International Journal of Engineering & Technology, 7 (4.31) (2018) 113-117

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Improving the Governance of Software Maintenance Process for Agile Software Development Team

Salfarina Abdullah[1]*, Mangaiarasi Subramaniam[2], Sazly Anuar[3]

[1,2]Faculty of Computer Science and Information Technology, UPM
[3]Universiti Kuala Lumpur Malaysia France Institute, Bangi
**Corresponding author E-mail: salfarina@upm.edu.my*

Abstract

Software maintenance is one of the most debated phases in software development process for so many years. Having reputed as the most expensive phase of software development life cycle (SDLC), it utilizes the maximum share of the overall project costs as well as time. Agile software development provides opportunities to assess the direction of a project throughout the development lifecycle. However, it does not ideally map with the existing software maintenance process. One of the highlighted issues is the difficulty for searching of information as well as lack of knowledge to solve the maintenance problems within certain time frame. Thus, the main objective of this study is to improve the governance of software maintenance process in an Agile development team. In doing so, a tool named Axita is developed to assist the software maintenance team for storing of information in central data repository and managing projects in more efficient and timely manner. Based on the literature review as well as mapping between the agile software development and the existing ISO software maintenance process, we also proposed six best practices to better govern the software maintenance process in an Agile development team, to overcome the difficulty of information finding and reduce the time spent to solve the maintenance issues. We believe that our study and findings complement the efforts that have been put forth in improving the way we manage software maintenance thus enhance the efficiency of the software development process.

Keywords: *Agile; Best practices; Software Maintenance*

1. Introduction

Software maintenance is one of the most important phases in the Software Development Life Cycle (SDLC). It plays an essential role in providing service to the client after the software product has been delivered. In general, there are four types of software maintenance namely, Adaptive, Perfective, Preventive and Corrective. Adaptive maintenance is modification of a software product performed after delivery to keep it usable in a changed or changing environment. Perfective maintenance is modification of a software product after delivery to improve performance or maintainability. Preventive maintenance is modification of a software product after delivery to detect and correct latent faults in the software product before they become effective faults. Finally, adaptive maintenance is modifying the system to cope up with changes in the software environment. There are always debates going on about software maintenance after the development is completed. It is an expensive activity that consumes a major portion of the cost of the total project [1]. The time spent, and the effort required to fix the defects at this phase consumes about 40-70% of the cost of the entire SDLC. [2] adds by saying "Agile Methods only apply to the Software Development portion of the lifecycle and not apply to the Software Maintenance portion of the lifecycle". A better way to conduct the process of maintenance should be practiced throughout the Agile software development life cycle to overcome the maintenance problems, bugs and enhancement after the development.
This study focuses on improving the governance of software development process (SDP) for Agile software development team. Our strategy is by mapping the existing software maintenance process with IEEE ISO software maintenance standard and other suggested processes from the literature review. A maintenance tool, namely Axita will be developed to assist the Agile maintenance team. It aims to improve the task management during maintenance process as well as issue of time consumption in performing maintenance activity.

2. Literature Review

Many researches have been invested in software maintenance issues and solutions. [7] conducted a research that optimizes the agile development practices for maintenance operation with nine heuristics. The research was cooperation between the maintenance unit at Aveva and the information system group at Aolborg University. According to them, to maintain an iterative connection between theoretical literature and action research study is by encouraging the literature study to influence under consideration in action research project and vice-versa.
It is generally accepted that agile methods share a group of common characteristics [18, 19], which include an iterative development process, focused work objectives around delivery points, small teams working closely together, close customer involvement, face-to-face communication, light documentation, frequent testing, intrinsic motivation through collective ownership, knowledge transfer through openness, and a focus on a high quality of code and product. These characteristics are also understood to function well, at least in some kinds of development situations [7]. Some challenges associated with agile maintenance were also high-

lighted. As we all know, the Agile development is the iteration development where the common changes and task list is missing on the maintenance [20]. Maintenance sprints are subject to interruption by urgent client's demands and there are few common delivery points or integrated releases making as another challenge in agile maintenance. In addition to this, the maintenance team must work closely with many different systems and always rely on customer involvement. Maintenance engineers have less face-to-face communication and often work side-by-side with customer. The necessary documentation is often neglected and incomplete, as well as gap in interaction that complicates the maintenance problem solving.

