Applying Agile Methodologies in Industry Projects: Benefits and Challenges

Kamaljeet Kaur, Anuj Jajoo, Manisha
Education, Training and Assessment, Infosys Limited
Kamaljeet_Kaur01@infosys.com, Anuj_Jajoo@infosys.com, Manisha@infosys.com

***Abstract*— Agile software development has been prevalent in IT industry since more than two decades now. Conceptually, Agile was introduced as an iterative, incremental and adaptive methodology for software development. There are quite a few studies that discuss the success of agile development projects on account of multiple factors like cost, time, quality and productivity. Owing to its popular virtues like enhanced flexibility to incorporate evolving requirements, incremental delivery, quick time to market and ability to keep pace with market trends [1], clients tend to leverage agile methods for a variety of project types in addition to typical small scale development projects. Through this paper, we discuss some of the practical aspects, pros and cons of applying agile principles to different types of IT projects including testing projects, maintenance projects and large-scale development projects.**

Keywords*— *Agile; projects; implementation; large scale development; testing; maintenance

I. INTRODUCTION

Agile methodologies is a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams and customer [2]. It is a time-boxed iterative approach, and encourages rapid and flexible response to the changes.

While there are many methodologies that fit under the banner of agile like Scrum, XP [3], DSDM [4], Lean [5], Crystal [6] etc., scrum is one of the most widely used agile framework and same has been taken into consideration here as well. Scrum is an iterative and incremental agile software development framework with a focus on flexible & holistic product development strategy where the development team works as a unit to reach a common goal [7].

The Scrum framework fits quite well on small scale development projects, where the team is small in size and co-located and has an easy access to client. However when we try to apply it on following variations of IT projects, we encounter certain challenges due to the inherent nature of these projects.

1. Development projects- The software projects where majority of the work involves developmental activities like requirement analysis, design, coding and testing. They can be broadly categorized into two types based on their requirement size, dimensions and expanse:
   a. Small/medium scale projects
   b. Large scale projects
2. Testing projects- These are the projects where majority of the work revolves around testing activities like black box testing, integration testing, system testing, regression testing and automation of testing process.
3. Maintenance projects- The projects where software is already in use by end users and the majority work revolves around maintenance activities pertaining to customer support and/or minor/major enhancements, which need modification and refinement to the current functionality.

We shall now discuss the application of agile principles in each of these project types. The actual names of clients and companies have been changed for confidentiality purposes.

II. APPLICATION OF AGILE IN LARGE SCALE SOFTWARE DEVELOPMENT PROJECTS

A. Project Background

The project we are referring to here is an online learning portal for a national open university Y which is affiliating 1000 courses across country. The intent of the creating the e-learning portal is to facilitate the online as well as offline web sessions and artifacts for students.

The university had shared the following initial high level requirements with Company X.

- The student can register for a program and course and attend online sessions.
- The recordings and study material for all the sessions to be made available based on the courses opted by a student.
- College admin to manage student registrations, schedule sessions, upload recordings and content for courses.
- University admin has an access to all the colleges and can schedule sessions at university level.

Agile project development approach was adopted to fine tune requirements as the project progresses. The first phase of the project was to be released in 60 days for 100 courses in 8 states of North India.

B. Project Development Methodology

1) Applying Scrum

Owing to the massive project requirements in each sprint, there was a need to scale up the scrum team and perform scrum activities in distributed fashion:

a) *Scrum Roles*

i. The chief product owner from the University and representatives from affiliated colleges formed a team of product owner.

978-1-4799-6892-3/15 $31.00 © 2015 IEEE
DOI 10.1109/ICCUBEA.2015.166

ii. There were multiple scrum teams, based out at different geographies.
iii. Every scrum team was led by a scrum master. Super scrum master was the marshal of all scrum masters.

b) *Scrum Ceremonies and Meetings*

i. Each scrum team had its separate scrum ceremonies.
ii. Scrum of scrums was conducted amongst the scrum masters on daily basis using video conferencing to address dependencies and enhance coordination.

c) *Scrum Artifacts*

i. Product backlog was split based on functionalities and modules. The related user stories were assigned to different scrum teams.
ii. Each team managed its own sprint backlog.

