International Journal of Engineering & Technology, 7 (3.27) (2018) 668-676

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Simplification Using the Levers of Agility and Devops at Large Financial Organizations

Anshu Premchand[1*], M. Sandhya[2], Sharmila Sankar[3]

[1]*Research Scholar, BS Abdur Rahman Crescent Institute of Science & Technology, Chennai.*
[2]*Professor& Head, Department of Computer Science & Engineering, BS Abdur Rahman Crescent Institute of Science & Technology, Chennai.*
[3]*Professor, Department of Computer Science & Engineering, BS Abdur Rahman Crescent Institute of Science & Technology, Chennai.*

Abstract

Most of the huge, conventional organizations we see today are, at a number of levels or the other, finding it difficult to entire in the market, assure their customers and create the software programs the business needs. This is since of the way we introduce and deliver software-there is require of agility with the purpose of business needs, additional constrained by the fact with the purpose of many teams that work for the similar project don't collaborate in the way needed. Large financial organizations have a completely different set of challenges that need tackling as compared with smaller companies. To solve this problem adoption of agility and DevOps principles for simplification and therefore transformation is more daunting; more so because simplification value needs to be delivered, measured, tracked and make a difference to end customers. This also ensures that some measurable value is delivered every step of the way. But to understand this value, we need to understand the issues that large financial organizations have with the traditional methods of software delivery such as the waterfall approach that have no space for agility and day-to-day business changes or needs. Agility and DevOps are response to many of these challenges in approaches such as waterfall method for software delivery in a large financial enterprise. This study presents a stark and welcome change from waterfall methodology where there will be very few examples of delivery within projected cost, time and effort. This plan will lead to micro-plans to manage each of the planned stages of agile transformation of the architecture space for the organization. It also focuses on simplification of complex technological world at large organizations and role of agility and DevOps therein to deliver maximum business & customer related results.

Keywords: *Financial organizations, DevOps, agility, customers, business, and waterfall approach.*

1. Introduction

Most of the large, traditional organizations we see today, are, at some level or the other, finding it challenging to complete in the market, satisfy their customers and create the software programs the business requires. This is because of the way we develop and deliver software-there is lack of agility that business needs, further constrained by the fact that various teams that work for the same project do not collaborate in the way required. This inhibits speed of change, competition and constrains new product creation and/or service delivery by financial business to the end customers. The story is very similar at all large financial organizations. They nevertheless try to replicate the success of using agile methodologies and DevOps at smaller software organizations by mimicking the agile team structures. The issue with this approach is that of size that one has to deal with at large financial institutions. Such organizations generally require integration of work-products across several teams that may span hundreds of software engineers. There are problems that cannot be fixed by one/two small agile teams. That is why it is extremely important to understand that DevOps and agility require top-down and bottom-up approach at the same time. Top-down, they need patrons of change and drivers of new agile culture and collaboration and bottom-up, they need actual grass root level software engineers to adapt to changes, assimilate new technologies, work together, collaborate and deliver in an agile way.

In this paper, we focus on simplification of complex technological world at large organizations and role of agility and DevOps therein to deliver maximum business & customer related results. "Agility" is an umbrella term for several iterative and incremental software development methodologies.

The most popular agile methodologies include

- Extreme Programming (XP)
- Scrum
- Crystal
- Dynamic Systems Development Method (DSDM)
- Lean Development
- Feature-Driven Development (FDD)

DevOps essentially means bringing "Development" and "Operations" teams together in a culture with heightened collaboration and communication for quick delivery (software and infrastructure) aided by "continuous pipelines" built using automation.

We present a case example which is inspired from a real life case of large financial organization where actual business value has been &is continuously being delivered by application of agile principles and DevOps at scale. This financial organization is from the Fortune-500 list where first wave of computerization started 40-50 years ago so there are ample opportunities &urgent need for simplification. The case example used in this paper is heavily inspired from our real-life experience with the said

organization where following benefits on adoption of Agility and DevOps were delivered (values very close to actual):

- 55% reduction in severity 1 and 2 defects in production
- 15% reduction in development defects
- $9 million worth of effort shifted to "Change the Bank" initiatives from "Run the Bank" initiatives
- $2.3 million savings due to lower rework
- 20% lower effort on development
- 45% reduction in total defects
- 10-25% productivity improvement in early adaptor teams

In this case example we found that most value of simplification via the levers of agility and DevOps is delivered due to improved collaboration and cultural change at the organization level along with productivity improvement. Hence the subject of this paper is how simplification of collaboration process due to agility and DevOps can bring about phenomenal business values just by collaborating well, integrating earlier than in current normal world and integrating 'ops' with 'dev'. Figure 1 shows a bird's eye view of the initial focus areas.

