Conference Proceeding Issue Published in International Journal of Trend in Research and Development (IJTRD), ISSN: 2394-9333, www.ijtrd.com

The Significance of Strategies and Planning and Designing a Framework Agnostic Agile Transformation using Agile Principles

[1]Padma Priya and [2]Mahesh Jade,
[1]Founder, Agile Bodhi, India
[2]Agile Thought Leader, India

Abstract: The entity named 'Agile' took shape in 2001 providing an umbrella to a lot of frameworks and practises or systems which were based on an iterative approach of software development. When Agile was founded in 2001 by 17 thought leaders they framed 4 values and 12 principles. The widely used framework such as Scrum was founded in the year 1995, and XP in 1997. When Agile is implemented in organisations the base values and the principles of Agile are usually given less importance and more for framework practices and events like Scrum, Kanban and XP.

Here is a research which attempts to showcase an approach that talks about platform agnostic agile transformation keeping the principles and values of Agile at the core.This paper also suggests that Lean is the mother of agile. Finally, this paper talks about the kind of questions team leaders and business partners/ product owners should seek to ask to keep a tap on valuable outcomes over output/speed/ ensuring agility is being practiced to its core purpose.

Keywords—*Agile, Scrum, Kanban, Framework agnostic, Lean, XP, transformation*

I. INTRODUCTION

Agile is a new way of software development approach with incremental and iterative design , development and delivery. Agile tries to eliminate the flaws of software project execution using earlier approaches like Waterfall, V Model, Spiral model etc., Agile consists of framework and processes which adheres iterative mode of software development. Agile provides the values to be followed and framework provides how to implement the values and principles into practice. Unfortunately, when Agile is implemented in organisations and teams the cultural events are implemented without realising the fundamental values. According to Tim Guay "A dogmatic approach to Agile, such as prescriptively adhering to the Scrum Guide, is not Agile and is a serious antipattern"[1].

II. WIDELY USED FRAMEWORKS

Framework which adheres to Agile Principles and Values [2] are agreed under Agile methodologies. Each framework gives a great way to look at particular levels and practices of agile maturity. Here is a quick look at ways from each framework:

A. Kanban

Kanban is a lean based pull system, visualization, and other tools to catalyse continuous flow and transparency. . Lean being evolved from Japan and Japanese industries which have primarily a growth mindset with minimum requirements. Lean mindset is used to implement agile software development creating a need for real-time communication of capacity and full transparency of work. Kanban is one such practice under lean. Work items in Kanban are represented visually on a Kanban board. That allows team members to see the state of every piece of work at any given time.

Potential things from Kanban to bring improvement of areas could be:

- Visualized workflow
- Limit on Work in Progress

B. Scrum

Scrum on the other hand is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value. Scrum is a framework that can really make a team understand how to work together. Scrum for team can bring the spirit of learning through experiences, self-organize while working on a problem, and reflect on their wins and losses to continuously improve. Scrum being highly adaptive and less prescriptive can very well be used to bring practices from other agile methodologies and frameworks very brilliantly. Scrum is widely used framework under Agile , since Scrum incorporated majority of Agile principles into its roles, artifacts and ceremonies. [3]

C. Xtreme Programming

XP is the first popular methodology to view software development as an exercise in coding rather than an exercise in management engineering practices test-driven development, the focus on automated testing, pair programming, simple design, refactoring, and so on. The biggest benefit of using Scrum and XP together is, Scrum is an agile framework and XP is an agile engineering practice.

