Our Journey to becoming Agile

Experiences with agile transformation in Samsung Electronics

Suhyun Kim, Hyungjong Lee, Yonghwi Kwon, Myeongsang Yu, Hyeeun Jo
Agile Office, Software R&D Center
Samsung Electronics
Seoul, Republic of Korea
Email: {suhyun47.kim, hj999.lee, yhwi.kwon, ms12.yu, hyeeun1223.jo}@samsung.com

***Abstract*— For traditional consumer electronics companies, it is important that requirements are clearly described and design based on the requirement analysis is well-defined in the early phases of a project, because well-defined design and interfaces guarantee that all necessary tasks for the project are identified and assigned to the appropriate teams to maximize the constructive interactions among multiple teams. There are several benefits to this; however, there are also some weaknesses. Changing or adding features are usually not permitted, and 'organizational silos' that decrease efficient cooperation among teams are inadvertently formed. We thought that our company could overcome these weaknesses using agile development methodologies, so we decided to transform our organization to become more agile from being reliant on traditional development process. In this paper, we describe our efforts to achieve this agile transformation including our road map, process revision, master plan to adopt engineering practices, cross functional team formations, and cultivation of an agile culture.**

Keywords - Agile Office, Agile Transformation, Scrum

I. INTRODUCTION

For traditional manufacturing companies which produce consumer electronics, it is almost impossible to change the hardware such as the chip, printed circuit board and figure after implementation, so waterfall development model is very pervasive. But, software which is considered as a part of the product should follow this model in order to collaborate with other parts of development even though the waterfall model might not be appropriate for current software development. As the market changes faster and portion of software increases, we need to evolve our software development to remove inefficiencies found in traditional way of doing work.

Major characteristics of companies following waterfall model are plan-driven development and organized in component team structure. In plan-driven development, it is important that requirements are clearly described in the earlier phase of a project and design based on the requirement analysis is reviewed rigorously, because well-defined design and interfaces between components based on the fixed requirements guarantee that all necessary tasks for the project are identified and assigned to the appropriate department at the earlier phase of the project, and some modules are able to be developed in parallel by multiple departments. In this approach, it is critical to observe the milestone for each department because they need time to integrate modules into the whole product [1]. However, in the traditional plan-driven development, it is difficult to change features to quickly adapt to market trend even in software, because all features are developed as planned to coordinate completion schedule of other departments, so no changes can be applied without agreement from all of them. In product development based on the component team model, there are many specialized groups such as business, finance, management, development, QA and operations. There are several benefits from component team model, for example specialized and well-trained resources are reserved in each group and increased productivity by optimizing and simplifying jobs [2]. However, in component team model, silo effects exist which means decision making is performed for maximizing the benefits for each individual group rather than for the whole product, for example business focus on signing contracts, finance focus on reducing expense, management focus on creating schedule, development focus on meeting schedule and QA focus on finding many bugs[3]. It's one of big obstacles for a successful product.

In our company, Agile Office was established on January 2015 for agile transformation, and Agile Office has been trying to adopt scrum[4] to relieve rigidity in plan driven development and organize cross-functional teams to reduce the silo effect from component based team model.

II. AGILE TRANSFORMATION OVERVIEW

Our journey to agile transformation started with identifying impediments from 153 developers who had experiences with performing agile practice in our company. The result is shown in Fig. 1. We found that we need to change our culture so that employees fully understand agile values.

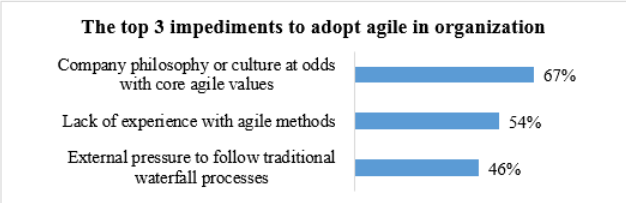

Fig. 1. The survey result of Samsung Agile Conference 2015

1530-1362/16 $31.00 © 2016 IEEE
DOI 10.1109/APSEC.2016.60

We established a step-by-step plan to expand and apply the agile culture gradually for this purpose, and we classified three phases: Process transformation, Engineering transformation, Organizational transformation as the overall journey and established the agile transformation roadmap like Fig. 2 as a deployment strategy with reference to the Scrum Master Focus model[5]. Currently our journey is in phase 2.