Based on [7] research analysis, the key issues derived from Aveva is looser relationships with customers compared with earlier work practice. For example, when the process is getting closer towards maintenance, the customer often prefers fast and easy exit. Besides, the mixture of emergency and difficulty with estimation causes inconsistence and unfinished work. The feedback from customer causes less confident while the incomplete documentation from the development team has causes a delay to understand the issue. The frequent change of resource in maintenance also causes impact in maintenance process. The less helpful use case with low communication value from customer also causing this issue not to get fixed and maintenance were unable to complete fixing on time. Besides, there is another researcher, [18] who did a research on agile support and maintenance of IT service. According to him, agile approaches considered fit with maintenance process and activities. He introduced a support and maintenance control framework which supported the Agile practices and approaches.

As we all know, Agile development is the iteration development where the common changes and task list is missing on the maintenance [20]. Maintenance sprints are subject to interruption by urgent customer demands and there are few common delivery points or integrated releases which make another challenge in agile maintenance.

Techniques and practices are concrete steps how to implement a principle in the real life. Agile practices are implemented incrementally according to our need aiming to achieve and contribute to enterprise business goals and to cover a gap in the principles. It is neither mandatory to implement all practices, nor to do it at the same time as a big bang. The result of the described approach (Production Phase, principles, practices) is an effective (not bureaucratic) maintenance process supporting objectives of delivery (project or IT service [14].

From the author's research, the empirical evaluation from the survey from service X – forest and service-telecommunication show that the 13 approaches (as per Agile manifesto) have improved the maintenance activities. Based on the results, we can state that described approach can mitigate problems of traditional methods (low innovativeness, creativity and motivation; process, not value oriented measures; quality problems). Maintenance is not a prescribed detailed process; appropriate process is built up from practices and differs in every team (different focus, practices to be implemented). The critical success factor mentioned by the team members and managers is hands on support by mentor (skilled and experienced person with agile approaches) helping to identify the root causes and implement principles and proper practices.

However, when we compare between [7] and [18] research, both have implied the Agile practices differently. [7] proposes the nine heuristics to improve the situational in software maintenances activity meanwhile, [18] introduces a new control framework for software maintenance. One thing common between these two is both only proposing practices which are more to the individual basic in the maintenance process. The solutions still depend whether the maintenance engineer decides to follow the practice. There are no proper tools that reflect the agile maintenance practices.

According to the IEEE standards of software maintenance, a proper process has been designed for widespread use of the maintenance team in any software development team. The main maintenance processes in this ISO software maintenance is process implementation, problem and modification analysis, modification implementation, maintenance review/acceptance, migration, and retirement (Refer Figure 1).

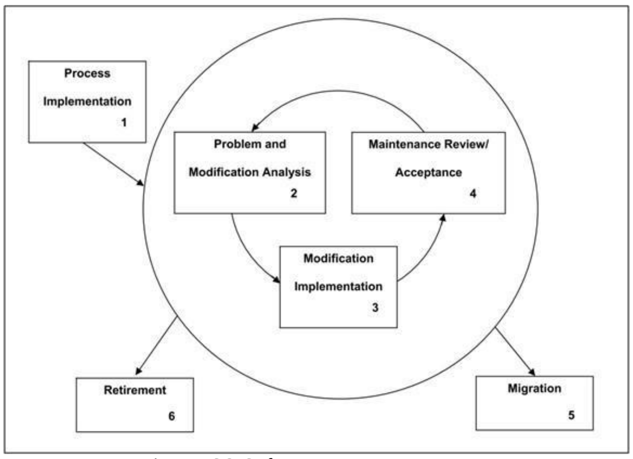

Process Implementation 1
Problem and Modification Analysis 2
Maintenance Review/ Acceptance 4
Modification Implementation 3
Retirement 6
Migration 5