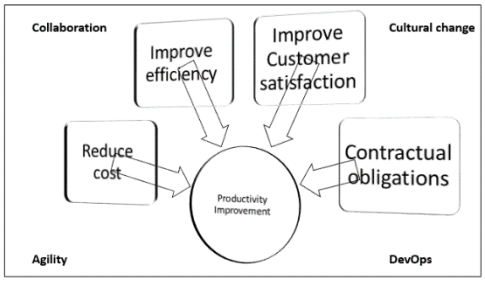

Fig. 1: Bird's eye view of focus areas

Integration early and collaboration ensure that conflicts are resolved very early in the lifecycle of software development and thereby lowering cost of correction. It is also important to note that simplification of IT systems at behemoths in the financial world is a continuous and continually improving process due to newer developments and business needs adding to the software serving business every day. Large financial organizations have a completely different set of challenges that need tackling as compared with smaller companies-so adoption of agility and DevOps principles for simplification and therefore transformation is more daunting, more so because simplification value needs to be delivered, measured, tracked and make a difference to end customers. This also ensures that some measurable value is delivered every step of the way. But to understand this value, we need to understand the issues that large financial organizations have with the traditional methods of software delivery such as the waterfall approach that have no space for agility and day-to-day business changes or needs. Agility and DevOps are response to many of these challenges in approaches such as waterfall method for software delivery in a large financial enterprise.

2. The First Steps

The first baby step, yet the most important one is for the top level decision makers to understand that their current processes and approaches are failing business needs or dragging the business and therefore not responding to end customer needs. This ensures everyone understands that going agile or DevOps is not because it is fashionable, but because it is required for improvement in turnaround time for business requests to technology and therefore quicker time to market and respond to customer needs. This is extremely important in the financial world because money is involved, and customers expect the financial organization they engage with to help them navigate the ever increasing complexity efficiently and in an agile way. Also, the most important people to lead this change in 'way of living' are the business and technology leaders because they understand where business and/or technology are challenged in the organization and therefore better positioned to identify objectives, go-no go criteria, priority, checks & balances and progress markers.

The process of simplification by going agile, using DevOps and improving continuously is a long and arduous journey which is made ever so challenging because it is difficult to measure process improvements in applications, code and software. Quantitative metrics alone will not help. A deeper understanding of the 'big picture', incremental simplification steps and areas that need improvement is needed. This needs serious collaboration all around. The key is to keep customer at the center of it all when looking for improvement areas. These key views are shown below in figure 2. If one kept process efficiency, operational management, mindset & behavior and organization and skills around customer in mind, one will be able to create a laundry list that can then lead to focus areas.

Fig. 2: Key views to identify improvement areas

Motivating the players is another key ingredient to ensure success, it is critical that all teams and team members feel important and motivated. With agility and DevOps, backlogs are created based on business need - the story boards and user stories that make the cut are the ones that make most business sense. Since continuously improving is a goal of simplification, knowledge management and learning become very important. Identifying what has happened, what did or did not work, who learnt what and how to spread the knowledge become critical because this will take the organization to the next step of simplification.

The plan for the simplification of the financial organization needs to be 'simple' so that the complexity of simplification doesn't kill the initiative. One would need to start small, and then expand the exercise. Agility and DevOps principles can be applied once key business objectives have been fixed and continuous improvement processes have been stabilized. The key to being agile would require the teams to be agile starting from the planning stage itself and DevOps needs to be internalized to ensure that code being created and delivered in an agile fashion also gets to the end of the cycle and deliver business results quickly. Hence DevOps pipelines (DevOps pipelines can be built for continuous development, continuous testing, continuous integration and continuous deployment) need to be built keeping in mind that code will have to be released to production in smaller batches, more frequently and as economically as possible. Of all of what has been discussed so far, the biggest mountain to climb will be building DevOps pipelines (and environments) that mimic actual production environments so that minimal effort and time is spent in promoting code to production once the pipelines are built. This means that the continuous deployment pipeline (longest in DevOps chain) has to be built for various projects / technology chains most prudently and efficiently to take care of that specific chain's issues and problems as this pipeline will provide quickly measurable and widely seen advantages across the financial organization's technology spectrum. One of the biggest advantages of DevOps pipelines is that it gives real-time feedback so that these chains can be improved and no time is lost in fixing errors in the code. It is best to add an agile or DevOps coach to the team who is well-versed with the principles of both and also knows enough about the organization and its challenges to be able to guide the teams, and set measurable, achievable goals. This is also important because successes will motivate the DevOps teams to achieve greater heights.

All financial behemoths have a patchwork quilt kind of architecture landscape due to decades of incremental development so adoption of both agility and DevOps will require massive technological changes and effort but the biggest changes will be to culture and mindset of people. These changes require patience and time but without the cultural change, the remaining changes will yield lesser value than anticipated. Both agility and DevOps are tectonic shifts from regular way of developing software but well worth the effort because of the business and financial benefits they can accrue with lesser investment, more stability and faster time to market.

3. Challenges with Increasing the Span of Agility

Small agile teams are not same as having agile teams across the breadth of a large organization. The business and technology stakeholders play a big role where a large scale adoption of agility is concerned. Typical and traditional adoption of agility for small teams has drawbacks that can be avoided so that benefits of a successful simplification exercise can be accrued.

The challenges of traditional ways that look at taking agility to scale are not the same as applying agile and DevOps methods at scale to a whole organization or a large team (>100-150 associates). The pitfalls of a traditional approach need to be understood if one were to ensure success of the simplification exercise using agility and DevOps.

