Author: Rahul Sudame

Title of the Paper: My 12 Agile blunders!

Theme: Project Management Leadership > In a Rapidly Changing World

Keywords: Agile, Scrum, Scaling, Lessons Learned

Abstract:

Many Agile enthusiast managers try to implement Agile methodology in all kind of projects and end up in burning their fingers, like me. Hence it is important to apply appropriate project management methodology based on the type, need and readiness of the project and organization.

I would be sharing some of my real life experiences where wrong application of Agile practices resulted in adverse impact on the project. 12 such experiences would be presented in the paper. First one is when we imposed Agile without creating a buy-in. In another case, we signed Agile contract for fix budget project and had to pull the plug in the end. We ended up building a bureaucratic environment while implementing SAFe and in another case, we got into doomed Agile state under the label of Disciplined Agile.
We also had challenges due to forcing Scrum for production support work, missing NFRs / Tech stories in backlog, forcing Metrics (Velocity & Agile Maturity Index) targets for projects. We also had some learning due to tussle between Architects and Scrum Masters, Islands of Agility and Stubborn project managers. We had one instance where team implemented Agile but there was no change for the end customer. And in another scenario, we used Agile when it was not really required.

All the practical experiences mentioned above would be presented with details and my learning from those incidents. The objective of the paper is to provide insights for implementing Agile in an appropriate manner.

Intent:

The intent of this paper is to share practical experiences and some of the failures in applying Agile practices. The key take away for the readers is to understand some of the wrong practices to be avoided, so that they can have successful Agile environment. The readers would have an opportunity to learn from the lessons learned shared by the author, which covers different aspects like contracts for Agile projects, Application of right methodology, Metrics in Agile environment etc. The readers will learn different best practices and guidelines for building outcome driven Agile environment.

Table of Contents

Introduction

Friends, do you remember an Akbar – Birbal story, where all the courtiers were afraid to tell the bad news to the emperor (that his beloved parrot is dead), but Birbal conveyed the message diplomatically? You might have met many Agile enthusiasts in similar manner, where they would tell only the good news about implementing Agile, but may not be comfortable in sharing the bad news. It is like recommending one single medicine for all kind of symptoms. But for successful transformation to Agile environment, being aware of the other side of the story is equally important.

Hence I am going to share some bad news where wrong implementation of Agile practices created more mess than actually translating into Faster – Better – Cheaper environment. All of these are my own experiences while trying to apply Agile in different projects & organizations.

Blunder #1: Imposing Agile without creating a buy-in

Scenario: A multi-national services organization decided to implement Agile methodology for its Information Technology division. The Chief Information Officer (CIO) believed that Agile methodology will help in addressing one the major concerns raised by the business team that the IT team is taking long time (actually years) to deliver the expected features. Due to the business pressure, the CIO announced that he wants to see the entire IT division following Agile in next 6 to 8 months.

Since this decision was taken by the CIO, there was no question of challenging it! The project delivery managers and Project Management Office (PMO) was tasked to start implementing Agile. But since this decision was imposed unilaterally on the team, it resulted in lot of chaos.

Impact:

- Disagreement on the interpretation of some of the Agile terms
- Conflicts between different sub-teams, process drivers and team members
- Sub-optimal implementation of Agile

Remediation:

Realizing the ground situation, the organization decided to invite Agile Coaches (including the author) to work with team and build the right environment. We started with first understanding the organization context and point of views of different teams. We first defined a roadmap consisting of creating awareness about Agile, defining single and consistent meaning of Agile practices in the organizational context, conducting a pilot and then starting the roll out in phased manner.

We then conducted training & series of workshops to create buy-in and address any concerns. We had many confrontations due to different viewpoints by the team members, but these candid conversations helped us get everyone on the same page and at the same build a tailored Agile methodology that was more relevant for the respective organization.

Blunder #2: Agile contract for fix budget project

Scenario: One of our customers came to us with a requirement of building a portal for specific user group. The customer was very passionate about this project idea and wanted to build this as a product through his bootstrapped startup organization. We were delighted to be part of this product development initiative and finalized the engagement with Time and Material (T&M) based contract, with an upper cap based on the initial funding available with the customer.

We proposed using Agile project management methodology. The intention was to build initial set of features, present it to the user group and fine-tune the subsequent implementation based on the feedback, instead directly presenting the finished product. This approach worked well and we received lot of feedback about the features we implemented. But at the same time, continuous feedback resulted into continuous changes and additional work for the team.

