Managing Software Maintenance Projects: Integrated Approach to Agile Methodology in Context of Large Investment Banks

Dr. R. K. Rana, Chancellor/Chairman, Madhav University, Abu Road, Sirohi, Rajasthan, India
Dr. H. L. Verma, Vice Chancellor, Jagan Nath University, Bahadurgarh, Haryana, India
Kevika Singla, Vice President, Project Manager, Royal Bank of Scotland, Gurgaon, Haryana, India

ABSTRACT

This paper proposes a methodology to maintain software projects laid down on systems that are prone to maintenance activities. Maintenance cannot be looked at detached from other software development processes as it plays a crucial role in the software life cycle. The cost of software maintenance is rising dramatically and it has been estimated that nowadays software maintenance accounts for around 80- 90% of the total cost of software, whereas it was around 50% a couple of decades ago. There are several reasons for this. We can see that software systems become hard to maintain over time, while every day more and more software is being produced. Documentation is lacking or incomplete and the people who know the software leave or retire without being replaced. Furthermore, systems tend to become increasingly complex. This makes extending a system difficult and costly. Rebuilding a system is usually not an option because the system that needs to be replaced is large, the test coverage unknown and the original and modified requirements are not well documented. So, the organizations are realizing the importance of spending huge investments in their systems in their daily activity through software projects and their systems must be continually adapted to the changing needs of the customers. They will be forced to implement the framework that can be extensively used and applicable for maintaining their software projects. This can be seen especially in large investment banks. From that, this paper presents the need of an integrated model approach that can be applied to various software maintenance projects to support the production of more maintainable software and hence reduce maintenance costs. The measures founded have been proposed to validate on an investment bank trading software project.

This paper highlights the need for unique maintenance measures and to develop an integrated approach to current agile approach of maintaining the software projects in large investment banks.

Keywords

Agile methodology, software development process models, software maintenance, process improvement.

1. INTRODUCTION

What is Software maintenance?

Organizations across all industries have huge investments in their software systems. These systems must be continually adapted to the changing needs of the organizations. Software maintenance and evolution refers to the process of modifying existing software systems to maintain their usefulness. In the world of software engineering, "***The modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment.***"[1]

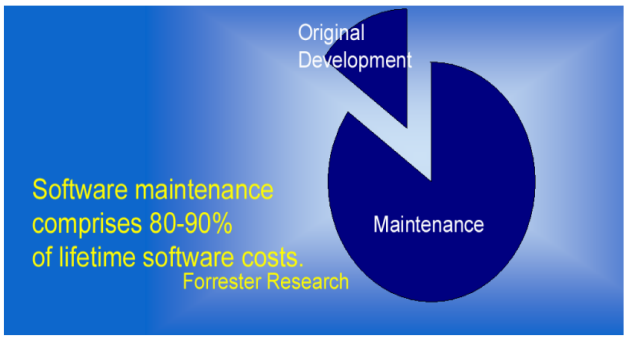

Figure 1. Software maintenance costs

Software maintenance is the concluding part of the software development process or lifecycle. Organizations across all industries have huge investments in their software systems. These systems must be continually adapted to the changing needs of the organizations.

Software maintenance has major economic importance of about 80-90% of lifetime software costs as shown in Figure 1 [2]:

*"A **software maintenance project** is a temporary endeavor having definite beginning and end to execute the activities that helps in maintaining the softwares of an organization to produce results desired by the customers of that software."*

2. KEY ISSUES IN SOFTWARE MAINTENANCE

Maintenance is considered to be the hardest part of SDLC and also considered a thankless job. Usually the maintainers are the unsung heroes of the software industry. Different researchers have found different problems in software maintenance [1, 2] presented in below Table 1:

Table 1. Problems in Software Maintenance

MANAGERIAL	TECHNICAL
Alignment with customer priorities and frequent changing priorities	Limited understanding
Alignment with organizational issues	Impact Analysis
Process issues	Insufficient documentation
Staffing	Difficulties in software comprehension
Organizational Aspects of Maintenance including Adapting to a rapidly changing business environment	Inadequate testing factors including regression testing
Outsourcing	Code duplication, Legacy code challenges and code complexity
Maintenance Personnel Turnover	Software aging
Cost and Estimation	Lack of tools and graphical user interface available for maintenance
	Software Maintenance Measurement
	Large backlog
	Performance measurement difficulties
	Ripple effect due to change at one place and impacting too many related areas which might be unknown

