Review Article

Explicit Risk Management in Agile Software Projects: its Relevance and Benefits

ABSTRACT

This study intends to review and deduce based on the risks identified, the relevance and benefits of formally managing risks in agile software development projects. To achieve our aim, we reviewed risk management procedures in a typical agile setting as well as researches that exposed the insufficiency of the inherent risk management process in agile projects and identified risks. Related research papers from peer reviewed journals and other reliable sources were reviewed to extract risks that occurred using agile without explicit risks management. The study inferred that some risks do exist that occurred with the introduction and use of the agile method itself. Also, there could be risks that surfaces when project size exceed a limit. Thus, managing risk explicitly will go a long way to address such risks. Consequently, we were able to deduce the relevance of implementing explicit risk management in agile software development project.

This study showed that it is beneficial to incorporate formal risk management procedure in agile software development when mega software project is being developed.

Keywords: [Explicit risk management, Agile software development methodology, Software projects}

1. INTRODUCTION

Like every other project, software development projects are faced with some level of uncertainty. Uncertainties in software projects may result to risks ranging from scope creep, budget overrun, not meeting the scheduled delivery deadline, production of wrong and or incomplete software specification and many more.

Risks are inevitable in every project and must be managed. Risk management is the systematic procedure that involves the identifying, analysing, prioritizing, monitoring and controlling of risks to reduce and possibly eliminate any negative outcome of the risks in actualizing the project's objectives. It is a must-follow process [1];[2] and need be carried out throughout the entire software development process.

Agile software development methodologies are a group of methodologies having some common features namely time-boxed development cycles, ability to change or modify requirements during development, constant communication with customers, involvement of self-organising cross-functional teams and regular testing after each iteration build among others. These inbuilt features manage software development risks. While this works well with small projects [3]; [4] large projects are bound to have higher number of iterations and some risks will be left unidentified or forgotten in the process. The resultant effect could be faced in later stage of the development cycle and this will likely be costlier to rework[1];[2].

To avoid this kind of scenario, it is important to incorporate risk management procedure explicitly with the agile methods. This is necessary because risk management is a systematic process that is continuous all through the development of the software and infact is a project of its own [28] which should be integrated formally. Managing risk in an unordered and unconscious manner will not deliver the best result. Agile methods do not recommend specific procedures to support risk management [5]. They think that the short iterations they utilize curtail risks however this is insufficient [6]. According to [7] the inherent risk management procedure by SCRUM which is an agile method is not as good as in the case of traditional management methods because some steps of the risk assessment are not fulfilled except for the activities of the risk identification. Consequently, it is suggested that SCRUM be enriched by some selected steps from PRINCE2 risk management which promises a better result for delivery even in global software development projects [7]. Agile teams are democratic and self-organizing ideally, consisting of developers with all the necessary required skills, however they do not consider a risk manager [2]; [1]; [8]; [9] whose sole duty is to take note of all risks identified by the team and also ensure that the risks are managed properly. It is believed that risk management is passive and implied in the agile process. [8]; [10] states that though Agile methods are speedy way of developing software, the need to implement a robust risk management practice cannot be rule out. Better ways to integrate proactive risk management measure need to be put in

place carefully without compromising the agile spirit. It is categorically stated by [11] that Agile methods needs a formal technique to manage risks when multiple agile teams work on the same product, stressing that a higher coordination effort is required and more formal practices are applied.
Even proponents of Agile Methods attest to the fact that former risk management is necessary in some projects. According to [12] most successful software projects utilize the hybrid of traditional risk management and the agile software cycle. In otherwords, traditional risk management and agile software cycle are complementary. Meanwhile, traditional risk management follows a formal process. As a practitioner, he pointed out that in Agile methods, formal way of managing risk becomes necessary when the project is expensive, have many touchpoints and involve the use of new technology. Previous works were made to identify possible risks that can emerge when utilizing agile methods in projects and in this paper, we intend to review and deduce based on the risks identified, the relevance and benefits of formally managing risks in the agile development projects.