Impact:

- Confusion regarding scope, due to lack of defined product backlog or functional specs
- Increase in scope due to user feedback, even though the budge was fixed
- Agile contract lacked clarity on handling scenario of scope inflation

Remediation:

The customer's budgetary limitations and lack of discipline in managing the scope of the project started resulting in cost overrun for the performing organization. Since the availability of next series of funding for the customer was dependent on revenue from the initial version of the product, it was decided to launch the product with minimal marketable features. The future development requests and customer feedback was logged in a product backlog as next phase, based on funding availability. We also learnt a lesson (hard way!) that if the budget of a project is fixed, then we need to have a structured mechanism to manage the scope of the project.

Blunder #3: Overhead while implementing SAFe

Scenario: A multi-national organization identified Scaled Agile Framework (SAFe) as their preferred methodology for Agile implementation. When the organization decided to move towards Agile, one of the major areas of concern was the roles and responsibilities of different existing stakeholders. Since Scrum methodology does not explicitly mention Project/Program/Delivery/QA Manager or PMO as roles, many team leads and managers were concerned about the impact of this process transformation on their role.

To handle these concerns, the organization decided to fit the existing roles in Scaled Agile Framework. Due to this, we ended up having new roles of SAFe like Release Train Engineer (RTE), Scrum Master, Product Owner as well as existing roles in the organization like Project / Program & QA manager. So instead of building an Agile environment, we ended up in having heavier org structure ☹

Impact:

- Multiple decision making authorities resulting in delays
- Different of opinion between Project Manager & Scrum Master resulting in confusion for the team
- Conflicting directions to the testing team by QA manager and Scrum Master

Remediation:

The senior management team of the organization realized the impact of over-accommodating organization structure and decided to streamline it. We formed a team to review the current structure objectively in line with Agile principles. We reviewed the roles recommended in Scrum as well as Scaled Agile Framework for Portfolio, Program and Project level. Based on that, we identified the optimized organization structure and also clearly defined responsibilities and decision making authority for each role. The cadence of meetings for different roles was also defined.

We conducted different sessions for the entire organization to explain the org structure and role / responsibilities. We also defined interest groups for Product Owners, Scrum Masters and Project Managers so that discussions related with specific role can be organized and it helped in streamlining the overall process.

Blunder #4: Doomed Agile instead of Disciplined Agile

Scenario: An organization having many independent departments / portfolios and product lines evaluated different Agile methodologies (like Scrum, XP, Kanban, SAFe, LeSS, DA 2.0) as its preferred project management methodology. The organization has multiple compliance and statutory requirements, which mandates User Acceptance Testing (UAT) to be conducted by an independent team covering all the compliance requirements. The organization also needed a dedicated requirements capturing cycle, due to its demand-supply cycle involving multiple lines of business.

Considering these aspects and after evaluation of different Agile methodologies, the organization decided to adopt Disciplined Agile Delivery (DA 2.0) scaling methodology, which its flexibility to tailor the methodology as per organizational context. This model certainly provided flexibility to have a dedicated phase for Requirements capturing at the start of a new phase, a dedicated validation / deployment cycle at the end of the phase, and development happening in iterations. But it started resulting in mini-waterfall quickly!

Impact:

- Business Analyst working on requirement capturing as per their own pace and timelines

- Architects asked for dedicated time for converting requirements into detailed Architecture
- The UAT team worked as per their availability, since they had multiple projects in hand

Remediation:

In effect, even though the project was tagged as an Agile project using Disciplined Agile methodology, the actual scenario on the ground was very much waterfall. Realizing this, the organization scheduled a workshop with the business, management, delivery and process consultant teams to review the situation. The different teams shared their perspective and the rationale for their current operating model.

The process consulting team (including the author) presented a concrete recommendation of how the entire cycle can be shorted if the business team provides only a subset of requirements. Then the project team (including Architects, Developers and Testers) can build the incremental design considering the subset and start the implementation iteratively. We also decided to involve UAT and deployment team in the release planning meeting so that they would plan their availability accordingly. There was nothing dramatically new in this approach, but the realization / acknowledgement of current situation and willingness to change certainly made significant difference in the outcome of this project.

