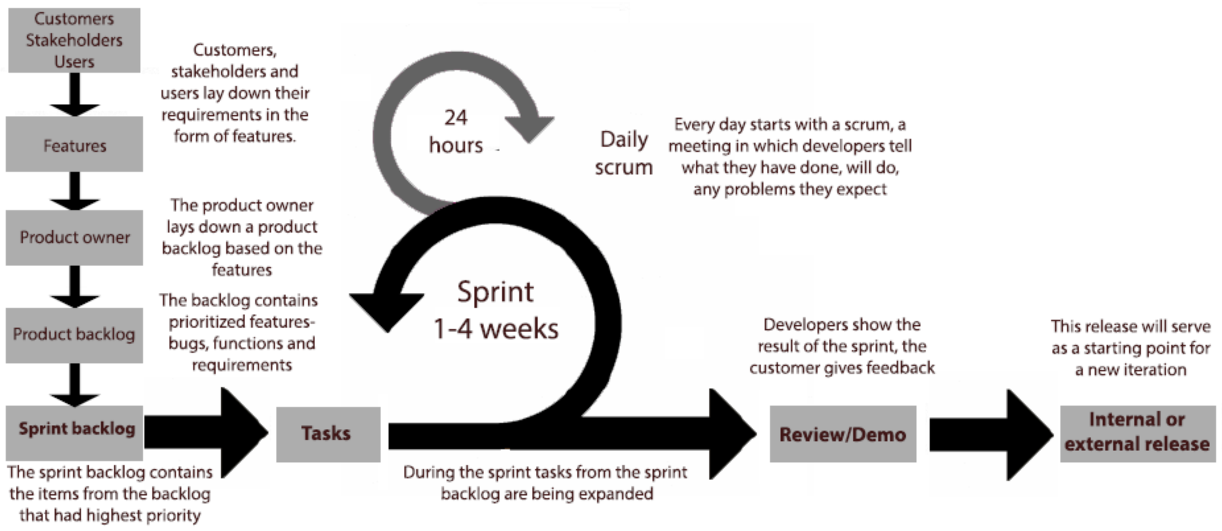
Customers Stakeholders Users
Customers, stakeholders and users lay down their requirements in the form of features.
Features
Product owner
The product owner lays down a product backlog based on the features
Product backlog
The backlog contains prioritized features- bugs, functions and requirements
Sprint backlog
The sprint backlog contains the items from the backlog that had highest priority
Tasks
24 hours
Daily scrum
Every day starts with a scrum, a meeting in which developers tell what they have done, will do, any problems they expect
Sprint 1-4 weeks
During the sprint tasks from the sprint backlog are being expanded
Developers show the result of the sprint, the customer gives feedback
Review/Demo
This release will serve as a starting point for a new iteration
Internal or external release

Figure 3 Scrum Schematic Overview

Project Characteristics of Agile and Waterfall

1. Objectives of Agile VS Waterfall

The goals and the objectives of both methodologies are widely discussed by the researchers from decades. However, vital set of goals and objectives are repeatability, expectedness and optimization for the conventional and traditional waterfall method. Meanwhile, rapid value and response to the changing are the main focuses of the agile development processes (Boehm, 1988).

2. Customer Relations

Relation of the client with developer team is significant in the software development techniques especially when the software development process is agile. The agility of the development requires devoted and dedicated sort of relationship between customer and the developer team for the best results. In this case the implicit knowledge of the client is very necessary and often conducive. If there is not adequate knowledge on the client side then the risks of the failure of the project is probable. However the waterfall model mitigated this risk by expediting the process of documentation (Boehm, 1988).

3. Planning & Control

Project management in software development requires keen attention towards development. Planning, cost estimation, coordination, cooperation, tracking and controlling is required for any successful project. These aspects along with documentation are ceremoniously and strictly covered in the waterfall methodology of the software development. However, agility mainly focuses on the planning than resulting documents.

4. Size

The scalability of the project is major part of the development process. Some methodologies dealt well with the small projects and other dealt with large projects well. However, as far as waterfall methodology is concerned it is true for large projects while agile methodologies do well with small projects. Moreover, the waterfall method is plan-driven and bureaucratic type of methodology, which requires a whole month of a person to develop the product. Thus, does not work well with the small projects.