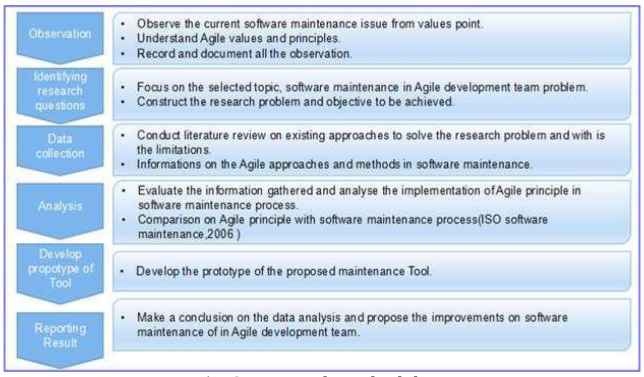
Fig. 1: ISO Software Maintenance Process.

From the literature review, we have identified six aspects that need to be considered to improve the software maintenance process in the Agile software development team.

1) Communication – an effective communication will ensure an effective solution to be delivered. Example, when an incident is reported, the software maintainer will discuss with the client and solve the issue fasten and effectively.
2) Repository, Knowledge management and Training – In order to work effectively, all the information of the application including requirement, design and implementation document must store in a repository. This is to improve the information seeking during the maintenance activity. Besides, the development team must provide regulate knowledge transfer and training whenever a new release deployment completed.
3) Documentation – Software maintainer must document all the changes that been done in the production and update in the repository for reference. Besides, the incident solutions also will be helpful to overcome reoccurring issue.
4) Cost effective maintenance and task management – from the existing paper, it is clear maintenance consume excessive cost and time. Thus, it is one of the main aspects we need to consider improving how we govern software maintenance process. Example is prioritizing the critical incident and allocating how much resource is needed.
5) Modification, impact analysis and testing - To solve any issue, software maintainer must make changes in production environment. Thus, modification and impact analysis are crucial factor to be considered.
6) Maintenance Tool – software maintenance tool plays significant role in improving the software maintenance process. For example, when all the information regarding the application, pass incident information and team information stored in one place, time taken for solve an issue is faster and effective

3. Methodology

This section describes the research method, which will be used in this study to accomplish the objectives (in section 1.2) of the research. The selection of research method is motivated. It also describes, which type of study is used to answer the research questions.



- Observe the current software maintenance issue from values point.
- Understand Agile values and principles.
- Record and document all the observation.

- Focus on the selected topic, software maintenance in Agile development team problem.
- Construct the research problem and objective to be achieved.

- Conduct literature review on existing approaches to solve the research problem and with is the limitations.
- Informations on the Agile approaches and methods in software maintenance.

- Evaluate the information gathered and analyse the implementation of Agile principle in software maintenance process.
- Comparison on Agile principle with software maintenance process(ISO software maintenance,2006)

- Develop the prototype of the proposed maintenance Tool.

- Make a conclusion on the data analysis and propose the improvements on software maintenance of in Agile development team.

Fig. 2: Research methodology.

4. Analysis and Interpretation

The ISO software maintenance processes [4] have been compared with the four core principles of the Agile development [3]. The main objective to make this comparison is to identify what are the differences between software maintenance process and Agile development team. It also helps to identify what is the lacking on applications handover to maintenance team by Agile team.

Table 1: Properties Comparison between Agile principle and ISO software maintenance.

Agile Principle	ISO practice	Mapping Finding
Individuals and Interactions Over Processes and Tools	ISO applies to planning, execution and control, review and evaluation, and closure of the Maintenance.	Agile Principle depends on individual more than process where else ISO software maintenance have list of process.
Working Software Over Comprehensive Documentation	The maintainer has to document the problem/modification request, the analysis results, and implementation options.	Agile Principle encourages light documentation where else in ISO software maintenance documentation is a core activity with configuration management.
Customer Collaboration Over Contract Negotiation	An estimate of maintenance costs. To provide cost-effective support to a software system	Agile Principle encourage iterations delivery approach over contracts where else in ISO software maintenance encourage cost estimation in any maintenance task
Responding to Change Over Following a Plan	ISO applies maintenance six main processes to follow. Process Implementation, Problem and Modification Analysis, Modification Implementation, Maintenance Review/Acceptance, Migration and Retirement.	In Agile principle, changes can be accepted at any stage, over the project plan where else ISO maintenance process need to follow the 6 primary stages.