2. METHODS

To achieve our aim, we reviewed risk management procedures in a typical agile setting as well as researches that expose the insufficiency of the inherent risk management process in agile projects and identified risks in such projects that relied on the implicit means of managing risks. Related research papers from peer reviewed journals and other sources were reviewed to extract risks that occurred using agile without explicit risk management. Consequently, we were able to infer the relevance and benefits of implementing explicit risk management in Agile software development projects.

3. Risk Management in a typical Agile Environment

[13] listed business risks, financial risks, and technical risks as the major risks in a software development. In his study, testing, validation and documentation that ensures regular delivery in short increments as well as continuous integration practices are seen as being practiced in scrum, a popular agile method. However, agile teams hardly do documentations. Moreover, in large projects, the number of iterations will increase and consequently, result in omission of some risks untreated. The engagement of cross functional teams is seen as complete solution to improving the awareness and use of latest technologies and ideas that will result in the production of quality software. This is true, however, the likelihood of leaving some risks unattended is obvious as there is no clear handler of risk management [1] It is perceived that the product owner plans how the budget utilized will be in the project. He ensures that the expectations of the stakeholders are met at quicker releases thereby reducing cost of production, but this is seeming in smaller projects where iterations and complexity of software are mild.

4. The need for Explicit Risk Management in an agile environment

[14] study on risks in agile methods of development, revealed that the inherent risk control practice utilized in agile is insufficient and suggested the use of a formal risk management approach. The study identified some risks which occurred as a result of the use of the agile method itself and other risks which become more pronounced when agile method was introduced. The study was conducted in a large telecommunication-based company. Six projects which involved the customizing of a mega software product to suit the needs of customers in their respective operation location were carried out. Major development and testing efforts were made. Customers' requirement needs and where the product is used were different though the same technology was deployed. Members of the development team working on a project are located in the same center however the project teams are located in different geographical areas. Three of the projects had their development teams in Poland, while two other projects had their teams in South Africa and the remaining one in Germany. Customers resided in locations other than the project development team that executed the project. Customers were large mobile network operators in Africa, Europe and Latin. Scrum, Extreme programming and Agile Project Management (APM) were the agile methods used in the six projects. Though in five of the projects, teams had no prior experience of agile methods, thorough training on agile practices was made. Scrum was the main method utilized though some practices of XP programming and Agile Project Management were applied. Semi-structured interview were conducted with key project stakeholders. The interviews were aimed at identifying the strength and weaknesses of the examined projects as regards Agile methods as well as risks and opportunities for future Agile projects. Identified risks were as follows: risk of neglecting continuous integration, development process risks, development system risks, contract risks. Most of these risks are further sub-divided into more risks. They are risks that emanated from the introduction of Agile or became more visible when agile was introduced.
According to [15] the method of tasks prioritization as a way of managing risks in the agile methods is inadequate. He pointed out that such activities as the proper use of resources in achieving the enterprise goals are not inherent within the

execution of tasks but are very important to be taken care of as they are considered part of the project. In otherwords, there are risks that cannot be addressed by implicit way of development employed by the agile methods.
Derfer (2016) stressed the need for implementing risk management in an explicit manner. According to him, techniques such as time-boxed iterations, demos, retrospectives and team ownership of each sprint's commitment go a long way in addressing some risks in the agile projects, however, these mechanisms are inadequate as they cannot be used to take care of risks in projects that are of medium to larger size which are beyond a team's control. Derfer emphasized the agile processes are better in identifying risks at early stage of development rather than proffering solutions to identified risks.
Despite the fact that agile based projects handle risks dynamically utilizing their inbuilt approach that is iterative and seasoned with daily meetings which together minimize risks, Prakash, and Viswanathan (2019) explained that they still require some form of explicit method of managing risks which is imperative but lacking. From literature reviewed, proper risk management procedure is a necessity as its absence in a software project results in failure of varying magnitude. The study revealed that globally only 16.2% of software projects are completed within the stipulated budget and time while 52.7% and 31.1% of projects are faced with issues of late delivery and budget overrun and outright cancellation respectively.