Blunder #5: Forcing Scrum for production support work

Scenario: One of our new product (mobile application) development projects worked successfully from Sprint 0 to Sprint 11, when we moved into regression testing phase. Since the team was predominantly working on testing and defect fixing activities, it was difficult for the team to project how many story points it would complete in the respective sprint. The project manager was still insisting on continuing the Scrum model and called the regression testing sprint as Hardening Spring.

The business team mentioned that the product cannot be released unless all the required defects are resolved. This resulted in extension of hardening sprint and in effect impacted all the metrics. The team also got confused about what to present during Sprint demo meeting. Some of the team members started working on backlog items for the next phase, which made the metrics calculation even more complicated.

Impact:

- Challenge for the sprint to predict the sprint backlog in regression testing phase
- Insistence by Software Quality Assurance team for all Agile metrics even during hardening
- Difficulty in using burndown chart and velocity due to variable complexity of tickets / defects

Remediation:

One of the major lessons we learned as a project team and organization that we should tailor and change the processes based on the phase of the project and one size does not fit all. We decided to pilot Kanban model during the production support team and during regression testing phase and it really worked well.

With Kanban model, the production support team maintained list of all the incidents / tickets and worked on it as per priority. We also used Kanban board to visualize the overall flow, providing us insight on readiness for the go-live. We changed the focus on completing the priority work items, instead of matching the sprint velocity and burndown chart. We also interacted with our customer and internal Quality Assurance team and mentioned to them about process tailoring based on the stage of the project and it was received well. This helped us to present relevant metrics and insights based on the project phase, instead of ‘cooking up’ data so that it fits in the conventional metrics.

Blunder #6: Missing NFRs / Tech stories in backlog

Scenario: In one of our project using Agile project management methodology in conventional Customer – Vendor model, we received list of epics from the customer. The team worked with customer’s product owner / business analyst to convert large epics into smaller user stories and defined product backlog. Then the team started implementing the user stories as per the business priority and presented demos at each defined intervals. The entire model worked fine until we reached regression testing phase!

During the regression testing phase, the production dataset was made available to the team. The dataset was large and it started resulting in performance degradation for the application. The business team mentioned that all the performance issues need to be fixed before the application can go live. During the same phase, multiple other release criteria’s were shared with the team like adherence to security standards, conducting vulnerability and compliance assessment, guidelines for including legal verbiage etc. This started resulting in significant inflation of the scope of the project. The customer’s management team mentioned that even though these requirements were not explicitly mentioned in the requirements specifications, is it implicitly assumed that the delivered product complies with all the non-functional requirements.

Impact:

- Late realization of additional non-functional requirements (NFR)
- Multiple unknowns (like client’s security requirements) unearthed late in the release cycle
- Conflicts between the customer and performing organization regarding in-scope and our of scope

- Additional efforts for the project team and impact to project timelines

Remediation:

Due to the nature of the contract, the performing organization had to work on these additional requirements and put additional efforts to have as less impact to the project timelines as possible. This directly impacted the profitability of the project. Based on these learning, the organization modified in contracting model to clearly define the items that are in-scope and out-of-scope, including non-functional requirements.

The categories of non-functional requirements were clearly defined, like Accessibility, Accuracy, Availability, Performance, Scaling, Auditing, Backup and Recovery, Concurrency, Configurability, Documentation, Error-Handling, Regulatory, Redundancy, Security, Usability etc. We also included all the relevant non-functional requirements in product backlog including its effort estimated. This helped us in having better planning and at the same time adherence to all the NFR expectations.

Blunder #7: Tussle between Architects & Scrum Masters

Scenario: When we moved to Agile methodology, the management team and project delivery team was super excited. The team was happy to get customer feedback earlier, due to which potential change requests and rework can be avoided. But as the product implementation started, we realized that the Architects group within the organization had some apprehensions. The Architects used to receive functional specs in waterfall environment, create low level / high level architecture, get it reviewed / approved by the Architecture board and then release it to the development team for implementation.

Impact:

- Dedicated design /architecture phase started resulting in delays for the development team
- The availability of Architecture forum for review / approvals became bottleneck
- The user story commitments as per the sprint plan directly got impacted

Remediation:

The push from Scrum Masters to build on partial design (based on only next few sprints) and resistance from Architects group considering overall product stability started resulting in lot of confrontations. Hence, we decided to have a focused discussion with all the involved stakeholders and discussed all the perspectives candidly. Then we came up with an optimal approach, to have a short Architectural Blueprint phase during Sprint 0 of a new product development project. For all the ongoing sprints, one Architect was aligned with every Scrum team who would focus on sprint specific backlog and build the design incrementally with the development team. This model is working well and helped in streamline our overall project development activities.

