Journal of Computer Science 8 (3): 389-392, 2012
ISSN 1549-3636

Lessons Learned in Transforming from Traditional to Agile Development

[1]Ganesh, N. and [2]S. Thangasamy
[1]Department of Computer Science and Engineering,
Anna University of Technology, Coimbatore, India
[2]Department of Research and Development,
Kumaraguru College of Technology, Coimbatore, 641 049, India

Abstract: Problem statement: This article presents the biggest challenge that the organization faces in transitioning the mindset of the team from that of a waterfall model to an agile thought pattern. **Approach:** The study is conducted from a real time live project, carried out in a software organization. **Results:** The software team found a major difference in their work culture resulting in collective ownership, forming a balanced self organized team, getting frequent feedback from the customer and making continuous deliverables. **Conclusion:** The main finding when implementing an agile software development is to respond to the changing needs or requirements, thereby satisfying the customer needs rather than following a specific set of practice.

Key words: Agile adoption, waterfall model, agile coach, team

INTRODUCTION

The ability to shorten software development times, to bring visibility into the development process and hopefully to better satisfy customers have led to a wide adoption of agile development practices in many companies. Many of the companies have seen the rise of software processes, where various improvements in software development have been pursued by adding more processes. Adoption of agility can be seen as a counterforce to the software process movement and software engineering in general.

Transition to agile processes can be used as an argument to remove nearly all existing processes that are available in the traditional methods. Initially, the development teams will enjoy this freedom. Later they find out that the large-scale use of agile processes requires techniques that are not so different from more traditional approaches. The key challenges seem to be managing a large number of agile teams, dividing work among those teams, achieving the system-wide properties of the software and guaranteeing the simultaneous releases of cross-cutting features.

Transition to the agile development explains the key practices of requirements engineering and their importance. The article shows the lack of these practices that hinders industrial product development.

In Extreme Programming (XP), Beck and Andres (2005) has said that the traditional way of getting requirements does not allow change and it is plan driven. Scrum (Schwaber and Beedle, 2008) is one of the most popular agile method in industry. In scrum the requirements are kept as a list of backlog items. A backlog contains a prioritized list of all product requirements (Schwaber and Beedle, 2008). This means that a typical backlog contains items that vary on the abstraction level, detail, focus and on how much design information they contain. Backlog items are implemented during a sprint that is usually a 15-day or shorter iteration cycle.

Literature study: Agile process is used in cases where speedy action is given importance. An agile approach provides a more flexible means of responding to change. Getting early release in an iterative development approach of the product, to the customer will have an impact on performance resulting in variation in product quality. Larman (2004) classified many issues pertaining to waterfall approaches as follows:

- Waterfall works best for projects with minimal change and low complexity
- In Waterfall, the high-risk and difficult elements are taken up during the end of the project
- Waterfall is not suitable to deal with changing requirements
- Integration is done at the last in waterfall

Corresponding Author: Ganesh, N., Department of Computer Science and Engineering, Anna University of Technology, Coimbatore, India

- Schedules and estimates are not reliable

Harrison and Coplien (1996) found patterns of organization that led to high productivity, the key factor is said to be an iterative approach. Some of the examples of traditional model include the waterfall model, the spiral model, the RAD model and the prototyping model.

Research context: The research can be characterized as constructive research, in which a case study forms the basis for further development and evaluation of the proposed agile deployment model and the methods integrated in it.

The case study taken in this article was conducted at a software firm in Chennai, India. The study is made to a banking domain team. The customer was a large player catering to various banking needs. The project involved enhancing and maintaining the existing systems that was carried out in a language called OO COBOL with DB2 as the backend. The entire work of the project was carried out on a mainframe server, with a planned production release of every 60-90 days. The 10-member team followed a waterfall cycle in the previous releases.

As an initial step of transitioning, a meeting was conducted which in agile called as a retrospective meeting that speaks about the current development process and the following problems were put forth:

- The team was overworking during the last few weeks preceding the release, which is the usual mentality of software engineers
- The team did not stick to the release date due to delays in development and testing
- The customer was not satisfied with the defects that he/she faces during the acceptance testing

The above problems was resolved to an extent after the team was given training on agile, rather than explaining them the exclusive agile practices. As that was the first time, the team was given the freedom to choose the set of required agile practices which would sort the above said problems. After all the interventions, the team decided to have the following activities:

The release would have two 4-week iterations, wherein the team believed the length of iteration could not be reduced any further. They felt that having a demo midway in the release would help them get early feedback.

The team manager decided to have complete draft of their work for 4 weeks with complete supporting documents.

As the mindset of the people in the team could not be changed by a single discussion, they were not forced to use any stringent agile practices which they were not comfortable with. Although it was in the mode of transition, the team was motivated by the need of doing early testing to solve the problems. Apart from this, the team was asked to create weekly builds from the second week of the iteration, wherein iteration is demarcated fortnightly.

The team manager was made responsible for explaining the business and technical challenges, which in agile practice called as time-boxing. Daily standup meetings was conducted for 10-15 min in a day, usually in the morning hours to find the work that they have completed the previous day, the work that is still pending and other issues in completing the work. Apart from this, interactive sessions were conducted week to week, which is again called as retrospectives in the agile practices.

Transitioning the project to agile:

Initial days of the project: The team was assigned specific tasks, usually called as stories in agile term. These stories were discussed during the stand-up meetings, which is a practice that is followed in scrum framework. A story board was kept on the room which was helpful in tracking stories and in disseminating the information to the entire team. The board depicted the status to showcase whether a particular story has been started or not and the present status of the story. Apart from the sticky boards, a white board was kept at the center of the room, wherein unsolved pending issues were listed so that any volunteer could come forward to take and solve them.