The result of the study conducted by [18] confirmed that agile methods by their inherent practices mitigate certain risks which had helped to facilitate speedy means of completing software production. However, their studies also discovered some risks factors associated with the use of agile software development methods. This implies the necessity of the use of other means of managing risks and also further buttressed the significance of utilizing a formal means of mitigating risks associated with agile methods. In their study, Agile method of software development practices were examined in 28 organisations. 112 Interviews and 25 interviews were conducted in the 28 organisations and with other agile software development consultants and contractors respectively. This was to investigate reasons behind increase in failure and let-up in the use of agile software practices by some companies. Key risk factors related to Agile methods were identified. These include technical debt, fractured development and operation, Increased defects in newly formed agile software development (ASD) teams, Fragmentation of project management tools, and Knowledge Retention. It is obvious that agile software developers are faced with conditions that sometime threaten the smooth flow of the process which in turn is likely to compromise agility.
According to the study result of [2], an estimated savings of 40% is made when a formal risk management process is integrated with agile software method in software projects. A mathematical risk management model for use in the agile methods was developed to implement an explicit risk management practice. Features that will estimate risk management process cost as well as cost-benefit for implementing a formal risk management procedure in agile software development projects was achieved. Use case model and activity diagram were developed to capture and understand activities involved, thereafter numerical calculation procedure derived appropriately, adopting risk cost estimation tool by [19]. Derivations for formal risk management cost and cost-benefit for implementing the formal risk management process made.
An online survey study which involved the distribution of questionnaires to 54 agile adopters by [20] was made. It was aimed at unearthing risks faced by agile practitioners and how such risks are moderated. Findings revealed that among others, requirements and schedule are the most common risks faced in software development projects using agile methods and except for the inbuilt risk control measures, there are no formal ways risk management procedures followed. It was also revealed that agile software developers seldom assign risk management role to keep track of identified risks and their management throughout the development process. This is evident in their result that over 80% of the respondents confirmed no risk management role was assigned in their project teams. It implies no record keeping of risks data and this in turn means some risks are likely to be half-handled and even overlooked. This could be the reason of most software project failures.
[21] study on how to improve risk management in large agile settings was made. A case study in a moderately large ecommerce company was conducted by interviewing four production leads and four cross-team project managers. While each Production lead took supervision of a single team where features of the product are developed in an autonomous way, the Cross-team Project leads supervised multiple agile teams where more than one team work on a requirement (s). Findings revealed that whereas the implicit risk management procedures were adequate enough for the single teams who work independently on a particular feature, more formal risk control strategies are required for the latter (cross-teams) where many teams work on a particular feature. Reasons given for this conclusion are that there are no clear organized responsibilities laid down on who and how risks is to be managed and this indicates that explicit risk control measure is necessary for production of large software. Consequently, the study recommended complementing agile practices with the Traditional software development practice in mega software projects.
A survey study by [22] compared risk management practices utilized in traditional methods and agile methods with the key aim of identifying risks associated with the methods. It reveals that Agile methods are suitable for short-term projects

that require little planning and documentation as opposed to traditional methods. In addition, agile methods are flexible such that requirements are modified and or added at every stage of development as opposed to the rigidity applied in the traditional methods. However, it was deduced that the use of agile methods itself, can lead to project size creep, improper sprint planning, absence of specific experts can cripple development process, users unable to assess software releases before next release of another module, lack of progress tracking among developers in complex environment.

5. Summary of Existing Risks in the Agile Software Development

Most of these risks are further sub-divided into more risks. however, sub risks are treated separately here.
The risk of neglecting continuous integration (CI) [14] Continuous Integration is a software engineering practice and in fact a best practice in agile software methodology. It involves routine integration of code changes into a shareable version control repository repeatedly and testing changes as early as possible to ensure defect free product production [23]. Failure to implement CI may result in spending more time in rework, uncertain project completion time and more development effort.

