Waterfall to Agile: Flipping the Switch

Bhushan Gupta
Nike Inc., Beaverton OR
bhushan.gupta@nike.com

Abstract

Agile software development methodology is transitioning from a fad to a practice. More and more software development practitioners are contemplating the switch from Waterfall to Agile; if not entirely, at least partially. Terms like scrum, sprint, stand up, story points, and velocity are becoming the *Lingua Franca*. While enough light has been shed on the mechanics of creating a release and iteration plan, managing story development, and sprint retrospectives, the team dynamics and behavior for success has not been given the mindshare it deserves.

Although the rules of the game, when moving from waterfall to agile development change substantially, the players and their goals do not. The player "product manager" now becomes "product owner" and needs to very actively pursue customer interest as a customer surrogate. Test engineers who took pride in breaking the code are now urged to work side by side with the developers and ensure that the defects are resolved immediately for a successful completion of a sprint. The project managers, who are supposed to drive the schedule, take on a role of facilitators making every effort to see that the sprint is complete and the velocity is adequate. Requirements that are frozen in the waterfall methodology rarely gel and the designs go through multiple refactoring cycles to make sure that the development progresses. The new product quality champions who were always looking for "the knee in the defect trends" now barely witness any fluctuations in the defect find trends from sprint to sprint. Refereeing the game, there is a new character in the theatre, "scrum master", who manages the delivery of the stories.

Can this change be achieved by simply creating a backlog, a release plan, and a sequence of iterations? No, it takes changes in people's perspectives, the way they think, and the way they achieve their goals. Would a Type A "product owner" who is used to taking a product to the market with all the bells and whistles be comfortable to do so with less than the planned functionality? Would a star developer forget about gold plating and would a test engineer who always took pride in finding errors in someone else's work be able to stand by developers and work with them? Most of all, will the orchestra be able to perform without the precisely written music; the "requirements document"? This paper analyzes the transition of a waterfall team to an agile environment and how changing the perspectives of the players can make the transition a success when the switch from Agile to Waterfall is flipped.

Biography

Bhushan Gupta has a M.S. in Computer Science from the New Mexico Institute of Mining and Technology, Socorro, NM. Bhushan has 27 years of experience in software engineering, 17 of which have been in the software industry. Currently a senior member of the GSS QA team at Nike, Bhushan is deeply involved with the agile development. Prior to joining Nike, Bhushan worked at Hewlett-Packard for 13 years in various capacities and led his groups in product development lifecycles, development methodology and execution processes, and software metrics for quality and software productivity. Bhushan has published and presented numerous articles in various conferences and has participated in panel discussions.

1 Introduction

Prior to the 1990's, the software development practice involved converting well defined customer requirements into a product using procedural languages, following a sequence of milestones and checkpoints, and "freezing" the current state at a milestone to avoid last minute changes. This methodology, commonly known as the "waterfall lifecycle", included best practices such as "test first" where the test cases were developed from the requirements in parallel with the code development. The waterfall lifecycle focused on a high level of customer involvement during requirements definition and early prototyping but this involvement faded away as the product development proceeded. The customer was brought back into the game after the product was fully developed, this time to accept the product.

In the 1990's two major changes were brought into the world of software development; the procedural programming was replaced by Object Oriented programming and "Extreme Programming" was introduced as an alternative to the waterfall lifecycle (Wikipedia 2012). The "Extreme Programming," introduced by Kent Beck in 1996, provided a new vector to software lifecycle with focus on incremental development and frequent customer review to better suit the ever-changing customer requirements. This new approach also provided the ability to meet the 'First to Market" challenges presented by the Internet boom. The approach adopted some of the best practices, such as the test first of waterfall lifecycle. "Xtreme" (commonly spelled) programming morphed over time as new practices evolved and took on a common name "agile" development. Over the past decade a significant amount of change has taken place and the agile development is taking a hold. Shifting paradigm requires a change in mind set of the people involved. Is it happening? Are key people changing their behaviors as we move forward with the agile approach?