Both because their framework is similar and because Scrum teams often adopt XP practices, and vice versa. XP and Scrum complement each other so well because, though they both embrace agile principles, they come to agility from different perspectives. For instance, the Product Owner decides what will be done in a project in Scrum. In XP customers interact with the development team directly. While Scrum delivers the increments in sprints ending in two-to-four weeks, XP delivers the work in

iterations and these iterations are one- to-two weeks long. After the Sprint Planning, once the Sprint Backlog is determined, Scrum does not allow any changes to be done on the Sprint Backlog. But XP allows teams to substitute a feature in exchange for a feature of equivalent size that the team did not start yet. Whether they realize it or not, the vast majority of teams that call themselves Scrum teams are operating as a Scrum- XP hybrid, or a Scrum-Kanban-XP hybrid, where they employ at least a subset of the technical practices that originated with XP. Note that user stories are part of XP. Estimation. Even though the vast majority of Scrum teams do perform estimation at some level, the practice of estimation is not part of Scrum. Velocity. Velocity is not part of Scrum either; the notion of project velocity is part of XP. Technical practices. As noted above, there is no notion of technical practices (CI, CD, pair programming, TDD, etc.) in Scrum. The major of the technical practices that teams typically follow originated with XP. As the agile maturity keeps up the pace with practices of scrum and ceremonies are well understood, best practices from XP can very much be the next set of improvements.

Potential things from XP to bring improvement of areas could be:

- Small Releases.
- Metaphor.
- Simple Design.
- Testing.
- Refactoring.
- Pair Programming.
- Collective Ownership.

D. Feature Driven Development (FDD)

Feature-driven development is an iterative and incremental software development process. It is lightweight as compared to XP.It looks at blending a number of industry-recognized best practices. As these practices are driven from a client-valued functionality perspective, it can very much be the next step to bring the agile maturity of the team by bringing the force of happy developers into customer obsessed mind set. Plan, design and build by feature. The mature agile teams can start looking at iteration as dedicated timeboxes dedicated for building features than completing stories.To achieve the ultimate goal of producing working software at the end of each iteration, best practices from FDD can very much be picked and practiced.

FDD showcases best ways to look at unit testing and code inspection. Practices of FDD at putting the code into main build are a thing the mature agile team should get very good at to be able to manage the flow of completing the work and making it available in production.FDD incorporates the best of different agile methodologies like Extreme Programming and Scrum.Class Ownership is important in FDD, Feature Teams take shape in FDD. This approach in agile transformation can imbibe the spirit of swarming in small groups and ownership very well.

While there are excellent frameworks as cited above , agile is not heavily dependent on frameworks. With its solid fundamental 12 principles any Agile professional is empowered to derive their own Framework to transform teams to Agile. In the below section we are providing the way to derive such a model.

III. PLAN YOUR OWN AGILE FRAMEWORK

Organisations are unique. The vision, mission, purpose, strategy, execution of each organisation are unique. When organisations are planning for an Agile transformation they usually provide a few days of training and with no prior experience the execution of Agile starts in the teams. The below are the factors which affect the Agile execution in this framework oriented Agile implementation:

- Misaligned organisation goal and Agile transformation goal.
- Inexperienced roles.
- No prior working experience in Agile
- Tailoring base values for the sake of Organisation needs
- Unrealising underlying organisational impediments
- Agile first approach vs customer first approach

Due to the above few listed complexities, the agile transformation may lead to undesirable outcomes from Agile and may affect the organisation's delivery to their customers.

Organisations are unique and hence the Agile implementation model for each organisation should be unique. As the first paragraph of the Agnostic Agile Oath says, “This means one size does not fit all, one framework is not the answer, and the ‘what and the how’ of what needs to be done, should be suited to customer context and to a wider strategic vision.”[4]

How to devise an Agile implementation model for organisations will be a curious question for every agile coach. Here we present our idea to develop the Agile implementation model for any organizations.

Step 1: As- Is analysis using tools such as Interviews, surveys, questionnaire, leadership discussion etc.,

Step 2: Analysis based on the data and facts received and try to figure out the organisational impediments.

Step3: Derive the goal for the Agile transformation with consent of stakeholders.

Step4 : Devise the team's norms & Lean mindset. Lean mindset becomes the fundamental basic ingredient for success of Agile implementation.

Step5: Based on the outcome from step 2 and referring to the Agile principles and values the Agile implementation model can be derived.