Blunder #8: Islands of Agility

Scenario: It is very common for any large organization to have multiple sub-divisions. We also had large sub-divisions and they were running as separate organizations with their own product lines. One of our product lines decided to move towards Agile methodology. We defined our release and sprint plan and during our planning exercise we realized that our Sprint 3 is dependent on some common utility functionality, which is owned by other product line in the organization. When we approached the manager of that product line, we realized that the other product-line was operating on waterfall methodology and they were planning make the respective utility available 1 month after our Sprint 3, since they had other high priority items on their plate. This suddenly added a major challenge for our planning exercise.

Impact:

- Dependency on non-Agile team directly impacted our sprint plans
- Different business priorities for different divisions resulted in inconsistencies
- The ability to plan was severely impacted due to uncontrolled dependencies

Remediation:

This islands of Agility was impacting our ability to plan with confident, since we did had a dependency on other island (other division), but we did not had control on their business priorities and plans. Hence, we decided to invite management & business team of the dependent division for our release planning meeting. This helped management and business teams of both the groups to review the collective product backlog items and prioritize the activities accordingly. After this agreement between both the teams, the teams followed their independent processes but coordinated regularly for integration of the work. This coordination and collaboration helped us to achieve better results, but consistency in processes followed by different groups within the same organization can certainly help in avoiding Islands of Agility.

Blunder #9: Stubborn Project Manager

Scenario: When our organization decided to move towards Agile, the roles and responsibility of multiple people changed. One of our project managers assumed role of Scrum Master and that was start of the disaster. She was very aggressive by nature and always liked to execute the project in her way. Being control freak, she started dictating the Scrum process based on her interpretations. During daily standup meeting, she would ask every team member about status of his work and confront about the effort estimations and delays. She also took control of the sprint planning and demo activities and created a detailed project plan for each sprint!

Impact:

- Significant demotivation for the team members
- Misinterpretation of Scrum methodology resulted in utter confusion
- Wrong implementation of Agile on ground resulting in chaos

Remediation:

The dominating approach of Project Manager started producing reverse effects instead of providing any benefits due to Agile methodology. The senior management team recommended the Project Manager to attend Agile & Scrum Master training and change management style to servant leadership style. But when the things did not change even after all of these efforts, the manager had to be replaced with a Scrum Master with right attitude and management style.

Blunder #10: Forcing Metrics targets for Agile projects

Scenario: We had another challenging scenario during our Agile transformation journey. The team moved to Scrum model and started delivering the user stories as agreed in the Sprint planning meeting. After few sprints, the Software Quality Assurance team asked for metrics data for the projects, since the organization is CMMi Level 5 organization. The team was asked to produce Schedule Variance, Effort Variance, Productivity, Defect Density, Cost of Quality and several other metrics. The team had moved to Agile and was not maintain some of these metrics. When the team informed Quality Control team about Agile nature of the project the Quality Control team defined and shared some metrics for Agile projects. It included Velocity, Story points planned vs. Delivered, Agile Maturity Index and also shared organization threshold for these metrics.

Impact:

- Forcing conventional metrics on Agile projects resulted in confusion
- The interpretation of metrics in Agile context (like doubling velocity every sprint) resulted in stress
- The team ended up maintaining lot of data points just to satisfy audit requirements

Remediation:

We realized that forcing conventional metrics on Agile projects is not helping; in fact it is creating more issues hence we discussed with Quality control team and refined the metrics in Agile context. This helped us in streamlining the process and we could really present the meaningful data and insights out of it.

Blunder #11: No change for the end customer

Scenario: In one of our Agile project implementation, we followed all Agile practices in consistent manner. The team conducted Sprint Planning, Standup, Demo and Retrospection regularly. The team was presenting the demo at the end of the iteration, but since the project involved multiple components and integration was planned at the last sprint, all the artifacts developed during prior sprints were getting shelved. Due to this the customer did not see any value of Agile, since he was still getting the end product at the end of 6 months, which was the case earlier.

Impact:

- Delayed integration resulted in non-consumable product throughout the sprint cycles
- The issues in integration made it a long cycle and further delayed the go-live plans
- The integration issues resulted in log of confusion and confrontations amongst the teams

Remediation:

After observing this pattern for one release and its impact, we decided to change our process model. We asked all the teams to work as one single, cohesive team and integrate their work items and the end of every sprint. We also reserved an Iteration Planning / Hardening sprint after 3 weeks, so that the integration can be thoroughly verified. This resulted in changing the production deployment cycles to every quarter instead of half year.