3. WHAT IS AGILE METHODOLOGY

Agile development is a group of iterative software development methodologies that includes Extreme Programming or XP for short, Scrum, Standard & Poor's, Feature Driven Development, Crystal, and Adaptive Software Development. In 2001, the ''agile manifesto" was written by the practitioners who proposed many of the agile development methods. The manifesto states that agile development should focus on four core values [3]:

- Individuals and interactions are valued over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Below Figure 2 depicts the agile development process:

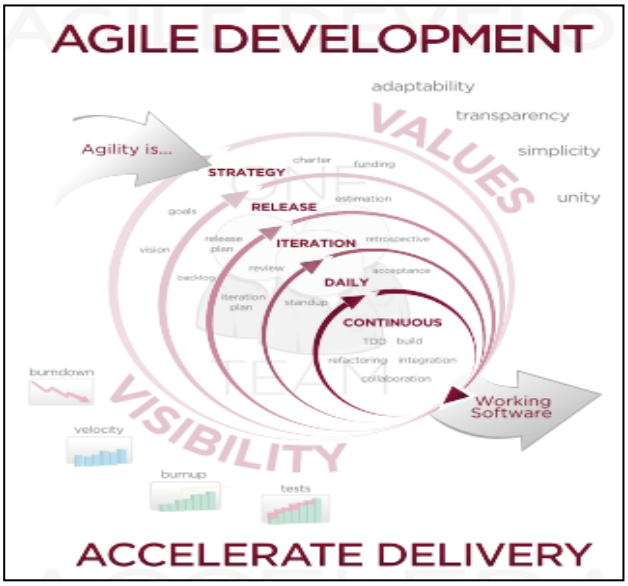

Figure 2. Agile Development

4. MAINTENANCE IN AGILE METHODOLOGY

While agile techniques vary in practices and process, they share common characteristics, which include development in small iterations, focus on interaction, communication, and the reduction of resource intensive processes. It is a software development paradigm that allows for tight collaboration between teams, intense customer involvement, and immediate feedback. Highest priority is

given to satisfying the customer though early and continuous delivery of valuable working software. This is a rather new approach and not much has been written about it with regard to maintenance. Maintenance does not belong to the practices of say Extreme Programming in Agile approach and is not explicitly embedded in the process. However, it is contained somehow because it is interesting to note that [Beck 2000] [4] says that "**maintenance is really the normal state of an Extreme Programming project" as maintenance contains naturally many characteristics of Extreme Programming like user driven, quick responses through small increments**. Nevertheless, it is not regarded as code and fix but as a proper engineering method. Furthermore, the consulting company ClearStream argues that software developed through XP is easier to maintain as it is "seen by many eyes, understood by many brains, verified by many tests" [ClearStream 2001] [4]. One of the practices of agile models that support this statement is **testing**. This practice ensures that occurring errors, at least most of them, are discovered quickly. Corrective maintenance takes place immediately and continuously to prevent or reduce the need for that category of maintenance after the product has been delivered to the customer. Another practice, which ensures that one can continue maintaining a software project easily in the future, is refactoring. From a maintenance perspective, it belongs to the category of perfective & preventive maintenance as it is described as "restructuring the system without changing its behavior" [Beck 2000] with the objective to "improve code maintainability and stability" [Poole and Huisman 2001] [4]. Nevertheless, in the same article [Poole and Huisman 2001] state that if the initial development does not follow XP practices, especially the application of coding standards and simple design, refactoring according to XP cannot be utilized. In those cases, a "wholesale reengineering" effort is necessary before refactoring and "extreme maintenance" can be applied to the software project.

5. COMPARING TRADITIONAL AND AGILE SOFTWARE MAINTENANCE PROJECT

The need for business to respond rapidly to the environment in an innovative, cost effective and efficient way is compelling the use of agile methods to develop and maintain software projects. According to the surveys done by Shine Technologies, the Standish Group and Cutter Consortium have shown that the percentage of companies using agile methods have increased each year [5]. Just as mobile phones have reduced the need for telephone landlines agile methods are reducing the need for heavyweight methodologies. The future of agile methodologies seems very dominant. In general, there are some aspects of software maintenance project that can benefit from an agile approach. There were many studies with cost and benefit data for both of these approaches, and found that agile approach has average of 200% better performance than big and expensive Traditional Methods [6].