If one were to pick any large financial organization across the face of the earth, one would realize that such an organization would be majorly using waterfall methodology of managing software development life cycle. In waterfall approach for example, by the time a large project reaches testing stage, the requirements may have changed drastically leading to serious cost and effort repercussions and heavy amount of rework and quality issues because of patch work development-a sort of vicious cycle begins here that becomes difficult to break out of and law of inertia also applies. To simplify, one needs to understand issues with currently employed software development life cycle approach and ensure that these issues are being addressed when using agility and DevOps. The entire premise of agility is that instead of completing each stage of software development fully and then going to next one which either does not have a feedback loop or has a very delayed and expensive one, there would be very short cycle of PDCA (Plan-Do-Check-Act). Just imagine if the waterfall method became two week long from initiation to deployment and allowed for feedback–that is agility principles at play. Now, to do this, the entire project would have to be split into small, manageable units that can complete their own version of waterfall and be ready to deploy/get deployed within a 2-3 week cycle. Together, these units, will become the whole complex project. In our experience, agility saves a lot of cost and effort this way. Also, we have found, in many cases, the end consumer is satisfied when 60-65% or initially envisaged features have been delivered and hence further effort is not required or is used to deliver "good to have" or "wow" features. This presents a stark and welcome change from waterfall methodology where there will be very few examples of delivery within projected cost, time and effort.

4. Managing the Shift from Waterfall to Agility

Going from traditional waterfall methodology to agile way of working is a huge change of mindset even for a small unit, let alone a large financial behemoth where not only technology and processes would need change and uplift but the business would also need delicate management. In our experience, many stakeholders only focus on technology solutions without really assimilating that cultural and mindset changes actually require more attention and effort, of course along with the technology solutions to aid the agile process. The biggest hurdle to cross is identifying sequence of changes related to simplification and improvement and rolling them out across the length and breadth of the organization. Then, the challenge of ensuring everyone follows the changes all the while keeping the teams motivated begins. Organization wide change management is a very powerful tool-one that should be used very carefully after understanding all the pros and cons of changes that are being rolled out for the simplification exercise. Here, business stakeholders and top level decision makers add value by ensuring that technology stakeholders understand which changes will be adopted faster mainly because of the benefits they bring, and also ensuring the technology teams are not impatient and understand the actual organizational capacity to embrace cultural, technology and process changes. In figure 3, we depict a sample transformation roadmap for a financial behemoth which shows a stage-wise, year-wise plan. This plan will lead to micro-plans to manage each of the planned stages of agile transformation of the architecture space for the organization.

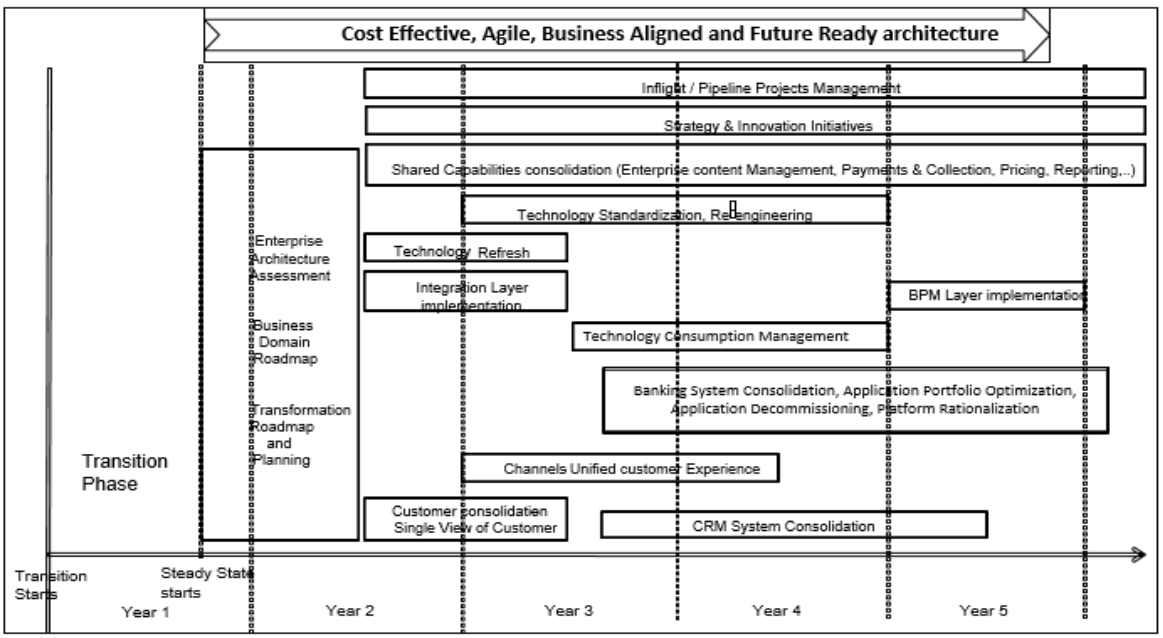

Fig. 3: Transformation roadmap for agile, future ready architecture

It needs to be mentioned that agile transformation is not flick of the hand kind of transformation, it requires patience and planning. Once implemented, agility leads to tremendous quantifiable benefits.