A. Phase 1. Process Transformation

In the first phase, we establish agile software development process like Fig. 3, and guide the development team to follow this process instead of waterfall process. We define the roles and responsibilities of the members such as team, scrum master and product owner, and operate process-related practices such as sprint planning, sprint review, retrospective, daily meeting. To overcome lack of experience with agile methods, Agile Office organized training sessions for agile coaches who help the project with the role of scrum master to train project members on practical practices based on actual projects. After applying this process we achieved 14.6% improvement on process satisfaction, visibility, SW quality, communication with team and productivity based on surveys which were conducted by developers after finishing their last sprint.

B. Phase 2. Engineering Transformation

In the second phase, the practices related to process are matured, but development team should improve their capability to achieve "definition of done" of each sprint. One of the great features of agile development is to release working software, called a potentially shippable product, in each iteration by implementing higher priority requirements in contrast to the traditional waterfall development which delivers working software after implementing all features. However, if developer teams don't have enough maturity to accomplish the definition of done, they will accumulate "technical debts", and their software will not actually be potentially shippable because they have to do more work in order to release software to settle the accumulated technical debts such as documentation, refactoring and so on. To reduce the "undone" work, it is necessary to perform engineering practices such as Pair-Programming, Test-Driven Development and Continuous Deployment well to operate software-centric development based on iterative and progressive improvement emphasized in the agile manifesto. However, employees' resistance against adopting TDD or pair-programming is quite substantial, because the practices are unfamiliar and are often considered secondary. So, our strategy on introducing engineering practices is to pursue them after members get enough experience to understand agile values from process-related practices.

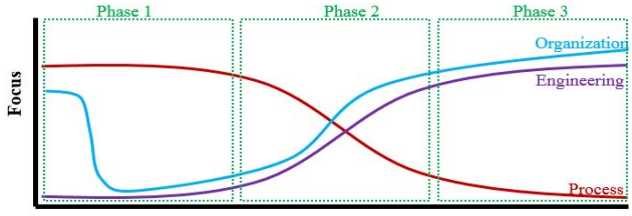

Fig. 2. Agile Transformation Roadmap

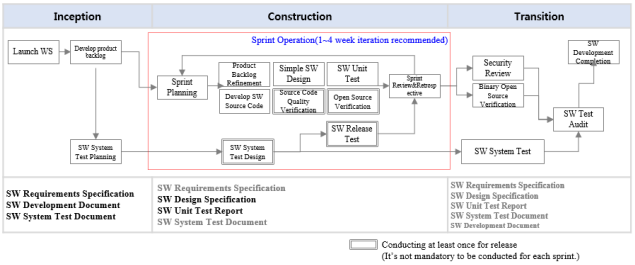

Fig. 3. Agile Process

C. Phase 3 Organizational Transformation

The third phase is the final step in the pursuit of agile transformation at enterprise-level. There are several limitations to operate the process and adopt engineering practices with only each scrum team's effort, because it's very difficult for an employee to overcome the constraints set by the organizational systems; therefore modifications to the HR operation, evaluation policy, organizational reconstruction, and so on need to be accompanied. However, this is not easy, because it needs commitment from all groups and members. In our company, we established the PO (Product Owner) system. It starts from defining small products for small teams instead of a big product for many small teams. And a team is basically whole team which consists of UX, Developer, QA, business, and PO who is authorized person to make a decision of each small product and to evaluate personal performance [6]. So in each iteration, the team gets fewer interrupt from outside the team and team morale increases.

D. Cultural Transformation

The ultimate goal of the agile transformation is changing the culture of the organization, as well as changing institutional process and introducing engineering practices to developers. These begin from changes of individual perception. So, we have performed several activities such as training, community, conference and newsletter.

III. Case Study

In this section, we introduce actual cases of agile adoption in a small team of less than 10 employees and a large team over 100 employees.

A. A Scrum for Small Team

This team was a small team composed of five employees. They had developed web applications in Tizen platform and released them to public for Tizen web app developers. In the beginning, we had a process-tailoring meeting to decide the process model, rough schedule and target goal. After that, team chose a scrum master. He had no experience with the role, so he took three days training course provided by Agile Office and an agile coach additionally supported the team for three months. The agile coach introduced agile fundamentals, role description, and overall process to the team and led the launch workshop for team members to get consensus on their product. These are activities that we perform before starting the first sprint. In the first sprint which we call sprint zero, we

performed release planning and decided on the duration of each sprint as two weeks. Then we made the product backlog and prioritized them. We described the first sprint backlogs in more detail, and wrote in less detail for items expected in later sprints. And we clearly defined the product using the Elevator Pitch practice, and after that we listed risk points that were expected during project progress though the Pre-Mortem practice. And then we started sprint#1 and proceeded sprint planning, estimation, daily meeting, sprint review and retrospective like in Fig. 4.