To support our argument, we mapped the aspects with the related existing research papers and returned the following result.

Table 2: The mapping of six identified aspects with existing research.

Authors	Documentation	Repository/ KM/ Training	Teamwork	Cost estimation	Modification/ Impact/ analysis/ testing	Tool
[1]			Y	Y	Y	
[5]		Y	Y	Y		Y
[8]	Y	Y	Y		Y	
[21]		Y		Y	Y	Y
[10]		Y	Y		Y	
[9]		Y	Y		Y	
[11]	Y	Y	Y			
[12]		Y			Y	
[13]		Y	Y			Y
[14]			Y		Y	
[15]		Y		Y		Y
[2]		Y		Y	Y	
[16]			Y	Y	Y	
[17]	Y				Y	
[20]			Y		Y	

As depicted in Table 2, we can conclude that team communication and task management are definitely very important aspects in software maintenance. Testing and impact analysis also received many attentions which show its importance. Some researchers also agreed on the role played by knowledge transfer, handover process and training to be similarly important in maintenance activity. One of the Agile approach principle is working software over comprehensive documentation, hence there is no surprise to see less authors discussed about documentation. The least aspect studied is tool development for the maintainers. Thus, in this research, our interest would be to improve the way maintenance process in Agile development team is governed which comprises of all these highlighted aspects: documentation, repository /knowledge management or training, team work and communication, cost estimation /task management, testing and tool development.

5. Results

In this research, we have provided two contributions. Firstly, the Axita tool that is developed to assist the software maintainers in doing their maintenance activity.

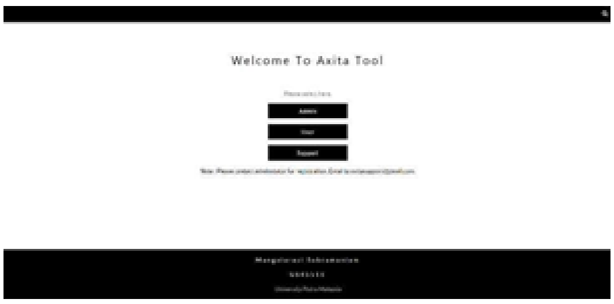


Fig. 3: The Axita tool main page.

Fig. 4: The Axita tool interface.

There are few main modules in the Axita Tool which includes home, task, project, report, repository, contacts, register, and calendar. The tool has 3 types of user, where end-user, administrator and maintainer. The task module is where incidents are logged and maintained. Project module is for the administrator to create new project and resource assigned. Report module to view the current reports for the incidents and repository to upload files and store information. Finally, calendar to remind about team activities, like meeting and events.

A case study has been conducted in operation team in Hewlett Packard Enterprise to test on the system. One incident from software maintenance team who adopt Agile during a project development in Hewlett-Packard Enterprise participates in the experiment. The team is supporting few finance application and a representative from the team is supported to test the Axita Tool. The team currently is using 4 tools to manage the maintenance task which include HPSM for ticketing, Sharepoint to store information, Runbook to keep pass tickets information and SVN to store the code. The incident that has been chosen to test the existing systems and Axita Tool is a data missing in application front end for Getpaid application.

The steps to solve the incident using existing Systems:

1. Login into the HPSM to check for the incident reported and assign to a resource.
2. Login into the Runbook to check is the incident is a re-occurrence issue.
3. Login into the SharePoint to check documents which are the table involve in the business process.
4. Login into application server to check the missing data in the database.
5. If still not able to check on the data, check into SVN and check the code for the front-end interface and process.

The steps taken to solve the incident using Axita Tool:

1. Login into Axita Tool. Check on the new incident reported in Task Tab and assign to a resource.
2. Filter closed incident in Task tab to check on re-occurrence issue.
3. Click on Repository Tab to get to check documents which are the table involve in the business process.
4. Login into application server to check the missing data in the database.
5. If still not able to check on the data, check into repository tab and check the code for the front-end interface and process.