Table. 1. Guidelines to develop framework based on Agile principles and values

	Agile Principle	Practice you can derive from the principle	Similar practice in Scrum/Kanban/XP
1	Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.	Iteration/Timebox. Customer focussed Reduce uncertainty Small releases. Prioritise the customer needs.	Sprint Iteration Product backlog prioritization Sprint review ceremony
2	Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.	Handling change frequency. Keep options open Flow of work	Estimation Just in time commitment Accept change in between sprints
3	Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.	Fixed Iteration cycle Low cost validations Faster validation of assumptions	Pair programming TDD(Test driven development) Spikes Sprint Planning meeting Potentially shippable product
4	Business people and developers must work together daily throughout the project.	Daily interaction meeting within teams Business should be ready for teams to answer any query Economics of small batch size Cost of Delay	Daily stand-up meeting Availability of Product Owner User stories Backlog Refinement meeting
5	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.	Psychological safety of team Measure work not workers Humble Agile	Empowered team members Self-managed, Self-decisive, Self-Estimating team members.
6	The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.	Communicate directly with teams Inter communication within teams seamlessly Visualized workflow	Face- to - face means avoidance of other literature medium like email, documents rather focus on human communication. Sprint backlog, Product backlog
7	Working software is the primary measure of progress.	Review meetings to measure the working software Measure the value of increment rather than amount of work done	Sprint Review Po validation of user stories Potentially shippable product
8	Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.	Common agreements like how business communicates the requirement to team like user stories, and the Artifacts, Estimation should be agreed. No hurry of work	Team is not forced to take work in cost of quality BDD CI,CD Continuous testing
9	Continuous attention to technical excellence and good design enhances agility.	New way of product design New way of Technical design	Value centric business prioritization Emergent design
10	Simplicity–the art of maximizing the amount of work not done–is essential.	Limit WIP, Metrics to measure the product progress and team progress.	Burn down charts, Cumulative flow diagram Velocity based estimation
11	The best architectures, requirements, and designs emerge from self-organizing teams.	Team culture Built in Quality Self-organised teams	Sprint Retrospective, Sprint review, daily stand up, Sprint planning A role called scrum master to facilitate Self-organized teams.
12	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.	Continuous improvement meeting Discuss every iteration on Product, Process and People.	Sprint Retrospective and all ceremonies help Inspect and adapt Faster feedback loops

	Agile values	Practice you can derive from the values
1	Individuals and interactions over processes and tools.	Team culture Communication between and with teams
2	Working software over comprehensive documentation.	Minimum artifacts More incremental delivery
3	Customer collaboration over contract negotiation.	Business and team communication on both ways
4	Responding to change over following a plan.	Responding to Change New way of project plan

Using the above guidelines anyone can derive the exclusive practices they need to deploy for the organisations to work in Agile model. This paper also promotes the usage of outcome driven matrix and not just matrix but the kind of questions the team should always keep asking each other from 3 perspectives to enable the transformation based outcomes and not output capacity.

Those 3 perspectives this paper proposes are Development Team, Business and Management.

A. On becoming more reliable:

Agile Delivery Managers/ Scrum Masters:

- Whether the stories are having dependencies
- What blockers/impediments are impacting story completion?
- Are we changing story sizing after sprint start?

Business:

- Are we adding/removing stories mid sprint?
- Do we have commitments from dependency tear for items we are picking up in the current sprint?

Management:

- Do we have the right amount of automation?
- Whether the team is taking the right amount of work?
- Does the team have T-shaped skills?

B. On achieving optimum velocity:

Agile Delivery Managers/ Scrum Masters:

- Whether team members are pulling work independently?
- Are there dependencies and blockers the team is waiting for?

Business:

- How can we have consistent velocity?
- What is impacting a team's velocity positively/negatively?

Management:

- How can automation or other development practices help the team?
- Are there any skill bottlenecks in the team?