Improper sprint /iteration plans [22]: Sprint/Iteration plan is a plan done at the beginning of each sprint/iteration planning meeting to agree and shortlist user stories or items agile team can complete in the sprint and how work can be done. Thus, bad sprint plans can make the team set expectations that are unrealistic thus making the development process complex.

Risk of Team not able to self-organise and make group decision [18]
Self-organisation is an integral part of agile software development projects.Though, it can improve team performance when appropriately done, it also have risks associated with it. For instance, when the team have too many work to process, coupled with the involvement of team with narrow expertise knowledge. In agile projects, apart from the team executing tasks in iteration builds, they also monitor], control and make decisions on how the task are done. Thus, team members do not only do project execution tasks but managerial tasks as well. Wrong decisions are likely to be made if full knowledge about the project is unknown and when decision of a team member is accepted because he/she is popular instead of on merit.

Missing of Infrastructure at customers' site for proper customer involvement in the development process [18]: All necessary infrastructure (both hardware and software) needs to be put in place. Such as dedicated hardware which includes hardware networking components and its installation. In fact, all required for implementing test platform should be provided and put in place even before the commencement of test otherwise the test will be not be effected at all or done haphazardly or even delay the agile spirit if it has to be arranged for test time.

The risk of lack of or limited compatibility of tools with Agile practices [18]
Agile engineering practices such as test driven development and continuous integration require some tools different from those used in the traditional development scene. Thus, there is bound to be compatibility issues when such old tools are used in place of the agile driven tools. In the existing literature, the risk of incompatibility was experienced. Besides, high cost of running regression tests as well as elongated run time of the testing process was evident.

Risks due to the use of faulty deliverables: this according to the study of WA involves the use of readymade components and software tools either produced in house or external by project team in the developing process to reduce costs and time without modification of such tools to suit its use in the agile settings.

Risk of Inefficient Scrum meetings and ineffective Scrum roles: Agile development process is associated with regular meetings as well as introduction of new roles different from that in the traditional methods. The meetings consume substantial time and efforts. New roles like the product owner and scrum masters introduced are costlier to maintain than the project roles of the traditional methods. Though daily meetings and roles are vital agile practices, they do not add direct values to customers and are significant project overheads that if not managed properly will result in delays and overruns.

Project size creep [22]: the incremental feature utilized by agile allows for the addition of new features at each sprint/iteration of development. This could expand the project size and increase product cost.

Absence of specific experts [22]: Agile teams are cross-functional. This implies each team is made up of people with different expertise and skills. The benefits is that all capabilities required to undertake its scope end-to-end without help from another team is avoided. Supposing a team member leaves or dies, it could hinder the development process from achieving its goal at the specified time.

Users unable to assess software releases before next release of another module [22]: Frequent software releases at the end of each sprint cycle prevent users the privilege to completely assess previous releases as a result, key performance indicators are not properly spotted out.

Lack of progress tracking among developers in complex environment [22].
This is the poor tracking, handling and managing of risks especially in large software projects [1]; [13] ;[20]; [21], [16]. In medium to large projects, the number of iterations is large, so also the risks that are identified. There must be a risk manager in each location. Depending on the size of the project and number of locations, Risk manager(s) must be available in meetings to keep track of every identified risks. This can be effectively done issuing risk track forms to team members who record risks identified. The risk manager enters these risks in an automated risk repository via risk application. That way, risks details can easily be recalled and status updated. Such risks tools as Agile Risk Track Sheet (ARITS) and Repository as developed by [2] are suggested because they are web-based with smart database thus, can be used by all teams working in different locations.

Lack of control of cross-teams implementing a single feature: The engagement of many teams in the implementation of a single feature of a software indicate that such a feature is large. This in turn means the software is a large one. According to [21]; [16], formal risk management process is needed to exercise control over who will deal with which risk and keeping record and track of it till migration.