This article establishes the personality traits that are essential to build a successful agile team. Once internalized, the traits can be acquired by a team as it blazes the agile development trail. The article analyzes the role of each key team member, identifies the human characteristics that are relevant, and establishes a transition that facilitates an effective agile development environment.

2 Brief Background on Personality Types and Traits

Before we drill deeper into agile teams it is prudent to draw a distinction between personality type and traits. A personality type refers to quality of a person while a personality trait embodies a smaller grouping of behavioral tendencies. According to Type A and Type B personality theory, impatient, achievement-oriented people are classified as Type A, whereas easy-going and relaxed individuals are designated Type B (Wikipedia 2012). The other classification of personality based upon the behavior is as follows:

- Neuroticism – tendency to experience unpleasant emotions relatively easily,
- Extraversion – tendency to seek simulation and enjoy company of others,
- Conscientiousness – tendency to show self-discipline, consultative, competence, order, dutifulness, and thorough,
- Agreeableness – tendency to be compassionate towards others,
- Openness to Experience – tendency to enjoy new intellectual experiences and ideas, and
- Cognitive – abstract level thinking, analytical etc. (relatively new).

3 Key Players in Software Development

The software development involves multiple actors; a *customer*, *product owner*, *project manager*, *program manager*, *development team*, *quality assurance (QA) team*, *business systems analyst*, and a *technical writer*. An additional character in the agile development is the *scrum master* whose job is to orchestrate the product development rhythm. In the agile development method, the product manager is

normally called “product owner”. The scrum master role in the traditional waterfall environment was normally attributed to the project manager who made sure that the development was on track. There is a transformation these key actors need to make to support the change.

3.1 The Customer

Regardless of the development methodology, the customer plays an important role in the product development. Hayes (Hayes) has described four categories of the customers: analytical, driver, amiable, and expressive. In the case of agile development the objective is to create a high customer value and thus work very closely with the customer. Often times, a real customer is not available on site to work with the development and a surrogate customer fulfills this role.

An “analytical” customer can enhance the stories by bringing order, precision, discipline, and logic to the development, resulting in a better quality product. However, to achieve success iteration after iteration and maintain the rhythm, a “driver” type customer is very desirable. A “driver” personality is action oriented, decisive, demanding, forceful, competitive in nature, and will contribute to finish each iteration on time. The “driver” personality should be complemented with the expressive personality to bring in motivation, influence, confidence, and optimism. An amiable customer will be appreciative of the results achieved in each sprint. It will be highly desirable to build a small team of surrogate customers that is inclusive of all four customer personality types.

Although it is highly beneficial to have a customer with the product delivery team, it is hard to convince one to dedicate time with the team unless the customer is internal. In most cases, the product owner becomes the customer surrogate. As discussed below, the product owners are often Type A personalities. To fulfill the customer role they also need to be drivers, forceful, decisive, commanding, action oriented, and possess superior interpersonal skills. It is a challenge to find someone with all these traits. What is the solution? This can be achieved by a tag-team where the scrum master can contribute to the customer role provided he/she is comfortable in surrendering control to the product owner who ultimately answers to the customer.

3.2 Product Owner

Often a market savvy individual, the product owner has a keen sense of customer requirements and a goal to make future products more successful than the current in the market place. Driving the product qualities to delight customers, the product manager’s motto is to give the customer all the bells and whistles so that the product can take the customer by storm and easily beat the competition. A product owner, normally a Type A personality full with ambitions, rigidly organized, takes on more than she/he can handle. They are often a highly achieving individual, are proactive and obsessed with time management. These personality traits translate into a strong desire to yield a market winner product delivered on time.

In a typical waterfall environment, the product manager facilitates gathering requirements, making sure that the requirements are understood and agreed upon by all stakeholders, and are NOT subject to change. The requirements are said to be frozen at this stage. Once the requirements are frozen, the product manager often does not necessarily interact with the customer or actively looks for new requirements that may have risen due to time lapse.