Table 2. Comparison between Scrum and Traditional Project Management

Parameters	Scrum	Traditional Project Management
Emphasis is on	People	Processes
Documentation	Minimal - only as required	Comprehensive
Process style	Iterative	Linear
Upfront planning	Low	High
Prioritization of Requirements	Based on business value and regularly updated	Fixed in the Project Plan
Quality assurance	Customer centric	Process centric
Organization	Self-organized	Managed
Management style	Decentralized	Centralized
Change	Updates to Productized Product Backlog	Formal Change Management System
Leadership	Collaborative, Servant Leadership	Command and control
Performance measurement	Business value	Plan conformity
Return on Investment	Early/throughout project life	End of project life
Customer involvement	High throughout the project	Varies depending on the project lifecycle

6. LIMITATIONS WITH AGILE APPROACH OF MAINTAINING SOFTWARE PROJECTS

However, agile methodology poses some limitations in maintaining the software projects specifically in large organizations.

- **Inefficacy in large maintenance projects -** Agile management methods fail when applied to large organizations and large software maintenance projects as its non-sequential method of working starts getting beyond control with too many members and voices. It is observed that agile practices in large organizations tend to be chaotic and stabilize only when a plan driven approach accompanies it.

- **Transition from software based projects -** One of the major hurdles to cross for agile projects is the crossover from software based maintenance projects to enterprise based maintenance projects. The principle it is based on has been developed by software engineers for software projects, and when projects are not software based people hit dead ends.

- **Software Product Owners** - A lot of agile maintenance projects tend to focus on software product owners and the ones who are not highly focused with the concerned projects have the burden of holding two full time job positions. The product owners pet agile project can work but not if that person is appointed specifically for it. Since it's the product owner's role to significantly represent the customer in the project appointing more than one person could be a solution.

- **Scrum fails to get traction or is a distraction from the real work of the project -** In order to be effective, a Scrum Master and as many team members as possible must have start-to-finish experience with team projects of enough duration to have had scheduling delays, non-project distractions, and requirements drift, among other things. Six months is usually long enough to have experienced at least some of these issues, but a longer duration has a way of compounding them and challenging the team even more. This experience has a way of making the value and purpose of agile practices vividly clear. It helps to have worked on a number of waterfall life-cycle projects and to have been frustrated by the impedance mismatch between how they were managed and how developers actually work.

- **Developers accustomed to working autonomously may find that Scrum is unnecessary and slows them down -** There is no question that Scrum adds some overhead to the maintenance process, as compared to a development process with no formal methodology. By design, Scrum is a management tool for agile projects; intended to give management a meaningful view of the health of the project, and the ability to make management decisions about how to proceed. This, unfortunately, entails some amount of overhead.

- **Some development efforts don't easily fit into a time-boxed sprint -** This is a real problem. Several kinds of maintenance projects resist being meaningfully squeezed into standard size sprints. Also, reduction in formal communication in those projects could lead project teams to discount system.

- **Design changes during the iterative sprints** - The maintenance project started with minimal business requirements from the business and the list of features required were identified. The project included redesigning the underlying data model of an existing application and enhancing some features identified to the application. The project did not have the detailed requirements at the start and the design, data model, build and testing started simultaneously following agile approach. As progressed through sprints, any design or data model changes, while developing the features down the line, eventually resulted in rework on the features completed in earlier sprints. The finished products were revisited every time there was a change in design which increased overall cost and caused delay in progressing with the remaining open activities. Hence, it would be challenge to complete the sprints on time if design is evolving across sprints.

- **Team member sharing while sprint is in progress** – The projects had a dedicated skilled team which was a key strength to get the development on time and of high quality. The maintenance project deployment was planned into multiple deployment phases so that the finished products can be released earlier. There were cross impacts due to other inflight project and the part of the development team was deviated to focus on sharing the knowledge of the project and its impact on other projects so as to enable first deploy release to go ahead as planned. As per Agile guidelines, the core sprint development team is expected not to be deviated from the locked down list of features. But

in the real world, the knowledge of the skilled team is required to mitigate any risks related to that project. There are delays on the sprint that was in progress in order to accommodate risk mitigation activities.