Technical debt: Also known as tech debt or design dept or code debt) describes consequences that occur when development teams utilize easier and faster approaches (shortcuts) to meeting delivery deadlines usually a software functionality within the specified time [24]; [25]. These consequences in which ever form it comes namely code debt, test debt, documentation debt, design debt involves some financial cost to refactor in future. Thus, it is the result of prioritizing speedy delivery over perfect code [26]. Technical DEBT is noticeable in agile and infact accumulate over the time due to their adherence to strict rules of delivering software features to clients within sprint in a consistent and continuous manner. Accumulation of such debt could lead to reworks and complexity which in turn could requires more effort, money and project delay and poor software quality as well in the long run [18].

Risks related with close involvement of Business Stakeholders/customers: According to the literature reviewed, though one of the successes achieved in agile is attributed to communicating constantly with stakeholders which makes them have an edge over the classical methods, stakeholders usually delist nonfunctional requirements such as security threats from the list of user stories. Probably due to lack of understanding of the technicalities involved. Meanwhile, Security threats are evident and need to be checked continuously, failure to do so, further increases technical debt [18].

Fractured development and operation risk: According to [18], this refers to the disconnect between the agile development team (ADT) and the IT operations team. While the ADT focuses on building new software modules and application as well as ensuring quick delivery to users, Operation team ensures users get a fast and bug free software products that is stable and reliable in its operational environment. The IT operations teams usually, include systems administrators, network engineers, and infrastructure specialists. on the other hand, a typical agile team consist of the product owner, team leader, specialists, architecture owner, development team members, scrum master, stakeholders which includes direct and indirect users, senior and portfolio managers [27] Though both groups work towards the common goal of providing good software builds and services, their approaches in accomplishing that is different as they have different jobs, job priorities, work practices and pace. As a result, the risks of delayed delivery of software builds within the scheduled time is evident. This is so as Operations Team attend to Infrastructure and service needs rather in a linear order contrary to the agile approach and pace of delivering builds in short time -boxed iterations. Another issue experienced in a research reviewed is that there is poor collaboration between the agile team and the IT operation team and this had resulted in the Agile team producing software not compatible for implementation in the operations environment without a rework for example, reworking a software to suit its use in a server of the operating environment. The agile team have their sole interest of delivering working software within allotted iteration without taking into

consideration necessary knowledge of the operational environment and deployment such as Infrastructure and networking operational issues which are necessary for the system to operate in its live environment, rework requests are not given utmost priority that it deserves but developing of new features is at the foremost. In the overall, the risk of delayed deployment of developed software is likely to occur.

There is **Increased defects in projects handled by agile software development (ASD) teams that are new to agile way of development.** This is risky as this implies more effort, time and cost to correct such defects. It was however seen that as developers gain experience in the agile environment, the occurrence of such defects begin to decrease and reaches acceptable level. This according to the study [18] was observed in small organisations.

Fragmentation of project management tools: Because Agile teams are self-organizing, they tend to choose project management tools they are conversant with to actualize their development needs. Though this motivates them and enhance their performance, it could also lead to confusion and complexities during integration of work across domain and teams [18].

Knowledge Retention
ASD by virtue of their policy, value face-to-face communication over written documentation. This means details of each system developed is retained until completion only if team members that started work on it is retained till the end and with the organization. Practically this is not so, members are often reshuffled and reassigned task in other teams from time to time. If a new team member joins a team, velocity and quality drops. The reason behind this is that Light documentation is utilized in agile software methods. Thus, details about the system developed is not enough to expose the new team member of the understanding of the codes. Consequently more time is used to unravel the details of the version of the software and the way forward.

Customer Involvement not done in the proper way as specifics of agile practice

Delay in customers' response to give valuable response

Customers' low level of interaction with Agile Team and in a way different from the Agile way
One major characteristics of the agile method of development is Customers close interaction with the development team from beginning to the end of the project. This is to ensure that software builds meet the requirement specification. However, according to the reviewed study, one of the risk noticed is that customers were not always in close contact with the development team and in most cases only make themselves available at the beginning and end of the project. With the manifestation of this risk comes another risk of customer not available to give feedback and responses to development team on requirements and other inquires on the next iteration/sprint.