5. Communication
Communication is the integral part of the agility development while waterfall model depends highly on the documentation of the project. In waterfall model all the documentation has to be begun and ends before the actual project starts.

6. Development
Agile software depends on the comprehensive documentation along with the simplicity of the project which maximizes the focus on the work not done. On the other hand the waterfall model along with the detailed documentation before the requirement phase it also emphasizes on the architecture of the software (Royce, 1970; Sundari, Mediaty, Habbe, Harryanto, 2018).

7. Developers
Waterfall model requires developers those are plan-oriented and sequence driven. It also requires that the developers should have adequate knowledge regarding project and have access to the external resources and knowledge. While in Agile development the developers also need to be agile those are adaptable towards changing situation and should have extensive knowledge about the project requirements and necessities. It also requires that the developers should be harmonious, unrestrained, collocated and skillful (Boehm, 1988; Malik & Manaf, 2018).

8. Test
Waterfall development software methodology is much adapted by the conventional assurance methods for testing the product, as these conventional assurance methods are well-documented and architectural. While in the Agile methodology the developers are encouraged to adopt the updated coding standards along with the design available internally and reviewing the code.

9. Culture
The success of the methodology depends upon the environmental factor of the developer team and requirement of the project. If the emphasize and focus is on the undone work then the Agile will succeed, while if the emphasize and focus is on the order of the work done then waterfall will succeed.

Comparison of Risk Management in Agile and Waterfall
This section is about the management and mitigation of the risk in Agile and waterfall methodologies. It is very important to understand the difference between approaches of these two methodologies towards risk management and Agile must not be treated the same way as Waterfall. In agile methodology the environment have to be very active and reactive to cop up with the daily life routine changes and so do the risk management. So in Agile methodology Risk management takes more active role while handling risks because of its adaptable nature. While on the other hand the waterfall model is offers more time to handle the situation. In waterfall methodology the longer projects have more time to plan before the project begun and also it is not adaptable and less often changes in the requirement occurs. In conventional and tradition approaches of software development the risk management have a role but it is not active as that of in Agile Methodology.

i) Key Factors of Success and Failure with Risk Management in Agile & Waterfall
Project Management have thorough, comprehensive and robust process around itself in both agile and waterfall methodologies. As far as the prospect Identity, quantity, prioritization, planning and management is concerned both the methodologies are seems to be very similar however, there is the big noticeable difference lies between these two methodologies when it comes to the periodic and frequent risk and its management processes. As it is well-established fact about the waterfall model that most of the documentation along with the risk management is done before the project begun so the risk management is only done once before starting the project and it is not that frequent, while in the Agile methodology, due to its iterative nature the risk management is done more frequently, so the risk management is more thorough and cyclic in Agile methodology. Thus due to iteration in Agile methodology, it brings up the hidden and unknown risks to the surface more frequently and can be addressed on daily basis. In waterfall model and other conventional and traditional methodologies only the well known and big risks can be identified in the start of the project and can be addressed, so making it difficult to mitigate risks occurs dynamically.
There are some key features given below that should be kept in hand when dealing with the risks in Agile development methodology.

- Small workshops of Team Risk Evaluation must be conducted and not the large planning session. Shorter sessions and workshops not only take minimum time but also it aided in avoiding the time spent for analyzing the risk that may not appear during iterations.
- Risk analysis and risk register should be made available to everyone in the team. Different techniques can be use to do the collaboration on the centralized information on the portal made for the team members and stakeholders.
- Agile development teams are tends to be organized and democratic thus making the process very dynamic. In some organization the role of Risk Manager is allocated to one of the team member. The responsibility of the Risk Manager is to deal with the risk, its identification, and classify the risks and also escalate the steps of the risks mitigation.
- Ranking of the risks must not be over think. The risk ranking can be done using soft and electronic techniques that can simply rate the risks according to the weight which in turns it can be very helpful to mitigate the highest ranked risk. This way the risk mitigation can also be done according the skills of the team members. Moreover, different types of risks and their mitigation techniques are detailed in the table below.