The scrum roles and artifacts are shown in *Fig. 1.*

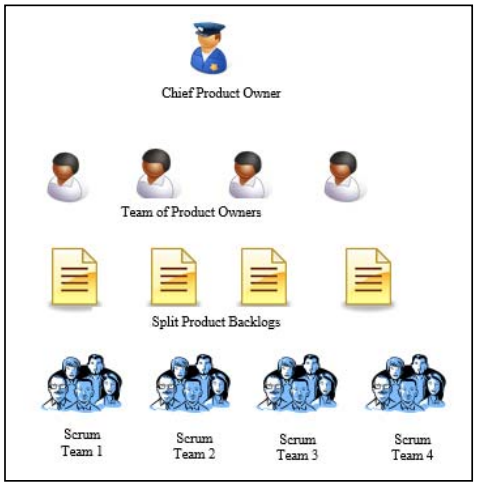

Fig. 1. Scaling Scrum

2) Challenges

- Large Scale agile-The introduction of additional scrum roles like chief product owner, super scrum masters and multiple scrum teams resulted in additional scrum ceremonies like scrum of scrums and distributed artifacts.
- Distributed agile- Teams had to rely on mechanisms like video conferences and online meetings to conduct various scrum ceremonies. Differences in time zones can further toil the situation.
- Delayed client communication- Inputs from product owners were delayed as they needed to synchronize amongst themselves before responding to scrum masters.
- Short Sighted View- With the increase in online courses and artifacts, the performance of the system was hit at later stages of the project. This aspect accentuates for large scale projects as the bigger picture is not thought of during initial sprints.
- Reduced team coordination- Owing to distributed teams, the coordination and resource sharing amongst the teams was also a challenge in itself.

3) Benefits

- Working Software- Since the e-learning portal was a large scale project, we could start with smaller achievable modules. This adds to the confidence in developing large applications gradually.
- Continuous client involvement- On account of regular client involvement and signing off the incremental builds, the product evolved in line with changing requirements and market trends.
- Reduced project failures- Involvement of client during the entire project lifecycle leads to lesser project failures.

Though for large scale development projects we need to enhance various agile practices, still adopting agile methodology is definitely a better approach for small as well as large scale distributed development projects.

III. Applying Agile to Testing Project

A. Project Background

XBank, a regional bank in Europe wanted to extend its services in an online mode to its customers. They wanted to reduce the manual workload by providing net banking facility to the customers. Considering the amount of work and risks involved, the bank management decided to outsource the development work to YSoft and testing job to ZSolutions.

B. Project Execution Methodology

In order to reduce time to market and keep pace with the competitors, it was decided that the model of development should be Agile and new features would be published on bank website every quarter. Every release was further divided into 3 weeks sprints: 2 weeks of development and 1 week of testing and bug fixing.

1) Applying Scrum

- The project began with a user story workshop attended by the developers from YSoft and the testers from ZSolutions. Product backlog was created as per the priorities given by the product owner (i.e. bank representative)
- Based on the scrum development release plan shared by YSoft, the testing team at ZSolutions formulated their scrum test release plan in a staggered manner as shown in *Fig. 2*.