Agile Product Owner blog (Agile Product Owner, 2012) explicitly lists the responsibilities of an agile product owner. When introduced to an agile development, a product manager, now the product owner often desires to have a complete picture of the product. If a product is being ported to a new platform, one of the motivations of this actor is to assure that the entire functionality is carried over to the new platform. There is a high probability that the product owner would insist on mapping the entire product functionality to epics and stories and gather details on each story to feel assured that no functionality is missing in the new product. Once the stories have evolved, the temptation is to develop the associated tasks and the acceptance criteria for each story. This leads to spending a large amount of time up front in gathering and

perfecting stories. It is equivalent to flipping back to the requirements gathering phase of the waterfall methodology. Now that all the stories have been defined, the product owner is ready to move to story prioritization.

The product owners have a difficult transition to make. For a product owner, it is important to understand that success in the agile development is achieved by perfecting one story at a time. They need to be cognizant of the fact that the product will grow over the entire development and what has been developed so far must provide value to the customer. Instead of nailing down all the epics, he/she needs focus on an epic that delivers the most value to the customer and builds customer interest. The product then evolves based upon the customer interest and the added value it provides. When porting a product to a new platform, it is important not to dwell into each epic and drill down every story just because the entire functionality is known. It is an opportunity to take an epic, analyze it from the customer perspective and enhance it to provide a superior customer experience. The product owner must focus on one story at a time.

3.3 Scrum Master

Scrum Master, the backbone of agile development, has a distinct personality. Peter Deyoe in his blog (Peter Deyoe, 2009) describes the following characteristics of a Scrum Master:

- Be humble enough to serve the team
- Have a strong character and be confident enough to stay in the background, promoting the team
- Display a high degree of integrity and maintain a trusting relationship
- Be politically savvy with a strong relationship with the Product Owner and,
- Be able to understand both business and technical people

“The best Scrum Masters are real team players, who receive as much satisfaction from facilitating others’ success as their own. They must also be comfortable surrendering control to the Product Owner and team. For those two reasons, traditional project managers don’t usually make great Scrum Masters (Scrum Methodology, 2010).”

The above statement could be controversial as the *project managers* do possess integrity and are trusting. They also work very closely with the product owner and have a good technical understanding of the product. The project managers are technical and thus can understand technical intricacies, dependencies, as well as help and support the team. Often the project managers have a good understanding of team capabilities and can help develop good sprint contents. However, a project manager, by the nature of his/her role, may not be able to stay in the background and relinquish control. Agile teams are self motivated and any attempt to control them can result into chaos.

Is it appropriate for a *program manager* to lead the scrum master role? A program manager oversees several related project aspects and is a strong driver of overall product delivery. Experience shows that the program manager often becomes a de facto Scrum Master. The nature of their job is not entirely different than that of a typical project manager and unless they are well trained they will run into the same problem as the project managers.

The program managers are strongly driven by the outcome and therefore maintain a sharp focus on schedule. By training, they are very conscientious of milestones and checkpoints. At the same time they are not accustomed to being in the background and are not current on the technical aspects. The role of the scrum master has been well defined and the scrum master skills can be acquired via formal training widely available. A well trained scrum master would make a team very successful.

3.4 Development Team

The development team in the current context refers to software architects, designers, and developers. These are the characters that are most impacted by change. Before we look into the change, let us first establish the traits of these professionals. By nature, software architects, designers, and developers are creative people. They are innovative, flexible, and are open to new experience and new ideas. They are always interested in improvements and like to discuss their ideas with their colleagues. Often under schedule pressure they are afraid of failure.