During the second week of the work, the team was not able to coop up with the weekly build. Immediately, a retrospective meeting was conducted and some strong decisions were taken in terms of having only relevant documentation and to have a simple design decisions. It was also shared to the customer and was also agreed. The entire work was carried out in the presence of an agile coach wherein the coach explained the importance of test driven development. So, the team agreed to write tests first.

As the team was transitioning their mindset from waterfall to agile, they had an ideology of speaking in terms of design and coding. It was the agile coach who explained the concept of parallel design and coding and it was tracked together on a story board, which was also accepted by the so-called dynamic agile team.

Forthcoming weeks: After the intervention of agile coach, the team was back on track with respect to plan.

A retrospective meeting was conducted and certain characteristics with respect to agile practices were identified such as self-discipline was emphasized, which means the team had become self-organized, in the meaning that they were able to identify their problems and bring out solutions on them, by two team members pairing together. The team started their testing from the fourth week of the inception of the project. The story board played a vital role rather than following the already available built-in project plan.

As the build frequency was made weekly, exclusively during the midweek, the team was not sure about the correct fixing of defects. Therefore, the team realized to have more frequent builds to check the efficacy of their work and the frequency was increased to thrice a week.

The agile coach focused on the concept of collective ownership, wherein the team would bring in more realistic estimates and it would pave way to complete the iteration on time.

The team later agreed that pairing during an analysis phase would have reduced the defects and started to understand the benefits of agile.

The visibility of collective ownership resulted in fulfilling the development work and system integration testing was said to be the prime focus. The build frequency was made to be a daily activity, which in turn increased the testing effort and the team members were able to identify defects earlier in the life cycle.

Sprint week and the lessons learned from iteration: Most of the planned stories were completed, with very few known defects/issues. The demo was given to the customer. The few defects were recorded and assurance was given that it would be addressed in the next iteration.

The four-week tenure of the first iteration fixed certain problems like not integrating early and not delivering frequently. Automation was identified as another area for improvement. They decided that continuous integration and continuous delivery will result in better quality. They decided to strengthen test driven development. Tests would be written upfront and would be updated as design and coding progressed.

The team decided to own two stories by a pair, wherein the analysis would be done in pairs and the remaining tasks may or may not be paired, which means collective responsibility and ownership. During this meeting, it was proposed that the pair would need to come up with the estimated effort for their stories.

The team started to complete iteration two, fulfilling the improvement actions identified during the earlier retrospective. Then they decided to have pairing continued beyond the analysis, wherein after a few hours of coding, the pairs would meet again to verify the code and test. The customer was literally satisfied with the output that he was showcased with very few suggestions from his end.

For the future releases, the team started measuring code complexity metrics that acted as the trigger for re-factoring. And they decided to minimize the iteration time to 2 weeks.

Lessons learned from the team members:

- The team members should be willing to adapt or welcome change.
- The team should have highly skilled people who are good at gathering requirements and executing them at ease.
- The team members should be a master in all trades
- The team members should have a social movement
- The team members should understand the values and principles of agile, rather than its practices
- The team should be self-organized
- The team members should take up collective responsibility, thereby should gain collective ownership
- The team members should be willing to do continuous integration, with continuous delivery and should be willing to adapt/change towards the continuous feedback from the customer end

Lessons learned by the agile coach:

- Slow motivation is required when transitioning from traditional to agile approach
- Handholding or mentoring is required from an agile coach. Proper guidance is mandatory at every initial stage
- Agile coach should act as a counselor and guide the team in a constructive way
- The coach should be responsible for increasing the rigor depending on the project needs
- Commitment of agile coach needs to be very high during the initial weeks of transition
- It is the responsibility of the agile coach to choose the measurements carefully, especially with respect to builds
- Changing the mindset of the team members and the project manager will be a challenge to the agile coach till the completion of the project as it is very difficult to satisfy all the needs of a particular person
- The agile coach should convene a meeting to have a discussion with the project managers who are willing to make a transition with the project manager who is already practicing agile

CONCLUSION

Waterfall development has been widely condemned in the literature and research of software development practices. But, even then transitioning from waterfall to agile cannot be done overnight or in a single step. It is very difficult for anyone to unlearn old traditional practices and move towards agile. It would be wise to remove unnecessary processes and tools when making the agile transition than to start from scratch and add new practices. Coaches need to be patient, positive and persistent while bringing in this change. Agile practices require people with a common mindset. If few people are inflexible and refuse to change their ways, it is best not to have these people in an agile project. In this article, the agile adoption of one project is summarized. The main suggestion to all agile coaches is to respond to the changing needs of the team rather than following a preconceived plan or set of practices in adopting agile. It is after all satisfying the customer needs rather than following a specific set of practice.

REFERENCES

Beck, K. and C. Andres, 2005. Extreme Programming Explained: Embrace Change. 2nd Edn., Addison-Wesley, Boston, ISBN: 0321278658, pp: 189.

Harrison, N.B. and J.O. Coplien, 1996. Patterns of productive software organizations. Bell Labs Techn. J., 1: 138-145. DOI: 10.1002/1538-7305(199622)1:1<138::AID-BLTJ2010>3.0.CO;2-M

Larman, C., 2004. Agile and Iterative Development: A Manager's Guide. 1st Edn., Addison-Wesley Professional, Boston, London, ISBN: 0131111558, pp: 342.

Schwaber, K. and M. Beedle, 2008. Agile Software Development with Scrum. 1st Edn., Pearson Education, Limited, Upper Saddle River, NJ., ISBN: 0132074893, pp: 158.