In the sprint planning, development team made up list of sprint backlog items from higher priority product backlog items and decided the scope of sprint by estimating story point of each sprint backlog item. In this project we defined one story point as the amount of work that one developer does for one day and we considered working hour for each day as 6 hours on the basis that developers spend more than two hours on email, meeting and communication with other groups. During sprint progress, it was very helpful to decide on the scope of sprint because we could grasp development velocity based on story point estimation practice which became more accurate as it was performed through sprints. We had a daily meeting in a stand-up meeting room near the work place, and it usually took no more than 15 minutes. Through the meeting team members revealed issues better because their communication was horizontal instead of hierarchical [8]. We usually had a sprint review as an internal demo with PO or stakeholders to get their feedbacks at the end of each sprint. Review was an important activity for the team because they listened to opinions from the user's point of view about new feature or UX through the demo. In the retrospective, they identified several difficult issues and formed consensus and discussed what to improve from previous sprints, and they reflected one or two things that we wanted to apply to the next sprint as improvements.

We found that the team's communication and collaboration are improved using a simple framework of scrum and practices when the team accepts agile. Overall we could see the change effect in the mindset of the team, and a little more aggressive, spontaneous atmosphere formed within the team.

B. A Scrum for Large Team (Cross-Functional Team)

The second team that we introduce is a project for software development kit. The team composed of almost 90 people including product owner, program managers, developers, UX designer, QA and agile coaches. In this team, there were over 50 developers, 15 UX designers and over 10 QA. Most of the team members worked in the same location at HQ, but some developers were located separately about 15km away. These members attended major events like sprint planning, review(demo) and retrospective. However, daily meeting was done through a conference call. This team consisted of five scrums, and Agile Office supported all scrums with agile coaches. First of all, the agile coaches helped in defining PO's role and organizing CFT(Cross Functional Team) including developers, UX designer and QA. The CFT was not composed as a physical whole team because some of developers were not in same place, and UX designers belonged to another department, so it was a virtual team. Agile coach guided the team to cooperate with each other closely as a whole team. Team members communicated with others face to face not only through daily meeting but also every hour in the office. In addition, scrum of scrums [9] were operated for sharing various issues of the scrums. The scrum of scrums meeting took place in the same format like a daily meeting.

After building the CFT, we had a vision sharing workshop as sprint zero's first activity where we defined target user and product and also made ground rules of the scrum team which helped team empathy. After that, we performed user story workshop and defined the internal and external release schedule for each scrum and the whole product. In this process, the scrums decided to release items in accordance with the priority. As sprint #1 began, agile coaches focused on practices related on the development process, and helped team members communicate with others well in the scrum team. At the same time, agile coaches also had a role in the coordination and troubleshooting issues between each scrum and within each scrum. Specifically, scrum team performed practices associated with the process; for example, planning, review, retrospective, daily meeting, and scrum of scrums meeting, clean-up, and backlog refinement and so on. In this team, engineering practices were not extensively performed due to lack of experience on those practices.

We have found the agile methodology based on scrum in large-scale projects effective on their project. According to the interviews, most team members think that current development methodology is much more efficient way than the old development methodology like in Fig. 5. In addition, team members feel that they are developing a more valuable product than in the past.

Fig. 4. Team activity photos(Vision Sharing Work Shop, User Story Workshop, Planning, Review, Retrospective)

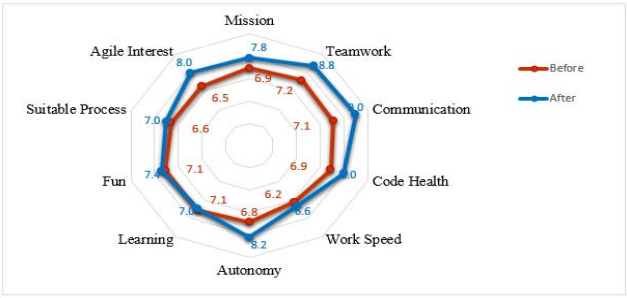

Fig. 5. Agile Health Check (Before and after results)

IV. CHALLENGES

A. More agile coaches are needed.

One of major characteristics of agile is that the value is more important than the practice. So, it means that practices can be performed in various ways to meet each team's own value. Therefore, it's very difficult to describe a standard guide to perform each practice, so it's helpful for each team to be guided by an agile coach who can give advice to the team to get value via each agile practice by tailoring it. But the maximum number of teams which one agile coach can support at the same time is limited to three; agile coaches in Agile Office cannot cover all the teams in our company. We should train developers as agile leaders.