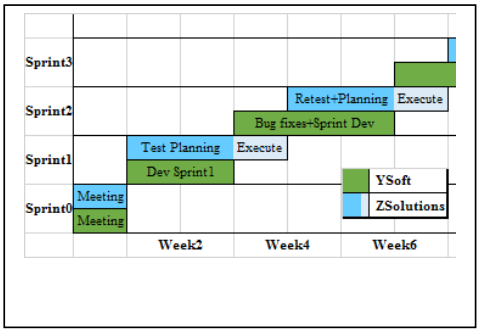

Fig. 2. Agile Test Planning

- After every two weeks YSoft would release the unit tested code; the testers at ZSolutions would then execute various test cases. The identified defects were taken into subsequent development sprint along with planned user stories at YSoft.
- Test strategy- Since testing needs to be done in a time bound fashion i.e. as per the sprint timelines, the challenge was to understand how much testing would be enough. So, "risk-based" testing was adopted.
- Risk identification and risk factor assignment: During sprint planning meetings, the product owner and the testers would brainstorm the risks associated with the user stories under development.
- Next, the risk mitigation was discussed. Risk mitigation involved identification of quality criteria which the application should meet to avert the said risk. Quality criteria would further lead to the generation of one or more test cases as shown in *Fig. 3*. For Example- Fund transfer related user story requires the quality criteria that transactions involved in money transfer from one account to the other should be atomic. Thus, the test cases which were generated from the various user stories (from the same sprint) formed the sprint backlog.

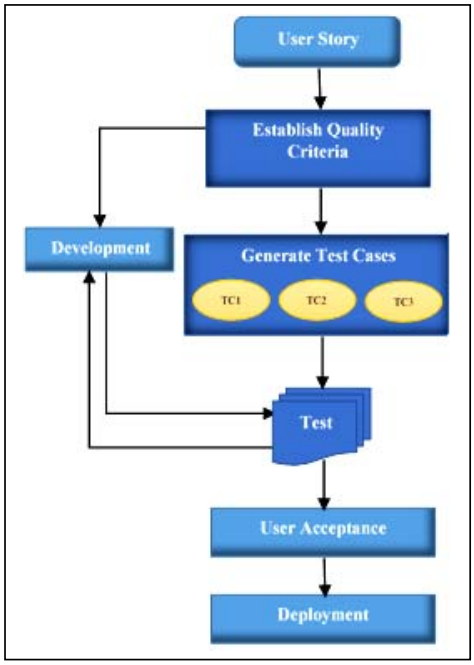

Fig. 3. Agile Test Cycle

- The risk-based testing strategy helped testers to determine how to spend their testing efforts wisely. The high priority test cases were executed first, thus making testing more efficient.
- At the time of automation of regression testing the priority list was followed. The high risk bugs were identified and fixed early.
- Pair testing was also helpful for checking quality of critical user stories. In pair testing, along with the tester, the product owner was involved. Together, they introduced their perspectives and experiences. This led to "exploratory testing".
- The scrum ceremonies like daily scrum, review and retrospective meeting facilitated a common understanding among all the testers.
- In addition, two of the testing team members were at onsite to ensure the software integrates well with the existing banking application in real time before going live.

1) Challenges

- Velocity mismatch- After some sprints, the project development velocity increased as the developers were more comfortable with the environment; however, the testing velocity decreased due to increase in the test cases under regression testing.
- Dependency- Since testing depends on the development completion, many a times the delay in development resulted in the delay in testing.
- Spread User Story- In cases where some part of user story was spread across two development sprints, testing team could not complete the black box testing and had to wait until the completion of next sprint release from the development team.
- Self-Managing team- Since the application was tested in parts, it was important for the team members to be self-managing and to have a constant focus on the bigger picture. At times, this was challenging for less experienced team members.

2) Benefits

In general, following are the advantages of discussed agile testing:

- The agile testing is no more a separate testing phase and follows the philosophy of "plan as you go", thus supporting the flexibility for product owner to change the priority of user stories during any of the sprint meetings.
- Since the risks are identified at early stage, the product owner can take the testing feedback and reassign the priorities or fine tune the user stories.
- Because the same development team is assigned for bug fixing, the time taken for resolving the bugs is reduced significantly in agile testing.
- Testing team may need certain time bound licensed tools (or paid add-on components of a tool) for testing (for example- Load Runner). The procurement of the licenses can be postponed until the sprint when it is required. This helps in saving some cost saving when expensive licenses are involved.