The software engineers have been measured as follows on these traits (Sodiya et al., 2007):

- Neuroticism: Low
- Extraversion: Medium
- Conscientiousness: Medium
- Openness To Experience: High
- Cognitive Capability: High
- Agreeableness : High

In a typical waterfall environment, the project is not only bounded, but the architectural, design, and coding activities are very structured. The scope is defined to the degree that the requirements are frozen and the probability of any change is very low. On the other hand, the stories in the agile environment are brief and the details are added in the form of tasks and acceptance criteria. Lack of details in the story is substantiated with on-going interactions between the stakeholders. There is more room for misinterpretations, misunderstandings, and missing requirements. Since the development is incremental in nature and the architecture and design elements evolve leading to refactoring, there is a higher degree of freedom to be creative and therefore a higher potential for chaos. In the waterfall model the discipline is achieved by milestones and checkpoints; in the agile methodology where the customer value is delivered every 2 or 3 weeks, self-discipline is very important. The nature of short delivery span enforces a high level of discipline, ongoing interaction with the stakeholders and on time delivery. Experience shows that successful agile teams always deliver the planned stories at the end of each sprint.

So, for a successful change from a waterfall to agile, a development team needs to posses strong cognition, extraversion, conscientiousness, and openness to experience traits. They need to learn how to deal with not-so-well defined requirements, continuously evolve the architecture and design, and be disciplined iteration after iteration after iteration. Vagueness in the deliverables though, provides opportunities to be creative but the short span of time in which the planned work needs to be delivered, requires making reasonably sound decisions. The nature of work changes form developing a product from the well defined requirements to rather loosely defined stories that need to be functional in a short span of time and deliver customer value. There are special needs that an architect faces to meet for a transition to be successful. Since the refactoring is a necessary evil in the agile development, an architect should have cognitive capability to be able to evolve the system architecture. It is necessary for the entire development team to possess openness to experience, extraversion, and agreeableness.

3.5 Quality Assurance (QA) Team

Studies have also measured the personality traits for software quality assurance professionals and the results are (Maverick Tester, 2010):

- Neuroticism: Low
- Extraversion: Medium
- Conscientiousness: Medium
- Openness To Experience: High
- Cognitive Capability: High
- Agreeableness : High

These traits are identical to the traits of the development team.

The Quality assurance team finds a high degree of change while moving from waterfall to agile. As discussed above, the requirements are brief and system behavior may not be well established except in the situations where the product is already in the market and going through enhancements. Often times, the story does not provide enough details and a significant amount of information is embedded in the acceptance criteria. Quality assurance team should not only review the stories and the acceptance criteria, it should also be on the lookout for any additional information that is relevant for testing. This requires extraversion, openness to experience, and agreeableness. The QA team must work very closely with the product owner and the business systems analyst to understand the functional requirements and derive the test cases with them on a continual basis; sprint after sprint. The test cases must be reviewed for each sprint. The environment requires a very close collaboration and not an “us versus them” mentality.

In a large project at Nike where the development is done by an external vendor, the QA team participates in information gathering sessions, communication exchanges, and story acceptance demos. The stories, after they are qualified by the vendor, are presented to the product owner for their approval to release to Nike QA team for final acceptance. The QA group has the liberty of contacting the vendor development team as often as necessary.

The QA team at Nike also works hand in hand with the development team. At first there is always resistance to direct communication from QA to developers, but soon both teams realize the criticality of direct and frequent communication between the two teams. At Nike, the development team continually encouraged the QA engineers to be actively involved with the development. As a result, the defect find, fix, and verification were expeditious in spite the fact that the QA team was not part of the development team. The defect resolution cycle was short and efficient. In another successful agile practice, a few members of the QA team were embedded into the development team. Whether the QA team is a part of the development team or not, its goal still remains the same – be a strong customer ally and a committed member of the project team.

The above discussion brings about the following points which are valuable for the transition from waterfall to agile:

1. The QA team does not have to be integrated with the development team and can still be effective and efficient.
2. A few members of the QA team can be embedded with the development team, temporarily reporting to the development (project) manager.
3. The team should be included in information exchange sessions and one-on-one discussions.

In agile development, the code is not “thrown over the wall” for testing. Successful teams will have a strong sense of inclusion regardless of the development methodology. Agile development environment forces this inclusion.