B. Experience on engineering practice is insufficient.

We conducted a survey for agile teams about how many agile practices are performed, and the result shows in Fig. 6 that process oriented practices such as sprint, daily meeting, product backlog, sprint retrospective and scrum board are well executed, but engineering practices such as Test-Driven Development, Continuous Integration and Pair-Programming are not. The reasons for this are insufficient number of developers who have understanding or experiences on engineering practice and a lack of consensus on the benefits. So, Agile Office should try to experience engineering practices and prepare guides on suitable practices to be adopted, benefits, costs and obstacles before introducing to developers.

C. Personal performance evaluation should be improved.

A CFT(Cross Functional Team) which includes all roles such as product owner, UX designers, developers and QA to develop a product brings many advantages. However, our company could not establish a physically co-located CFT but virtual team. In the virtual team, each person partially participates in the scrum which is called the CFT while they still belong to their original department. So, they are not evaluated by the scrum team members who are working for the product, but by their department leader, so the motivation to concentrate on the scrum team is decreased. So Agile Office should find the most reasonable evaluation policy given the constraints.

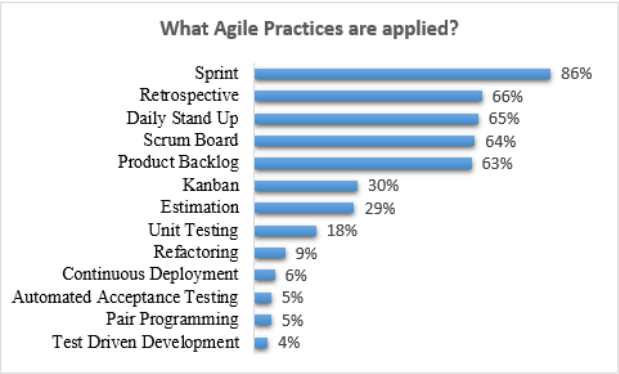

Fig. 6. What Agile Practice is applied?

D. Keeping small team is important.

In our company, it is not strange to have more than hundreds of employees participate in a product such as smartphone or smart TV whereas two pizza team is strongly recommended for establishing a product team[10]. So, it's almost impossible to deliver working software by one scrum team because it is not a product team and the software they develop without cooperation with other scrum teams cannot be released to customer.

V. CONCLUSION

In this paper, we describe our efforts to achieve agile transformation. Our company as a manufacturing company which makes consumer electronic product has been developing software based on the waterfall model. But now, we are trying to adopt agile software development methodologies to survive in the upcoming software centric world. For successful agile transformation, first we made a road map which contains three phases. Second, we revised our process. Third, we established a master plan to adopt engineering practices. Fourth, we have tried to build cross functional teams. Last, we have cultivated our agile culture by operating many agile training courses, holding conferences, and publishing agile newsletters.

In the future, we will try to overcome challenges which are mentioned in chapter IV. We'll try to train more agile coaches who are not only process enablers but also technical leaders for guiding engineering practices such as Test-Driven Development and Pair-Programming. And we will establish some pilot projects which are organized as a whole team - members such as UX, developers, QA - and also empowered to validate transforming organizational structure and performance evaluation method to attain even more positive benefits than the present.

REFERENCES

[1] Bell, Thomas E., and Thomas A. Thayer. "Software requirements: Are they really a problem?." Proceedings of the 2nd international conference on Software engineering. IEEE Computer Society Press, 1976.

[2] Taylor, Frederick Winslow. Scientific management. Routledge, 2004.

[3] Beck, Kent, et al. "The agile manifesto." ,2001

[4] Schwaber, K. and Beedle, Agile Software Development with Scrum. 2001: Upper Saddle River: Prentice Hall

[5] Larman, Craig, and Bas Vodde. Large-Scale Scrum: More with LeSS. Addison-Wesley Professional, 2016.

[6] Samsung Implements 'Product Ownership System', Korea Bizwire, 2016.

[7] Klein, Laura. UX for lean startups: Faster, smarter user experience research and design. " O'Reilly Media, Inc.", 2013

[8] Deemer, Pete, et al. "The Scrum Primer: A Lightweight Guide to the Theory and Practice of Scrum." Retrieved August 3 (2012)

[9] Sutherland, Jeff. "Future of scrum: Parallel pipelining of sprints in complex projects." Proceedings of the Agile Development Conference. IEEE Computer Society, 2005.

[10] Atlas, Alan. "Accidental adoption: The story of scrum at amazon. com." Agile Conference, 2009. AGILE'09. IEEE, 2009