Category of Risk	**Note**	**Results**
Cost Risks	Resources required to complete the product are facilitated by their more accurate assessment and depends on the each iteration and the ability of targeting the fewer elements in the entire project. The risk in the major inconsistencies in the actual and expected	Reduced & cost is increased

	cost of the project can be mitigated and hence the cost of the planning and management can be increased.	
Technology Risks	Software products with highest quality can be produced by Agile Methodology because it equips the developers with the ability of exploring, experiments and testing various technologies. Due to this risk can be reduced that might occurs while choosing technologies. The risk of neglecting test of any part of the product can be mitigated due to its iterative development.	Reduced
External Risks	The varying variables in the environment can be accounted with the help of small sections and iterative nature which enables the developers to take the required action to lemmatize the effects and consequences of the changing environment on the final product.	Reduced
Schedule Risks	Planning in small periods gives more effective way of performing tasks and defining terms. Time required for the completion of the full product can be increased due to involvement and inclusion of the stakeholder e.g. owner or may be due to the large number of iterations	Risks increased

	because this may lead to abrupt design change and additional iterations. Beside the time spent on the product increases it also increase the quality of the product and gratify the anticipation of the owner of the product.	
Operational Risks	When the project has the risk of losing or it is not well planned then operational risks can be occurred. These risks are critical when large applications are being developed or there are major changes are to be done in the existing product. In agile methodology the typical task is dividing the whole process into small development periods or intervals that offers a closer assessment of every phase and to mitigate the risk of underrated products.	Reduced

Conclusion

In this paper multiple dimensions of the Agile and waterfall methodologies were addressed and introduced. In terms of project management, plan driven waterfall method and change driven agile method both have their own merits and demerits. As far as the small projects are concerned the Agile methodology works good for them. However, big and complex plans work well when waterfall methodology is applied.

On the other hand, in terms of Risk management in the product development each methodology has different approach. Software development methodologies which are adaptable towards changes offer freedom with developers during development of the project and increase interaction between stakeholders. Flexible approach may contributes in minimizing the risks involved in the project and associated stakeholders and produces best results but it is not the best choice in all type of projects. If there are not much communication options between developers and the owner then Agile methodology will not give the desired results according to the expectations of the owners. Incremental methodology and continuous planning are proved to be successful techniques and tools in mitigating the operating risks. It also mitigates the risks involved in the varying environment and unstable actions of the stakeholders and other competitors along with the risks due technological shifts

and planning. Agile methodologies as compare to conventional methodologies also offer periodic team gatherings in order to minimize the individual or team problems which in turn mitigate the risk in project associated to team. Agile methodologies requires large budget because of its longer period of development, and also increase in number of team gatherings and official meetings can elevate the use of resources which then increase the budget. However, the budget in conventional methodologies cannot vary or increased as in the agile. Finally, the agile methodologies can be considered as the good way of development according to the dynamics of businesses process, changing environment, needs and requirements consequently meeting the demands of the project.
Moreover, multiple conclusions can be drawn from the characterizations of the software development and the organizational sustainability related to it. Sustainability is very wide concept that is intimately coupled with the values of the projects. The values of the stakeholders are tightly coupled with the context of the project and impact the domain dependent sustainability. In addition, the set apart in other quality concepts and sustainability in software development life cycle methods is consequential. This is because the distinction between these two again requires the construal of sustainability. However, sustainability that is domain dependent can be developed in to pre-identified requirement categories.

References

Ahmed, M. N., & Mohammed, S. R. (2019). Developing a Risk Management Framework in Construction Project Based on Agile Management Approach. *Civil Engineering Journal, 5*(3), 608-615.

Amjad, S., Ahmad, N., Saba, T., Anjum, A., Manzoor, U., Balubaid, M. A., & Malik, S. U. R. (2017). Calculating completeness of agile scope in scaled agile development. *IEEE Access, 6*, 5822-5847.

Boehm, B. W. (1988). A spiral model of software development and enhancement. *Computer*(5), 61-72.

Bouquet, J., Faure, S., Fève, F., Foucault, M., Garcia, U., Guérin, F., . . . Munier, P. (2018). *Model Quality Objectives for embedded software development with MATLAB and Simulink*.

Buganová, K., & Šimíčková, J. (2019). Risk management in traditional and agile project management. *Transportation Research Procedia, 40*, 986-993.

Cohen, D., Lindvall, M., & Costa, P. (2003). Agile software development. *DACS SOAR Report, 11*, 2003.

Eastman, C. M. (2018). *Building product models: computer environments, supporting design and construction*: CRC press.

Emejom, A. A., Burgess, C., Pepper, D., & Adkins, J. (2019). Agile Approaches for Successfully Managing and Executing Projects in the Fourth Industrial Revolution *Agile Approaches for Successfully Managing and Executing Projects in the Fourth Industrial Revolution* (pp. 1-19): IGI Global.