Requirements and schedule risks are the most common risks faced in software development projects using agile methods [20]

6. Relevance and Benefits deduced from Existing Explicit Risk Management Practice in an Agile setting
Generally, the relevance and benefits of explicit risk management are inherent in the various steps and roles of the risk management process.
Agile software development methodologies also manage risks but in an implicit way. The modular development of software and its unit testing in each iteration as well as integration testing with other modules alongside with acceptance testing to check its conformance with requirements specifications are all effective ways to mitigate risks. Inspite of this, from the literature reviewed, other risks do exist that occur with the introduction and use of the agile method itself. Also, there could be risks that surfaces when project size exceed a limit. Thus, managing risk explicitly will go long way to address such risks. In a nutshell, explicit risk management in Agile software development methods will do the following:

1. Helps to keep track of risk data by the introduction of a risk manager. Risks data is very important to keep track of risks identified, closely monitor the treatment of the risks identified and ensure its mitigation. Thus, the presence

of Risk Manager in the develop team is very important to avoid skipping of identified risks. Such role is not explicitly spelt out in a core agile setting [1].

2. Helps to initiate the proper use of resources in achieving the enterprise goals. Agile methods need utilize some formal means since it is not part of the execution of tasks assigned in the inherent process yet it is an important aspect [15].
3. Helps to address risks that emanates from the introduction of Agile or became more visible when agile was introduced as revealed in studies reviewed. Such risks can be well taken care of by external processes [14].
4. Helps to manage risks when multiple agile teams work on the same product/feature. This is so because higher coordination effort is required, and application of more formal practices needed when cross-teams work on a product. According to [21], there are no clear organized responsibilities laid down on who and how risks is to be managed and this indicates that explicit risk control measure is necessary for production of large software.
5. Generally, explicit risk management is a necessity in the development of large software projects using agile method of development. The execution of large software projects in the agile environment is completed in many iterations and more efforts to track and control risks consciously is needed otherwise some risk may be left unattended to and may escalate to bigger problems at some point in the development cycle.
6. Explicit risk management may likely help to reduce monies expended for rework of risks by about 40% where only the inherent agile risk control practices is being utilized [2].

7. CONCLUSION

Explicit risk management in medium to large scale agile software development is highly recommended to identify, keep track and control risks including those risks that will emanate as a result of adopting agile method of development. Agile methods are methods practiced by many because of its features of swiftness in software products delivery however, the introduction of agile methods is associated with risks and must be dealt with.

REFERENCES

1 Thom-Manuel OM, Ugwu C, Onyejegbu N. An Extended Agile Software Development Project Budget Model.International Journal of Computer Science and Software Engineering. 2017: 6(12):306-314.

2 Thom-Manuel OM, Ugwu C, & Onyejegbu LN A New Mathematical Risk Management Model for Agile Software Development Methodologies. International Journal of Software Engineering & Applications. 2018; 9(2)*:*67-86.

3. Cohn, Mike. (2009). Succeeding with Agile: Software Development Using Scrum. Kindle ed. Addison- Wesley

4. Alharbi ET, Qureshi MRJ. Implementation of Risk Management with SCRUM to Achieve CMMI Requirements. Computer Network and Information Security. 2014; 11:20-25

5. Moran A. Agile risk management. In Agile Risk Management. 2014 (pp.33-60). Springer, Cham.

6 Chaoucha S, Mejrib A, Ghannouchia SA. A framework for risk management in Scrum development process. Procedia Computer Science. 2019; 164, 187-192.