Blunder #12: Used Agile when it was not really required

Scenario: When we started working on a new project, everyone was of an opinion that Agile is the only option we should consider for developing this project. The project was for building a storage array, which needs multiple integrations with other components like firmware, Application Programming Interface (APIs) and third party components. The team soon realized that even though they complete all of their features, they cannot go-live since the hardware components and firmware was not going to be ready by then. The team also realized that all the customers also need dedicated validation cycles, since the new version of storage array needs to go through validation cycles.

Impact:

- Misalignment of expectations by one team vs. the other
- Frustration amongst the team that was excited to use Agile for each project
- Overhead of conducting Scrum ceremonies, since the outcome was not getting consumed

Remediation:

After analyzing the real customer need, we realized that this is a conventional waterfall project, where all the activities need to happen in the defined sequence and interim releases are not required. We also understood that even if we provide interim releases, it would be create an overhead for the customer, because he has to go through long validation cycle before applying new version of our story array. We also understood that there was no need to force Agile project management methodology for all kind of projects. Based on these learnings we created an Agile Readiness Checklist, where we started reviewing the customer's real business needs and then identify appropriate project management methodology.

Conclusion

Following are some of lesson learned through various project experiences mentioned above:

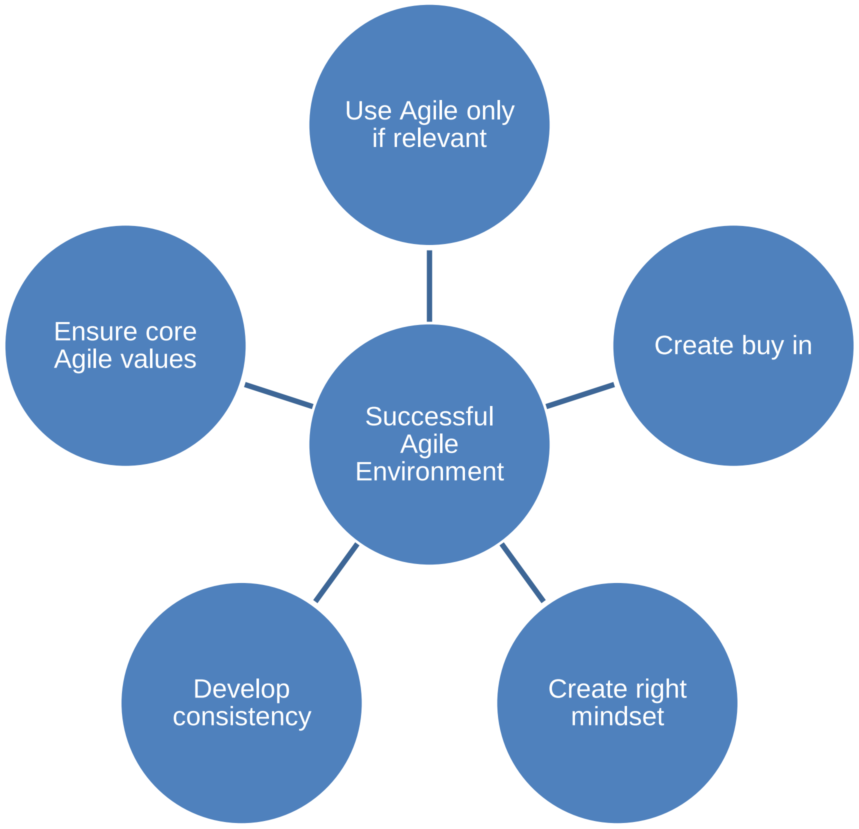

Figure 1: Lessons Learned – Agile Implementation

An Agile project management methodology can certainly provide significant business benefits and value for the end customer, if it is implemented properly. Poorly implemented Agile can create more damage than providing any benefit. Hence, it is important for every Project Manager / Scrum Master and other stakeholders implementing Agile projects to review their implementation regularly and should ensure that their projects are implementing Agile in appropriate manner.

References

[1] Scaled Agile Framework - http://www.scaledagileframework.com

[2] Large-Scale Scrum - less.works

[3] Disciplined Agile Delivery - http://www.disciplinedagiledelivery.com