3.6 Build and Deployment Team

Often, build and deployment is a function of the development team. However, the responsibility is confined to a single or, at most, a few individuals. In the waterfall environment, build and deployment takes a big-bang approach where the code is delivered to the stakeholders towards the end. In the Agile environment the build and deployment happens in every sprint. There is a need to update the build and the opportunity to incrementally improve this process for accuracy and efficiency and finally, to perfection as the product gets closer to release.

The actors responsible for this activity require discipline to keep the process up to date and documented. In addition, the individuals should be proactively looking for the process improvement where the tools utilized to build and deploy are simple “like a single click script”. One should be evaluating the process at

each build and make changes to make it efficient. The two important personality traits for these individuals are extraversion and cognitive.

4 Making the Transition

The key pre-requisites to a making any transition are awareness, understanding, and commitment from the stakeholders (Bridges, 1991). Once these aspects of change are well understood and evaluated, then only a go - no go decision should be made. This understanding should include an evaluation of the project needs, team evaluation – personality traits, and training and coaching availability. Most successful agile teams start at a very small scale and go through an adaptation process. A project that is heavily dependent on the external partners may not be a good candidate for agile development. Ultimately, a project's success is measured on cost, schedule, and product quality. If flipping the switch does not achieve these success criteria or hampers any it is not the time for transition. Listed below are the steps that will facilitate the switch and ease the transition to agile:

4.1 Evaluate Potential and Need for Switch

- Assess if switching to agile reduces cost, shortens schedule, or improves quality of your product delivery
- Assess if the switch should be achieved via adaption over a multiple short projects or the team is ready to switch for a current complex project.

4.2 Evaluate Team Characteristics

- Let the team review personality traits of each role and assess the personal change that will be needed to make the switch
- Once change is identified, evaluate the commitment
- If the project involves an external (3rd party) vendor, assess if the vendor is capable to meet the requirements of change. To change an external vendor is beyond the team control.

4.3 Mentoring and Coaching

- Assess if the experienced mentors will be available to assist the team players as the team goes through the transition
- Prepare senior management to internalize the disruption caused by the change and the potential benefits of the change.

4.4 Measuring the Success

- Develop measures, not just the project progress, but the change in people behavior, project rhythm to understand the progress

4.5 Readiness

- Evaluate readiness before switching

5 Conclusion

Flipping a switch from Waterfall to Agile goes beyond the operational changes. It also includes careful evaluation of traits team members currently possess and the new traits they need to acquire. Even before considering a flip, the team needs to decide whether or not the switch will facilitate cost reduction, shorter schedule, and yet deliver the intended quality. Unlike waterfall, an agile development requires an interactive environment for the duration of the project. These interactions are very prominent between the *customer* and the *product owner*, *product owner* and *scrum master* as well as between the development and the QA teams. The personality traits of each player are critical for a successful transition to Agile.

References

Wikipedia, 2012, http://en.wikipedia.org/wiki/Extreme_programming

Wikipedia, 2012, http://en.wikipedia.org/wiki/Personality_type

Adam Hayes, “If You Only Understood Your Customer’s Personality Styles”, http://www.ahfx.net/weblog/37

Agile Product, Owner, LLC, http://agileproductowner.com/

Peter DeYoe, 2009, The Personality of a Great Scrum master, http://it-insight-blog.com/2009/10/the-personality-of-a-great-scrum-master/

Scrum Methodology, 2010, http://scrummethodology.com/the-scrummaster-role/

Sodiya. A.S., Onashoga, S.A., Longe, H. O. D., and Awodele O., An Improved Assessment of Personlaity Traits in Software Engineering, Interdisciplinary Journal of Information, Knowledge, and Management, Vol.2, 2007.

http://mavericktester.com/archive/personality-traits-in-software-testing/

Bridges, William, 1991, Managing Transitions, 3rd Edition, De capo Press, 1991, ISBN13 978-0-7382-1380-4Author-date style is usually used for references in the sciences.