7. Katarína B, Šimíčková J. Risk management in traditional and agile project management. International Scientific Conference on Sustainable, Modern and Safe Transport (TRANSCOM). pp. 986–993 High Tatras, Novy Smokovec – Grand Hotel Bellevue, Slovak Republic: Elsevier B.V

8. Micheal, K. (June 14, 2012). How is Risk Management in Agile Development Different From Risk Management in Waterfall Model? https://pm.stackexchange.com/questions/5957/how-is-risk-management-in-agile-development-different-from-risk-management-in-wa

9. Hammad M, Inayat I, Zahid M. Risk Management in Agile Software Development: A Survey. International Conference on Frontiers of Information Technology (FIT). 2019. https://ieeexplore.ieee.org/abstract/document/8991647

10. Deloitte Future of risk Management in Financial Services: Integrating Risk Management and agile Projects. 2019. https://www2.deloitte.com/content/dam/Deloitte/lu/Documents/risk/lu-risk-future-of-risk-series-integrating-risk-agile.pdf

11. Schön EM, Radtke D, Jordan C. (2020) Improving Risk Management in a Scaled Agile Environment. In: Stray V., Hoda R, Paasivaara M., Kruchten P. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2020. Lecture Notes in Business Information Processing, vol 383. Springer, Cham

12. Smith, G. (2012). Risk Management in an Agile Lifecycle. Available: https://www.agilealliance.org/wp-content/uploads/2016/01/Agile-Risk-Management-Agile- 2012.pdf

13. Kendis Team, (May 23, 2019). Risk Management in Agile Scrum. Available : https://medium.com/@media_75624/risk-management-in-agile-scrum-e0e5d18f0cf

14. Walczak, W & Kuchta, D. (2013). Risks characteristic to Agile project management methodologies and responses to them. *Operaions Research and Decisions, 23,* 75-95.

15. Moran, A. (2014). Agile risk management. In Agile Risk Management (pp. 33-60). Springer, Cham https://scinapse.io/papers/1042857008

16. Derfer, B. (2016) "Introducing the Agile Risk Management Framework, Agile Six Applications, Inc". Available on https://www.agilegovleaders.org/wpcontent/uploads/2016/03/Agile_Risk_ Management _ Framework.pdf

17. Prakash, B. and Vijay Viswanathan. "Risk Prioritization for Software Development using Grey Wolf Optimization 1458." (2019).

18. Elbanna A, Sarker S. The Risks of Agile Software Development: Learning from Adopters. *IEEE Software*. 2016. 33(5):72-79. doi: 10.1109/MS.2015.150.

19. Khatavakhotan AS, Hashemi NT and Ow SH. A Mathematical Risk Management Model for Iterative IT Projects based on the Smart Database. International journal of information and Electronic Engineering. 2011;1(3):229-233

20. Hammad M, Inayat I, Zahid M. Risk Management in Agile Software Development: A Survey. International Conference on Frontiers of Information Technology (FIT). 2019. https://ieeexplore.ieee.org/abstract/document/8991647. doi:

21. Schön EM., Radtke D., Jordan C. Improving Risk Management in a Scaled Agile Environment. In: Stray V., Hoda R., Paasivaara M., Kruchten P. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2020. Lecture Notes in Business Information Processing, vol 383. Springer, Cham

22. Alshathry, S., Alnamlah, B., Alkassim, N. & Jamail, NSM. Risk Management in Agile and Waterfall Models: A Review. International Journal of Advanced Science and Technology 2020; 29(9):1149-1157

23. Radigan D. Continuous Integration. 2020. Available: https://www.atlassian.com/agile/software-development/continuous-integration

24. Holvitie, J., Licorish' S. A., Spínola , R. O., Hyrynsalmi, S., MacDonell, S. G., Mendes, T. G., Buchan , J., and Leppänen, V. . Technical debt and agile software development practices and processes: An industry practitioner survey. *Information and Software Technology*. 2018; 96(141-160).

25. Wolpers, S. (March 22, 2019). Technical Debt and Scrum: Who Is Responsible? Agile Zone.https://dzone.com/articles/technical-debt-amp-scrum-who-is-responsible

26. Productplan (2020). Technical Debt . Available :https://www.productplan.com/glossary/technical-debt/

27.
Lynn, R. (2020). Agile Roles in Software Development. https://www.planview.com/resources/articles/agile-roles-software-development/

28. Ylimannela, Ville. A model for risk management in agile software development. (2013).