5. What is not Right with Scaling Traditional Agile?

This example is inspired from real life happening at a large financial organization. Assume a financial organization with a 500 people IT workforce. The financial organization has decided to adopt agile methodology of working. So, they hire a couple of agile coaches, and pick a few pilot teams and run these teams like a test bed. The agile teams will also need tools to manage agile lifecycle which would cost in the range of 0.5-1.0 million USD, depending on toolset, vendor, licensing terms and relationships. Once the test is successful, they create a plan for adopting agile methodology organization, so if there are 8-10 people in each agile team, at the end of this exercise there are about 50 agile teams in the organization. If each coach could help create 5 agile teams a year, the cost of this alone would be around 1 million USD (assuming 100-150$ per hour for an agile coach). Extending this very slightly to DevOps, if each team should own a complete functionality, then the team will have to have members from various component teams. This means a total reorganization of the 500 strong workforce will be required which will be an additional cost of 0.5-1.0 million USD. Adding all this will be the cost of transitioning alone. This kind of investment requires commitment from highest to lowest levels of the organization. Yet, in such a method, the middle levels of the organizational pyramid have no role to play. That is a big problem. Also, cohesion amongst teams, handover-takeover protocols, code releases, backlogs, pipelines have not been defined so these are even bigger problems, and then the culture which needs very delicate handling has not been addressed yet. We mentioned law of inertia earlier, which is an issue too. This is a lot of changes to make without any improvement in the way or timelines in which code is delivered to end users.

One of the fastest ways to look at challenges and identifying key improvement areas is by creating a fish-bone diagram. You will find that fish-bone helps you do a quick root cause analysis. Also, different focus areas require adoption of agility in different ways like resource utilization,competency management, communication rationalization, requirement management etc. all need context specific adoption of agile methodology/framework. Figure 4 is a sample of a fish-bone diagram to identify improvement areas. One can then create a priority list from the areas thus identified.

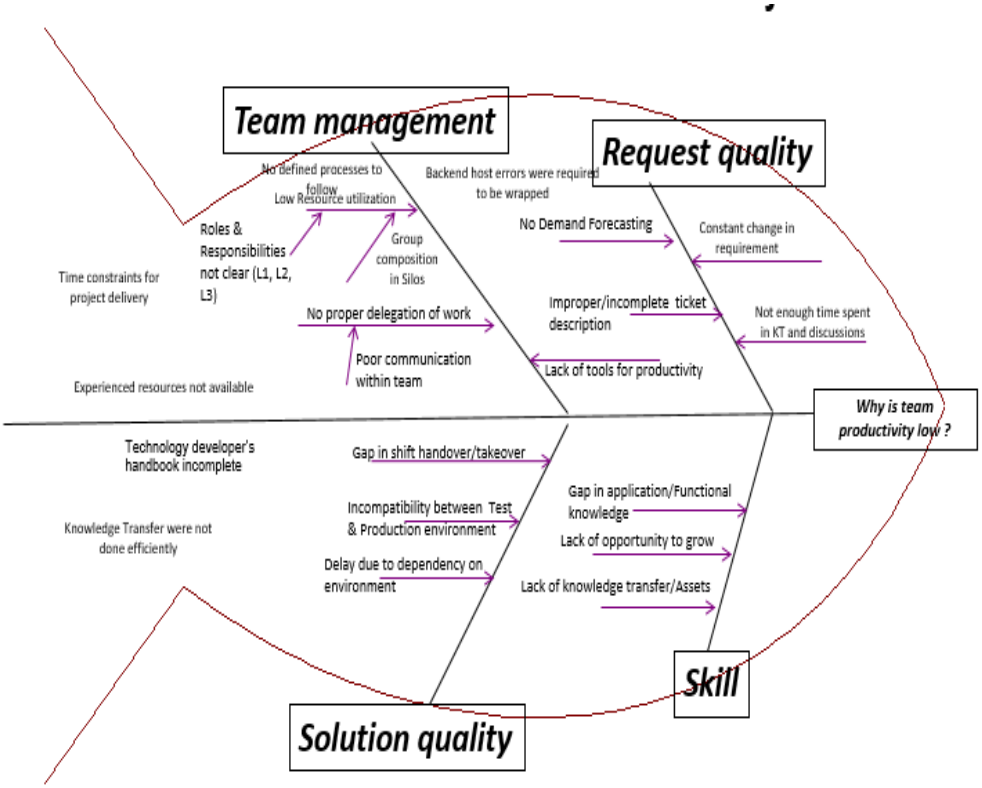

Fig. 4: Identifying key improvement areas using fish-bone diagram

This means that it is the duty of the business and technology leaders to define the framework, governance model and guiding principles but allow the teams flexibility to arrive at their own equilibrium. They will take more ownership, learn more, be more interested and adapt more quickly to any changes that one threw their way. A measure at this point will show starting results on any simplification and improvement scale the organization may have.

6. Planning for Agility

People who have worked in traditional financial organizations in business and/or technology roles for a period of time find it challenging to plan for agility and DevOps as they are not familiar with techniques of planning for these methodologies. They are comfortable with waterfall kind of model and assume that modern software can well be managed the same way. Also, it is impossible to plan perfectly for the duration that a project needs to deliver software under waterfall model. To take full advantage of agile workforce that employs DevOps is to be agile in planning and introduce the advantages of flexibility that agility has to planning for it as well. Software is inherently flexible, together with agility and DevOps, it should yield immense benefits to any organization. Agility and DevOps are especially helpful in areas where exploration is still being done for any number of reasons and requirements are not firm – which they never are, till delivery of

code. This is what all business and technology stakeholders need to understand.