- **Requirements unclear even while sprint is in progress** – Agile framework facilitates the business SMEs to change or finalize requirements while the feature is being developed. But in order for the development team to complete the development within the sprint time, the business should be clear on the requirements for the features in the current sprint to enable on time completion. Ambiguous requirements, while sprint is in progress, lead to delays in completing the feature within the sprint and may result in increased backlog. Hence, agile practices work if performance requirements are made explicit early in the maintenance project. The business subject matter experts have to have the requirements clear to enable the team to complete the project on time.
- **Documentation -** Extensive documentation is not encouraged which makes later stages in the software development cycle such as maintenance harder to implement in that project. While Agile framework accepts less documentation, this needs to understand that the main objective of agile framework is quick delivery. Documents that enable quick delivery and better control on time and cost are still required in agile project management which becomes challenging due to time constraints.

Table 3[7] provides an overview of an Internet and document search on the problems, challenges, and issues with agile project management approaches. These challenges can be grouped into four areas and contain just under 50 situations faced introducing or using agile methodologies. According to the VersionOne survey, organizations that have gone agile see some of these challenges as barriers to the further agile adoption.

Table 3. Agile challenges and problems

Communicating, changing culture and mindset	Buy-in from management, customers, and team members	Day-to-day operational problems	Experience Making it Work
• Communication problems - Too narrow communication bandwidth; Communication between development and product owner and Communication between development and quality assurance • Company culture • Lack of culture transition • Organization change • Mindset	• Management buy-in • Only developers engaged • Big visible charts are intrusive • Insufficient training Team does not like showing work • Unwillingness of team • Customer not signing off • Customer fear Customer won't commit to agile • External pressure • Getting customer into loop Getting stakeholders to agree • Departmental fragmentation • Lack of management awareness • No single product owner authority • Management not changing behaviour	• Back loading documentation • Back loading testing • Decreased visibility of project progress • Effort estimations • Good requirements development • Integration with other systems • Interruptions Lack of decisions on architecture Lack of time to fix failed tests • Large projects • Project complexity • Too big backlog • Too old backlog • Too many meetings • Too many open issues • Too many unplanned tasks • Getting test automated • Regression testing	• Budgeting • Giving-up too early • Inexperienced Product Owners • Inexperienced Scrum Masters • New to agile • Process not understood • Reinvent agile methodology • Scaling to large projects

There was also a research conducted [8] to explore around the various weaknesses which must be considered before going head first into software development or maintenance, so that the quality is not compromised. The major advantage of agile methodologies is the active participation of the customer or user throughout the development lifecycle but this can also lead to the major weaknesses. Sometimes Customer do not have the time to interact. Agile methodologies generally use small teams to develop their projects which can sometimes make it challenging to complete large projects. Team members need to be at the same location throughout their work, but this can get difficult as it is not possible for those teams that work on the different projects and are far away from each other to come together and work at the same physical location. This makes coordination difficult. Also as requirements can be added any time, this will lead to the never-ending project. Miscommunication is the major factor that leads to the problem during the implementation of the agile methodologies in the software development lifecycle. Testing is conducted throughout the software development lifecycle therefore it requires the testers to be at the same place during the lifespan of the project development. This will unnecessarily increase the resources of the project and increases the overall cost. It becomes difficult to find the pace for the software development. The overall weaknesses of agile are summarized in Figure 3 [8]:

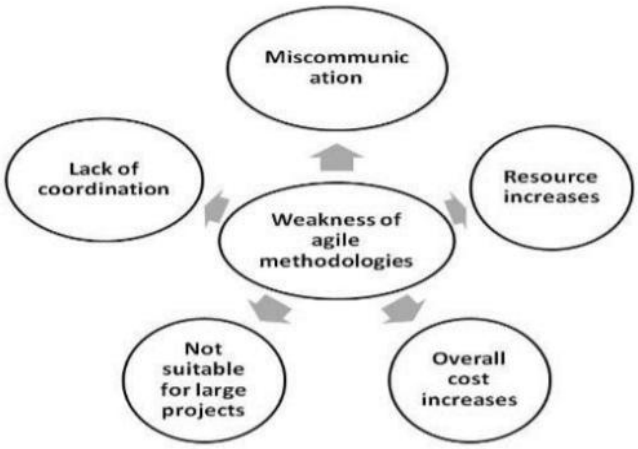

Figure 3. Agile Methodology weaknesses

Selecting and Customizing the Right Methodology

Problems with the cultural transition, management buy-in and only engaging the developers can be avoided or at least mitigated by selecting and customizing a methodology that fits the company and the project. Hence there is a need to integrate a new approach to existing agile approach which will help maintaining the software projects efficiently. So, this paper will study **an integrated model which will act as an enhancement to agile methodology to solve various problems encountered in managing the software maintenance projects especially in context of large investment banks.**

7. PROPOSED WORK

The goal of this proposed research is to discuss the common problems encountered while executing the processes of software maintenance projects, and to investigate positive developments in the field which includes integrating some approaches in the current agile methodology for maintaining the software.