Enríquez, J. G., Sánchez-Begínes, J. M., Domínguez-Mayo, F., García-García, J., & Escalona, M. (2019). An approach to characterize and evaluate the quality of Product Lifecycle Management Software Systems. *Computer Standards & Interfaces, 61*, 77-88.

Gnatz, M., Marschall, F., Popp, G., Rausch, A., & Schwerin, W. (2002). *Towards a tool support for a Living Software Development Process.* Paper presented at the Proceedings of the 35th Annual Hawaii International Conference on System Sciences.

Highsmith, J., & Cockburn, A. (2001). Agile software development: The business of innovation. *Computer, 34*(9), 120-127.

Jiménez, M., Piattini, M., & Vizcaíno, A. (2009). Challenges and improvements in distributed software development: A systematic review. *Advances in Software Engineering, 2009*, 3.

Malik, S. & Manaf, U.K. (2018). The Effect of Orientation and Mobility Curriculum on the Academic Achievement of Visually Impaired Children among Public and Private Institutions of Lahore. Multilingual Academic Journal of Education and Social Sciences, 6(1), 66–75.

Melegati, J., Goldman, A., Kon, F., & Wang, X. (2019). A model of requirements engineering in software startups. *Information and Software Technology, 109*, 92-107.

Mohagheghi, P., Dehlen, V., & Neple, T. (2009). Definitions and approaches to model quality in model-based software development–A review of literature. *Information and software technology, 51*(12), 1646-1669.

Paetsch, F., Eberlein, A., & Maurer, F. (2003). *Requirements engineering and agile software development.* Paper presented at the WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003.

Penzenstadler, B. (2013). *What does Sustainability mean in and for Software Engineering.* Paper presented at the Proceedings of the 1st International Conference on ICT for Sustainability (ICT4S).

Penzenstadler, B., & Femmer, H. (2013). *Towards a definition of sustainability in and for software engineering.* Paper presented at the SAC.

Penzenstadler, B., Tomlinson, B., Baumer, E., Pufal, M., Raturi, A., Richardson, D., . . . Chitchyan, R. (2014). *ICT4S 2029: What will be the systems supporting sustainability in 15 years.* Paper presented at the ICT for Sustainability 2014 (ICT4S-14).

Petersen, K., Wohlin, C., & Baca, D. (2009). *The waterfall model in large-scale development.* Paper presented at the International Conference on Product-Focused Software Process Improvement.

Royce, D. (1970). Winston. *Proceedings, Managing the Development of Large Software Systems, IEEE WESCON*.

Sohaib, O., & Khan, K. (2010). *Integrating usability engineering and agile software development: A literature review.* Paper presented at the 2010 international conference on Computer design and applications.

Sundari, S., Mediaty, Habbe, A.H., Harryanto (2018). Heuristic of Representativeness and Anchoring-Adjustment in Budgeting, International Journal of Academic Research in Accounting, Finance and Management Sciences 8 (4): 52-60.

Tavares, B. G., da Silva, C. E. S., & de Souza, A. D. (2019a). Practices to Improve Risk Management in Agile Projects. *International Journal of Software Engineering and Knowledge Engineering, 29*(03), 381-399.

Tavares, B. G., da Silva, C. E. S., & de Souza, A. D. (2019b). Risk management analysis in scrum software projects. *International Transactions in Operational Research, 26*(5), 1884-1905.

Turk, D., Robert, F., & Rumpe, B. (2005). Assumptions underlying agile software-development processes. *Journal of Database Management (JDM), 16*(4), 62-87.

Unhelkar, B. (2016). *The art of agile practice: A composite approach for projects and organizations*: Auerbach Publications.

Casteren, V. W. (2017). The Waterfall Model and the Agile Methodologies: A comparison by project characteristics: Open University of Netherlands.

Williams, L. (2010). Agile software development methodologies and practices *Advances in Computers* (Vol. 80, pp. 1-44): Elsevier.

Odike, M., & Nnaekwe, U. K. (2018). Influence Of Teachers' Attitude Towards Teaching Profession On Under Graduate Non-Education Students Perception Of Teacher Education. International Journal Of Academic Research In Progressive Education And Development, 7(4), 67–79.