Summarizing- with agile testing, the automation becomes more significant. A right balance of manual and automation testing can lead to optimum quality check for the software application in less time. Pair testing, risk based testing and exploratory testing are some of the key practices to achieve the goal of quality control.

IV. Application of Agile In Software Maintenance Projects

A. Project Background

This case is about an online banking software maintenance project where a state bank in US serving around 1 million online customers wanted its internet banking services to be maintained by company X.

High level requirements on following lines were shared by the Bank:

1. Addressing system failure and customer complaints.
2. Resolution of issues raised by online customers at the Bank's web portal.
3. Regular business analytics reports to be shared with Bank's head office.
4. Regular customer database backup.
5. Responding to customer queries via email and toll free numbers shared at Bank's web portal.
6. Besides above, following enhancements needed to be implemented:
   - UI enhancements to improvise user experience.
   - Migration of customer database from Oracle to SQL Server.
   - Performance improvements in screen refresh functions.
   - Adding UI support for color blind customers.

B. Project Execution Methodology

Conventionally, maintenance projects have following life cycle:

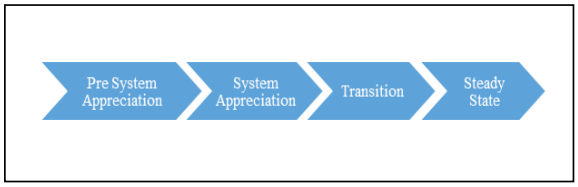

Fig. 5. Conventional Maintenace Project Life Cycle

As during the project deal discussions, Agile Scrum was decided as the methodology for the project execution, it was a challenge in front of team at Company X to map the scenario to agile life cycle. While they had some experience on agile development project, a maintenance project in Agile was altogether a new ball game.

1) Applying Scrum

- Pre-system appreciation, System appreciation and Transition were carried out in conventional way as they were meant more for understanding the system and taking charge.
- Scrum cycle started from Steady State phase where requirements were classified in two major buckets:
  - Support requirements i.e. Req. 1 to 5
  - Enhancements i.e. Req. no. 6

  Two scrum teams were formed with respective Scrum masters in order to handle the above in a streamlined fashion. The bank designated their IT department's head as the Product owner for the both the Scrum teams.
- For Scrum team 2, it was like a small scale development project where Product backlog contained the enhancements and a Scrum release plan was created based on the priorities given by Product owner and available capacity of the team.
- Scrum team 1 had a mix of requirements that were repeatable in nature like req. no. 3 and 4, and real time requirements like req. no. 1, 2 and 5 which were sporadic in nature.
  - For repeatable requirements, details were formulated with Product owner, estimations carried out and based on that, a release plan was created where each sprint had tasks for regular business analytics reports and database backup. However, effort buffer was left in each sprint to accommodate sporadic or run time requirements.
  - Sporadic requirements like addressing system failure and customer complaints posed a real challenge in terms of applying Scrum framework. The primary issues encountered are mentioned in brief below:

2) Challenges

- Estimation- The effort needed to fix the complaint or respond to a customer query would depend on the issue received, so it was not possible to do estimations in advance. Without estimates, sprint planning had no significant base.
- Sprint length- Iteration or Sprint length couldn't be fixed easily.
- Issue escalations- In order to take advantage of time-bound principle of agile, team started with a 1 week sprint, where issues received till sprint beginning were collated in an Issue Tracker. The set of issues mapped to a sprint backlog and story-points were assigned to them during sprint planning, followed by their resolution in sprint execution. Still, escalated customer issues would disrupt the sprint execution in-between.
- Issue prioritization- Most customer service requests had associated SLAs which had to be kept in view while planning daily/sprint tasks.