We have done five rounds of test and eventually were able to come out with the following result.

Table 3: The result average time taken for round case study.

Steps	Time taken using current system (minutes)	Time taken using new system (minutes)
Step 1	3.5	7
Step 2	5.8	2
Step 3	5.1	3
Step 5	7.24	5

From the case study above, we can conclude that the Axita tool has improved the maintenance process by time taken to solve any incident. By reducing the time taken, the cost for the maintenance process also improved as cost is depending on the time. Axita Tool also has overcome the issue of the maintainer to refer many systems to solve any incident. The information can be found in one system, Axita Tool.

Secondly, we propose six best practices to improve the governance of software maintenance in Agile software development team. The best practices are as the following:

A. Communication between maintenance team, development team and customer.
   Weekly sync up meeting between the maintenance team and development team will help the teams to perform better. The 30 minutes meeting is mainly for maintainer to raise the concerns and doubts to the development team to understand the application better and satisfy customer needs. This can solve the time and cost problem in software maintenance process. Besides, maintainer also needs to communicate with customer, if possible face to face or through phone calls to better understand the user's need.
B. Training, Knowledge management and documentation from both development and maintainers team.
   Since agile development method encourages light documentation, software maintainers often facing difficulties to get the required information in solving any incidents. Thus, training and knowledge transfer modules must be planned and documented to avoid the missing information problem faced by maintainer. This included code storage and code with comments for the new comers to understand the application easier and faster. Pair programming also helps the understanding on the application.
C. Task management.
   This is to measure the progress of the team. These will ensure every maintainer, developer and management to see in real time exactly what is being done and who is assigned to what. This will help to prioritize the capacity of team on the task assigned. Task management also helps the team by estimating the time and cost spent through the maintenance service.
   Example:

60% planned incident: approved enhancement work in the project.
20% ad-hoc customer requests: slight changes and specific configurations.
20% other support and maintenance activity. (training and management).

D. Software maintenance management tool
   Having an incident management tools can help to store information and conduct report on the performance of the team. This can help to solve the maintainer's information seeking issues. All the documentation and project related information can be stored in the tool. This will also save a lot of time in solving the issue. The incident records will be stored as well in the tool for future reference. It can be a center point to gather all information.

6. Conclusion

The main objective of this research is to improve the governance of software maintenance process in Agile development team. The first part of the research is focusing on the maintenance process challenges and issues in Agile maintenance team. The main paper referred for software maintenance is international standard ISO / IEC Software Engineering — Software Life [4]. The main aspects to be improved have been identified from the literature which includes communication, repository, knowledge management, training, documentation, cost-effective maintenance, task management, modification, impact analysis, testing and maintenance tool. These aspects have been mapped with 15 existing research papers whose contribution was mainly in proposing solutions for agile maintenance. From the analysis, the result shows that most of the researchers focused in communication, repository, knowledge management, training, modification, impact analysis and testing. Meanwhile, lesser attention was given to documentation, task management and maintenance tool. Thus, this research is geared in improving the way maintenance process should be governed with all the aspects and the development of the Axita tool. The second part of the research was the development and testing of Axita tool. This tool is designed with the aim to establish a centralize data repository that makes reaching of information easier and faster. It also promotes an efficient task management that shorten the duration of solving the maintenance problems.

7. Future Work

There are several recommendations for improving the governance of software maintenance process in Agile software development team. Currently, the Axita tool was developed to support only web based. Hence, future work would be to expand its usage to mobile platform. The case study used to test out the tool was focused at only one maintenance team's experience. Therefore, we believe by extending the research case study to multiple teams can help to understand the maintenance process better in other contexts as well.

References

[1] Malhotra R & Chug A, (2016), Comparative Analysis of Agile Methods and Iterative Enhancement Model in Assessment of Soft-

ware Maintenance. *23rd International Conference on Computing for Sustainable Global Development*.

[2] David FR, "Agile Methods and Software Maintenance", (2007), available online: http://davidfrico.com, last visit: 9.7.2018.