We all know the importance of small things like wireframes to understand requirements which are very helpful in clearly understanding what business wants. Many times a requirement from business is misunderstood and there has been a price attached to fixing that error. The cost of such errors is higher in financial world.The way in-house / outside software development teams address such slippage and errors is in three ways. One, by adding buffer sometimes up to 50% of the actual predicted schedule so one would commit to software delivery in 9 months if the predicted time schedule was 6 months. Two, by adding more people to the team so a team of 10 gets inflated to 15 when there is slippage or errors are found and three, by making the team work 60-70 hour workweeks so the team becomes even lesser productive and various other issues crop up. Bottom-line is that the schedule is still seriously inaccurate.

In our experience, an excellent approach to agile and DevOps adoption is to look at the entire architecture space as candidate for continuous improvement. Then, divide the entire big picture into three levels (strategic, tactical and operational levels) and finally include all best practices from ITIL, Six Sigma, CMMI, ISO:20000 etc. This helps at several levels – improves stakeholder confidence that everything that has been a way of life so far is not being ditched, identifying pain areas (improvement cases) that resonate, contextualizing DevOps and agile adoption and so on. From our experience with several financial majors, and the case that we have been following, figure 5 shows an optimal method of identifying lowest hanging fruits, and candidate cases for agile &DevOps adoption when simplifying architecture and application spaces.

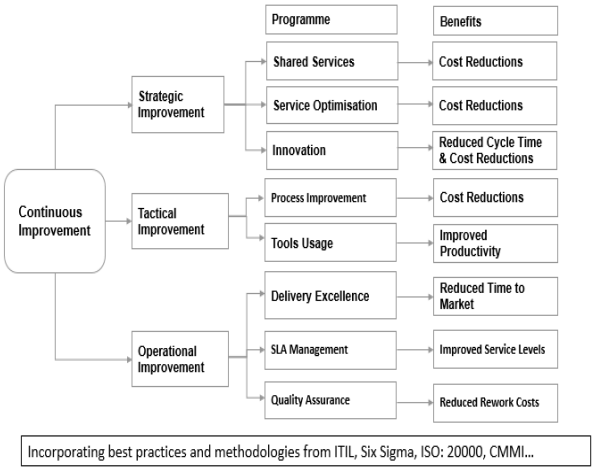

Fig. 5: Agile &DevOpsadoption approach

Unlearning waterfall way of working and learning agile and DevOps principles for not only software development life cycle but for planning for SDLC as well can take time and effort. The planning and SDLC both need to go agile to deliver tangible and quantifiable business benefit to the end customers.There can be no agile projects without agile estimation and planning.

This does not mean that there should only be short team sprint plans and no long term and medium term plans. Of course there should be as they are required for strategic direction and management but they should not drive all of the development effort. The issue with long term plans driving all of the development effort is that firstly there will be no capacity left for any alterations, and over time, requirements will change so alternations are inevitable, but there will be no capacity to address them. Secondly, where will the capacity to handle ad-hoc or new requirements come from? The backlogs should comply with the long term and medium term plans, but the backlog itself should be planned in an agile way so should the user stories. This will make the process flexible so that any immediate and urgent requirements can find place in the backlogs. Also, the best benefit out of DevOps and agility can be derived if these user stories go till deployment within the 2-3 week sprints and fetch end-customer feedback for the development teams. This means that business stakeholders are continuously giving feedback on whether the technology teams are complying with the strategic initiatives and their priorities are being managed.

7. The Financial Organization Case Example

Continuing with our example, the financial organization decided to move its complex claims processing and management system along with its external application suite to agile and DevOps methodologies after a pilot which was extremely well received.

The financial organization's goal was to roll out a completely revamped claims processing and management system along with its external application suite within 12 months. The new system was to also support all new digital platforms like mobile, tablets etc. Since the competition in financial market is very tough, on-time delivery of the project was critical. The organization also invested a lot of effort and money to revamp the planning process to ensure accurate planning. The system required a lot of code-porting and then re-engineering and development. It also came with a list of "Critical" or "Must Have" requirements from business users. Unfortunately, the business users had marked nearly everything as "Must Have" and existing work was already consuming 75-80% capacity. While planning and estimating, it quickly became clear to the team that all the "Must Have" features cannot be delivered with given capacity and existing application support and maintenance that had to continue for the year while the new system was being developed.

As one can well imagine, this was not a great situation. To break the logjam, decision was taken to re-architect, re-engineer and streamline the code so that pre-existing features did not take a lot of time and testing process was automated so investment was made for buying the right toolsets for the same as well. This ensured that about 50% capacity was freed up for taking up the new system. Going back to what we discussed on planning for agility and DevOps, this means that about 50% capacity was being used for long term plans and 50% became available to execute medium term and short term plans. So if the remaining capacity were to be split in 50-50 ratio, 25% was available to execute short term plans and another 25% was available for short term plans that could become part of priority backlog. This "layered" approach to planning brought in a lot of flexibility and reduced time spent on planning which in turn again became available for development.

The longest term plan in this case was for a year - to be able to make the claims system available which required a plan which was broken into medium term plan to look at 3 month and 6 months. Backlogs were created keeping these in mind. The short term plans focused on 2-3 sprints at a time and kept space for priority items that sprang up due to maintenance or changing business needs. The next step was to look at each user story and see where it fit the existing backlog. We estimated each backlog to be able to deliver 80 business requirements. All business needs were classified into stories and given a priority. This helped the teams decide which requirements will be delivered in a given backlog. The backlogs were broken down into detailed user stories as they came closer.