3) Benefits

Nevertheless, some of the agile practices still served useful purpose for Scrum team 1 (in addition to Scrum team 2):

- Daily scrum- Daily scrum meetings to discuss issues addressed, plan for current day and any impediments
- Visual Board- Visual task board displaying In-progress, completed or reopened issues.
- Client collaboration- Requirement workshop with Product owner for understanding complex issues. Usage of Wiki for collaborating with the Bank's IT department also led to timely issue resolution.

- Issue categorization- Rather than fixing the issues as they come, sprint planning activity facilitated analyzing the issues in a comprehensive and structured manner.

In brief, from the point of view of application of agile principles and practices, above project's analysis and experience shows that requirements of a typical maintenance project can be divided into following categories:

1. Enhancements– can be considered like a development project and agile practices can be applied overall.
2. Regular Support Services– requirements that are repeatable in nature e.g. database backup, generation of reports etc. Again, Agile can be applied to a great extent.
3. Run Time Issue Handling– requirements pertaining to run time issues, bugs and complaints. Customer issue severity at run time remains the driving force and agile can be applied to a limited extent.

V. CONCLUSION

As discussed in the beginning, there are multiple flavors of Agile like Feature Driven Development, Kanban, XP, etc. Kanban, due to its inherent nature being task oriented is better suitable for maintenance projects as compared to Scrum. On similar lines Feature Driven Development [9] is more often used for product development where multiple versions of the same product exist. For complex module implementations-pair programming principle of XP is applicable as it facilitates features being released in short time.

The table below summarizes the applicability aspects of agile principles on different IT projects:

TABLE I. APPLICABILITY OF AGILE ASPECTS ON IT PROJECTS

S.N.	Agile Principle	Applicability of Agile Projects on		
		Large Scale Development Projects	***Testing Projects***	***Maintenance Projects***
1	Welcoming changing requirements	High	Medium	Low
2	Incremental Project delivery	High	Medium	Low
3	Application of Scrum Ceremonies,Teams and artifacts	Medium	Medium	Medium
4	Active client involvement	Medium	High	High
5	Teams' empowerment to take decisions	Medium	Medium	Low

To conclude, Agile might not be as perfect a fit for testing, maintenance or distributed & large scale development projects, as it is for small & co-located development projects, but it is indeed a powerful methodology that can enable teams to improve productivity, enhance visibility and achieve higher customer satisfaction. It essentially gives freedom to project teams for adaptive planning and evolutionary development. Through this paper we have discussed various challenges and benefits of using agile approach for large scale development, testing and maintenance projects.

REFERENCES

[1] Beck, K.; Beedle, M.; Bennekum, A. van; Cockburn, A.;Cunningham, W.; Fowler, M.; Grenning, J.; Highsmith, J.;Hunt, A.; Jeffries, R.; Kern, J.; Marick, B.; Martin, R. C.;Mellor, S.; Schwaber, K.; Sutherland, J.; Thomas, D."Manifesto for Agile Software Development. 2001."Available at: <http://www.agilemanifesto.org/>. Last access: 16 abr.2012.

[2] Manisha, Manoj Manuja, "Moving Agile Based Projects on Cloud" ,2014 IEEE International Advance Computing Conference.

[3] Lindstrom, L.; Jeffries, R. "Extreme Programming and Agile Software Development Methodologies." Information Systems Management, v. 21, Issue 3, pp 41-52. 2004.

[4] DSDM http://en.wikipedia.org/wiki/Dynamic_systems_development_method

[5] Lean http://en.wikipedia.org/wiki/Lean_software_development

[6] CrystalClear http://en.wikipedia.org/wiki/Crystal_Clear_(software_development)

[7] Schwaber, K.; Beedle, "M. Agile Software Development with SCRUM." Prentice-Hall. 158 p. 2001.

[8] Jenkins http://jenkins-ci.org/

[9] Kuda Nageswara Rao, G. Kavita Naidu, Praneeth Chakka, "A Study of the Agile Software Development Methods, Applicability and Implications in Industry" , 2011 International Journal of Software Engineering and Its Applications.