[3] Beedle M, Bennekum AV, Cockburn A, Cunningham W, Fowler M, Highsmith J, Hunt A, Jeffries R, Kern J, Marick B, Martin RC, Schwaber K, Sutherland J & Thomas D, "Manifesto for Agile Software Development", (2001), available online: http://agilemanifesto.org/, last visit: 9.7.2018.

[4] International Organization for Standardization, (2006), "ISO/IEC 14764:2006 Software Engineering – Software Life Cycle Processes – Maintenance, available online: https://www.iso.org/standard/39064.html, last visit: 9.7.2018.

[5] Cohn-Muroy D & Pow-Sang JA, (2016), Situational Factors Which Have an Impact on the Successful Usage of an Agile Methodology for Software Maintenance: An Empirical Study. *Advances in Intelligent Systems and Computing 405*.

[6] McCalden S, Tumilty M, & Bustard D, (2016), Smoothing the Transition from Agile Software Development to Agile Software Maintenance. *Proceedings of the International Conference on Agile Software Development, in Processes in Software Engineering and Extreme Programming*, 209-216.

[7] Heeager LT & Rose J, (2015), Optimising Agile Development Practices for the Maintenance Operation: Nine Heuristics. *Proceedings of the Empirical Software Engineering*, 1762-1784.

[8] Harvie DP & Agah A, (2016), Targeted Scrum: Applying Mission Command to Agile Software Development. *IEEE Transactions on Software Engineering*, 42(5), 476-489.

[9] Heeager LT & Rose J, (2015), Optimising Agile Development Practices for the Maintenance Operation: Nine Heuristics. *Proceedings of the Empirical Software Engineering,* 1762-1784.

[10] Devulapally GK, "Agile in the Context of Software Maintainability: A Case Study", *Blekinge Institute of Technology Thesis,* (2015), available online: http://www.diva-portal.org/smash/get/diva2:868367/FULLTEXT02, last visit 9.7.2018.

[11] Stettina CJ & Kroon E, (2013), Is There an Agile Handover? An Empirical Study of Documentation and Project Handover Practices across Agile Software Teams. *Proceedings of 2013 International Conference on Engineering, Technology and Innovation & IEEE International Technology Management Conference*, 1-12.

[12] Yamato Y, (2015), Software Maintenance Evaluation of Agile Software Development Method Based on OpenStack. *IEICE Transactions on Information and Systems*, 98(7), 1377-1380.

[13] Laanti M, (2013), Agile and Wellbeing - Stress, Empowerment, and Performance in Scrum and Kanban teams. *Proceedings of the 46th Hawaii International Conference on System Sciences*, 4761-4770.

[14] Lang M, (2011), Agile Support and Maintenance of IT Services, Information Systems Development. *Asian Experiences*, 461-474.

[15] Hanssen GK, Yamashita AF, Conradi R, & Moonen L, (2009), Maintenance and Agile Development: Challenges, Opportunities and Future Directions. *Proceedings of IEEE International Conference on Software Maintenance*.

[16] Svensson H & Host M, (2005), Introducing an Agile Process in a Software Maintenance and Evolution Organization. *Proceedings of Ninth European Conference on Software Maintenance and Reengineering*.

[17] de Souza SCB, Anquetil N & de Oliveira KM, (2005), A Study of the Documentation Essential to Software Maintenance. *Proceedings of the 23rd Annual International Conference on Design of Communication: Documenting & Designing for Pervasive Information*, 68-75.

[18] Prochazka J, (2011), Agile Support and Maintenance of IT Services. *Proceedings of Information Systems Development*, 597–609.

[19] Abrahamsson P, Salo O, Ronkainen J & Warsta J, (2002), Agile *Software Development Methods: Review and Analysis,* VTT Technical Research Centre of Finland.

[20] Poole CJ, Murphy T, Huisman JW, & Higgins A, (2001), Extreme maintenance. *Proceedings of the IEEE International Conference on Software Maintenance*, 301-309.

[21] Tharwani S & Chug A, (2016), Agile Methodologies in Software Maintenance: A Systematic Review. *International Journal of Computing and Informatics (Informatica),* 40(4), 415-426.