To begin with the entire back log would be discussed in each kick-off meeting for a sprint. Daily stand up meetings were conducted. The developers would also estimate requirements and break down the user stories into coding requirements or detailed design. This also helped the business re-prioritize as needed and be able to see system being delivered and give feedback very quickly - at a frequency of 2-3 weeks. The quick feedback cycle helped lower defects by 45% as stated in the beginning of the paper. An overall reduction of 15% in development defects was measured and a

20% lower effort on development of similar sized projects. This led to cost savings of 2.3 million USD due to lower rework as compared with similar sized projects. The initiative also helped release capacity worth 9 million USD towards "Change the Bank" initiatives. The teams realized a productivity improvement in the range of 10-25%. The cherry on the cake was the time saved on planning exercises themselves which contributed to the savings mentioned above and helped team morale as well.

It became evident to all involved that the savings were accrued because the planning went agile as soon as agile methodology and DevOps were adopted for SDLC. The agility helped in making sure that capacity was locked for a requirement only when the requirement became due.

8. Does Agile Planning Actually Work?

Agile planning works because it allows for a layered system of planning and lets the teams plan for each sprint which are very short duration in nature. Agile planning is “layered” into three levels – long term, medium term and short term. Figure 3 is an example of a long term plan. This essentially means that the long term plan which is split into two or more medium term plans and hence revised at least the number of terms there are. The medium term plan itself gets revised for each sprint. Every kick-off meeting and stand-up meeting is an opportunity for re-planning. Also, the teams learn from each sprint and apply that knowledge to planning for the next sprint. This reduces anxiety about the long term plan being accurate. This allows teams to focus on what is important than spending effort on trying to create a perfect plan. In the beginning of the agile journey in our example, a team committed to delivery of 150-250 user stories in two sprints (about a month). By the time the year was out, the same team was able to predict delivery of 160-170 user stories a month. Agile planning assumes that knowledge is not complete and is ever evolving. It allows for learning from current happenings and taking latest inputs from all stakeholders into consideration while planning, which is why it works.

Also, in agile planning, the estimates for size and duration are not confused with each other. The first step is estimation of user stories in story points and then a rate of progress is determined by the team. The estimates of story points and rate of progress are combined together to arrive at estimation of time duration. This clear separation of size and duration helps agile planning be precise and accurate.Agile planning at short term level takes inputs from the entire team, prioritizes user stories, learns, plans on facts, re-plans and tracks progress. All this makes agile planning work.

9. Planning for DevOps

The most fundamental benefit of applying agile and DevOps principles at scale, across the length and breadth of IT in a financial organization is that small functionalities can be developed and deployed at the speed of a hundred meter dash at relatively marginal cost to business and not at the end of a marathon like in waterfall method at a much higher cost. This means that the IT teams have a "pipeline" for continuous development, continuous testing, continuous integration and/or continuous deployment that help release code into production very quickly so that business users can give feedback at the speed of sprints. In turn, this means that end customers get to use incremental features which aids generation of business. This creates a victorious cycle that strengthens IT teams’ confidence and belief in the change to simplification which helps business confidence in turn.

This benefit is not available to teams that have looked at DevOps and agility through the waterfall glass and hence have scaled agile teams but not created pipelines. The primary purpose of DevOps is to create teams that have no demarcation between development and operations so that pipelines can be created which allow code for new functionalities or changes to flow into production and be available to production quickly and without human intervention as far as possible. As you may have guessed, automation is applied to great effect while creating pipelines for DevOps. The reason we said pipelines can be for continuous development, continuous testing, continuous integration and continuous deployment is because in DevOps "One size does not fit all". The pipelines and their governance is very context specific. In some places, it will be possible, feasible and profitable to create continuous deployment pipelines while in the others, one may be able to create pipelines upto testing or integration only. Such opportunities will be explored in detail in our next paper. DevOps alone with agility give massive efficiency, effectiveness and productivity improvements for any large scale simplification exercise as in the example of the financial organization we have been using in this paper. We cannot emphasize enough on the importance of cultural change while embracing DevOps and agility. Without a change in waterfall mind-set, the benefits of simplification through these two levers cannot be realized.

While undertaking the simplification exercise, the key objectives have to be kept in mind at all times during the transition to agility and DevOps. Now some of those objectives will be obvious and documented, some will not be. It is of utmost importance that the criteria of judging success is well understood by all concerned parties and agreed upon. When I was young, I always used to wonder why my mother would prepare one curry and two vegetable dishes for lunch because we would all be at school and eat only dinner at home so once I asked her and she said that is the way my father liked it. You see, the primary success criteria is very clear to her. Of course there are other things as well, like taste, freshness of food that matter.

In our example of the financial organization, apart from the long term plan for the new claims system to be up and running within 12 months, the success criteria was very clear for simplification exercise using agility and DevOps principles:

- Shorten the cycle time of feedback from business reaching developers
- Drastically shorten the production release cycle time
- Manage cost of development and production
- Reduce cost of error correction
- Manage cost of planning
- Remove duplication of effort throughout the development lifecycle
- Improve team level efficiency and productivity
- Improve efficiency and repeatability of testing, integration, building and deploying of code
- Use automation to reach aforementioned goals

Neither agility nor DevOps can be adopted overnight and like we said "One size does not fit all", so the time required for adoption of these principles would vary from one organization to another. Hence, success criteria is extremely important which is well articulated and clearly communicated to all stakeholders. This success criteria would help identify areas that need immediate attention and the ones that can wait awhile etc. Again, culture and agile planning play a very important role in DevOps adoption and so does tracking of changes being done for the ultimate goal of simplification. Pilots are important in the journey and applying DevOps principles across the organization will need evolution of the entire software development life cycle to ensure frequent production releases are supported, the process is repeatable, ever evolving and ever improving.