C. On having more throughput:

Agile Delivery Managers/ Scrum Masters:

- Is the story size spread consistent?
- Are team members taking stories by themselves?

Business:

- Are stories in Ready state?
- What stories the team is working on?

Management:

- What's upcoming work and What training team members might require?
- Are there right technical practices enabling teams to work on completing stories parallelly?

D. On achieving consistent lead time:

Agile Delivery Managers/ Scrum Masters:

- How does it tie to feature release? (How to do just enough refinement)

- Why an item might be sitting in the backlog for >90days

Business:

- How is the trend in the last few sprints? Understand why the backlog is not moving?
- How relevant are those aging items (>90d or >180d) against the current priority context?

Management:

- Does Capability or Capacity deficiency adversely impact the Lead Time?
- Does Environment/Tool readiness causing a delay and in turn adversely impact the Lead Time?
- How do we get better?
- What are the impediments or dependencies that we have which is slowing us down?

E. On reaching to production well ensuring good flow skills and work:

Agile Delivery Managers/ Scrum Masters:

- Which stages/work items are taking more time to complete?
- What are the impediments, root causes and how to resolve it?

Business:

- How can we split stories while still providing value?
- Are we spending enough time for Backlog refinement?
- How can we enrich stories with more details?

Management:

- Is there a skill gap in the team?
- Whether regular feedback is provided?
- How good is the DoD and what XP practices being followed?

With this holistic approach the entire advantage of Agile can be gained rather than dogmatic, short sighted practice of any one part of the framework. This Agnostic approach also have been proved in few organisations with good results [5]

V. THE METAPHOR

Agile transformation cannot be an unified practice across organisations. Finalising an agile transformation model for an organisation is similar with making an flower bouquet. First checking the requirement of the bouquet, smell and choose the best flowers and pick the flower and bundle the bouquet. The flowers can be any of the frameworks available or even new framework innovated. Whatever may the framework but the lace which ties the bouquet together should be the Agile principles and values.

CONCLUSION

Agile agnostic approach strongly suggests that for each agile transformation the set of practices based on principles should be planned and developed exclusively. A team looking at agile transformation and learning better ways of becoming agile can look at the whole umbrella of agile methodologies. It is common for many in the industry to use terms like Scrum but in a pejorative sense, implying that teams should only be following the small set of practices that are part of Scrum. To deliver values to the customer in the way the customer wants and in the time the customer needs , Scrum/Kanban/XP by itself is not enough. Hence we conclude that for every Agile transformation the strategy, the plan and the approach should be devised exclusively for each teams.

Acknowledgment

We are thankful to Shane Hastie for his interest in this work. We are indebted to Harihara Prasath and Prasad Saranathan for their review comments and guidance. Mahesh Jade is thankful to his Family members and all the people he could interact with at various meet-ups in the last 5 years who helped co-learn and build a perspective that fosters agility on the ground. He is also submitting his due thanks to Vasco Duarte, Host, renowned podcast named Scrum Master Toolbox for his constant feed to real agile transformation stories, Padma priya is thankful to Padmavathy and Devarajan, Gokulakrishnan, Sangamithra and Prathuman. She is also submitting her due thanks to Shajuthomas , KnowledgeHut and Naveen kumarsingh for their support in Agile career.

References

[1] Tim Guay “Agnostic Agile: The Key to a Successful Lean Agile Transformation” 2018, https://www.infoq.com/articles/agnostic-agile-success,
[2] Jim Highsmith “About the manifesto” 2001,https://agilemanifesto.org,
[3] Kenneth S Rubin “Essential Scrum ”Addison Wesley, P 29-52, 2012.
[4] Sam Zawadi “Time To Inspect About “The Agile Model” 2017, https://thedigitalprojectmanager.com/perfect-agile-model-agnostic-agile/
[5] Adrian Lander “An Agnostic Agile case in Infrastructure and Service Management” 2017,
[6] https://agnosticagile.org/2018/05/an-agnostic-agile-case-in-infrastructure-and-service-management/#more-238