The research will be logically studied in four parts, to help conclude an integrated approach to agile methodology in maintaining the trading system software projects of large investment banks:

1. Analyze the current agile approach for software maintenance projects and limitations to it. It includes investigation of maintenance activities in the context of software maintenance projects and their applicability and usage in practice.

2. Overcome these limitations and suggest further improvements that will build a collaborative framework of maintenance activities. An integrated methodology of maintenance activities will be established that will further improve on agile approach of maintaining the software projects. The research work will touch base each maintenance activity and go into depth into some of the maintenance activities which can be embedded with various techniques and tools for making them more efficient and thus increase speed and reduce cost of maintaining software.

3. Estimate the overall improvement of efficiency, speed and ROI of changes and thus reducing costs, by identifying key problems, promising solution strategies and topics of importance. This will measure the output of software maintenance project activities.

4. Special reference to maintenance projects of trading softwares would be provided which is in the context of large investment banks. The research work will explain the organizational context and processes in which different maintenance models

are applied, which activities and techniques are used and then apply the integrated maintenance model agile approach for better maintenance of their trading softwares.

The research work will be carried out in five stages.

- Stage 1: Comprehensive literature review, with special focus on theories and material related to improving agile methodology for software maintenance.
- Stage 2: Data collection (industry wide and company specific)
- Stage 3: Specific administration of a detailed questionnaire
- Stage 4: Analysis and Interpretation
- Stage 5: Recommendations and validation

8. CONCLUSION AND FURTHER RESEARCH

This paper is a step towards proposing a new integrated project methodology approach to maintain the trading software projects of large investment firms. The methodology is required due to existing problems of using the plain agile approach in software maintenance projects (read in this paper) and hence can be overridden by adopting to new processes, tools and techniques under the integrated project management methodology.

The future aspect is to find out the right framework that can be extensively used and applicable for managing bigger projects in large investment banks. Since investment banks have been experiencing rapid growth and expansion, necessitating technology changes are required to maintain a competitive edge. Hence, the further research will try guiding the effectiveness of new integrated methodology in terms of varied parameters like project, cost, timeline and schedule. It will also try to aim the development projects besides the maintenance projects and generalize the project management methodology approach to manage all type of projects.

9. LIMITATIONS OF THE STUDY

1. Due to time constraints, the research might not be able to cover all aspects of real life situations in this field.
2. The validity of results will be limited to set of project management methodologies that can be studied in context of maintenance projects of an investment bank.
3. As the study results are based on the limited number of responses so they are not enough for broad generalization. The findings can be refined and made more robust by replicating the study on large sample of population.

REFERENCES

[1] https://en.wikipedia.org/wiki/Software_maintenance

[2] http://sce.uhcl.edu/helm/SWEBOK_IEEE/data/swebok_chapter_06.pdf

[3] T. Dyba and T. Dingsøyr. "Empirical studies of agile software development: A systematic review.", *Inform.Softw. Technol*. (2008), doi:10.1016/j.infsof.2008.01.006

[4] Urs Kuhlmann. "Maintenance Activities in Software Process Models: Theory and Case Study Practice.", *University of Koblenz Landau, Institute of Software Engineering 2003*, Master Thesis

[5] M. A. Awad. "A Comparison between Agile and Traditional Software Development Methodologies.", *The University of Western Australia, 2005, School of Computer Science and software Engineering*, Report

[6] https://medium.com/@SCRUMstudy/scrum-vs-traditional-project-management-60f15f1e63f

[7] Miller, G. J. (2013). "Agile problems, challenges, & failures.", *Paper presented at PMI® Global Congress 2013*—North America, New Orleans, LA. Newtown Square, PA: Project Management Institute

[8] Sandhya Tarwani and Anuradha Chug. "Agile Methodologies in Software Maintenance: A Systematic Review", *Informatica*, Vol 40 No 4 (2016), 415–426.