10. Stirring the Cultural Pot for DevOps

DevOps adoption across IT space for a large financial organization has two non-negotiable attributes. One, cultural and mindset change and two, technology solutions for automation to create "pipelines". Of these, bigger task is changing the way

people work and think across the organization. No amount on investment and planning can help unless right tools are used to develop pipelines and developers across the board use these pipelines to continuously deliver code (test, integrate, move to production) in an environment which mirrors production and receive continuous feedback. There is a mental barrier in the technology and business community because DevOps turns conventional thinking on its head. Law of inertia applies and so does the fear of the unknown. That is why we said that agility and DevOps adoption can be a success only with a combination of top-down and bottom-up approach. The business and technology stakeholders must keep at the task of motivating, managing transition and change, planning, monitoring and once the developers adopt these methodologies whole heartedly, they will see for themselves the benefits which include productivity and efficiency improvement, lesser downtime, more freedom and effectiveness to conduct their tasks etc. Once they experience all these and weigh benefits over waterfall kind of methodologies, they will become flag bearers of the simplification exercise. A word of caution though, the initial exercise of DevOps& agile team creation might shake up trees that cause jitters, this will need delicate handling. The management teams will also have to initially enforce following of rules, which will require some firmness. They will also have to insist on ensuring feedback is looped back into the system so that the pipelines learn, adapt and improve continuously.

On the technology side, along with learning to frequently release code, automate, reduce manual intervention etc., the developers will have to learn agile and DevOps principles, code control, re-architecture, reuse, neo-design and estimation techniques.

In our case example of the financial organization, positive reinforcements (star performer, star team, soft benefits, most improved team etc.) were introduced to ensure the rule book was followed. At the same time, disregarding rules led to immediate flagging to senior project manager and discussion to insist upon the importance of following the rule book. Some examples of rules from the rule book are:

- The build is sacrosanct and nobody can commit code to a broken build.
- The developer responsible for building code will ensure integration completion before moving to next task.
- The responsibility of integration and build release will be rotated across all members of the team in a round robin basis.
- Everyone has to be present for a daily stand-up meeting.

Enforcing the rule book and ensuring pipeline effectiveness will take a lot of effort from the management team initially. In the financial organization's case, the process of ensuring that nobody added code to a broken build was initially manual. After a couple of iterations, a source code management and version control tool was pressed into service to eliminate manual intervention. This led to additional productivity and efficiency improvement and also helped remove a major mental burden for the team leads.

We learnt that we should not wait for the pipeline to be built completely and tooling to be complete before starting the process, instead allow it to be built in a spiral, ever improving way after of course setting up the most important check-dams. We also learnt that continuous feedback and continuous improvement have a big role to play in the entire scheme of things. Investment in tooling and improvement should be a continuous process. So driving change in culture from top to bottom is important but equally important is bottom-up adoption of the simplification process.

Simplification at such a large scale requires patience above everything else. It is a transitioning, planning, pipeline building and management challenge. If the cultural change is not managed well, then technology toolsets will not be able to help. All the teams and developers must comply by the rulebook and everyone should take responsibility to ensure the transition to bringing development and operations teams together into DevOps mold works. Most of the developers will possibly argue against the change till they see it deliver value. Till then, the business and technology stakeholders must continue driving the change.

11. Building a Strong Core

Building the core for agility and DevOps is like as developing a kernel for an operating system or building foundation for a building. Unless the core is strong and solid, there will be unnecessary and avoidable struggle during the simplification process. Look at the core as key enablers for agility and DevOps. We understood from our financial organization example that DevOps and agility require a lot of automation but the automation has to be done with a clear purpose or else it will become a logistical and maintenance nightmare and not serve the purpose at all.

There are three fundamental pillars of the core. First pillar is to ensure that the architecture allows for smaller sprint teams (8-10 people at any point in time) to be effective and run independently. The second pillar is to create a code management system that aids reuse and helps create seamless builds. The third pillar is assurance or test automation. We have already stated that DevOps cannot be adopted unless we build pipelines. Assurance automation is a key step in building those pipelines. We learnt that these three are most important things to do before making sweeping organization wide changes for applying agile and DevOps principles.

In building a strong core, measurement plays an important role. Without measurement and metrics, it will become challenging to continuously improve agile and DevOps processes. For example, in the case of automation, if one is not able to statistically analyze the test metrics, one will not be able to draw much benefit from test automation as it is practically impossible to look at thousands of test results every day. Plus we are adding human intervention into the process which would mean delays and potential misses. Test metrics will allow identification of patterns, and fall/rise in accuracy which can then be used to look at specific test results to drive quality.

In financial organization case example that we have been using, test automation can be used for documentation purposes because any artifacts including code that deal with money come under close scrutiny because of laws like Sarbanes-Oxley Act of 2002. Until these three pillars are in place to ensure strong core, the simplification process using agility and DevOps as levers will not be able to deliver optimal results.

12. Measuring the Improvement from Agility and DevOps

Sooner or later, the teams and stakeholders will have to decide on the metrics and diagnostics to be used to measure success of the simplification program using the levers of agility and DevOps. This is necessary as metrics will help improve trust levels with business, increase visibility of work accomplishment, help identify issues and aid course correction, if needed.

The metrics and diagnostics chosen will have to be contextual and will have to make sense to all stakeholders. For example, in our financial organization case, we found cycle time and velocity to be a very important metrics from agility and DevOps adoption perspective to begin with.Cycle time is a program/project level metric whereas velocity is a sprint level metric. Cycle time describes the amount of time required from the start of the development process to the beginning of revenue generation from what was developed. Velocity is a metric that provides information about the “rate of progress” for the team. For example, velocity can be the number of user stories a team can perform in one sprint where it is important to include testing and making the user stories available in production in the sprint.

We also learnt in our case example that the metrics that the teams would find useful will change over time. For example, velocity is a most useful metric at sprint level to begin with because it allows teams to predict how much work can be accomplished based on prior results. But as time progresses, for a stable team in same project, the velocity will plateau. So in longer projects that have stabilized, teams can choose to suspend collecting velocity metric as it becomes near constant. Also, not all metrics can be compared for different teams. Again, using the velocity example, we learnt that team velocities make sense only to the team providing data – different teams will measure their work differently. Velocity measurement helps the team itself till such time that it plateaus, it cannot be used to compare team performances.

Some of the metrics that we found useful to stakeholders are:

- Cycle time
- User stories handled versus user stories released to production
- Monthly releases to production against target
- User story completion rate in each sprint
- Average process time
- Average on-hold time
- Estimated efficiency per month
- Defect density
- First time pass rate
- Estimation accuracy per sprint
- Velocity
- Metrics around quality of build& testing

Choosing good agile metrics is important because

- They will improve trust with customers
- Motivate the team
- Enable improvement
- Identify issues
- Enable learning
- Re-inforce desired behavior within the teams

The teams will need to build a simple yet effective dashboard with good metrics. In our financial organization example, we learnt that communicating metrics clearly and visually is as important as collecting them. Good metrics are easy to collect for the given context. They help measure trends, measure results, not output and should be vital few. Good metrics help the teams make better decisions. They are also a vital part of "Continuous Improvement" that we have been saying is critical to adoption of agility and DevOps.

13. The Cycle Called Continuous Improvement

It will be extremely beneficial to look for opportunities for improvement. We have emphasized upon this point in figure 1, 4 and 5 as well. A simple classification of continuous improvement, as in figure 5, can be on the basis of nature of initiatives - strategic improvements (program level, innovation, shared service creation etc.), tactical improvements (process improvements, tool usage etc.) and operational improvements (assurance, service level management, code quality management etc.). Continuous improvement has to be a "continuous" focus for all stakeholders. Once the initiative has reached steady state, it is imperative that opportunities for improvement are sought. In our financial organization example, one of our key improvement area was identification of repetitive tasks and their automation to reduce cycle time, reduce defects and reduce rework. So we analyzed process maps, chose "low hanging tasks" to begin with, designed and implemented or bought and integrated tools to occupy that space.

Continuous improvement focus will also strengthen stability of the system. One extremely important area for improvement is ensuring stable code is available for builds, this means ensuring a strong development process which helps continuous deployment. Focusing on this ask will help build automated tests and also improve stability of deployment pipeline over time. The idea is that teams should be able to release code as frequently as needed, and in as small batches as needed. This means that deployment pipeline should mirror production and have little or no manual intervention. This was a huge learning experience for us in our case example because when we began, every code promotion to next environment required signoff from technology and business managers which would invariably induce delays.[17][18] This sign off requirement was over and above requirements of user story sign-off, no defect clearance, release criteria being met and having a green build. All these were areas of improvement for us. Tracking these metrics allowed us to identify areas that needed work to ensure stable deployment pipeline and hence less work leading to productivity improvement. It is important that the teams look at continuous improvement as a part of their daily duties because that will ensure that looking for opportunities of improvement and working on those opportunities will become part of larger organization culture. Also, statistical analysis helps compare various results and draw conclusions that drive continuous improvement action.[19]

Honestly, this requires a lot of commitment, vision and rigor but anything that is worth one's time and money does not come easy. But continuous improvement focus will help the journey towards change of organizational culture and processes for the better.

14. Conclusion

The technology and business stakeholders should not get overwhelmed by the magnitude of the simplification exercise using agility and DevOps. As we said, it is not for the faint of heart. Everyone needs to understand that the world we are living in and its demands on business will make it impossible to survive competition with waterfall kind of methodologies. So the focus needs to be on driving cultural change, tooling to help that change and ensuring an organization wide focus on continuous improvement. The objectives, go-no go criteria, priority, checks & balances and progress markers have to focused on to ensure the organization is able to see progress along its path towards simplification and also help stay on target. Continuous improvement should get due attention and hence dedicated effort. One would do well to remember that the reason why agility and DevOps work is because they allow frequent measurement and course correction. Religiously automating, measuring and adjusting path will show results that will strengthen all stakeholders' resolve to stay the path. There will be confusion and resistance to change, but a few iterations through, if done well, benefits of the exercise will convert even most stringent sceptics. Allow leeway to the teams to arrive at their own equilibriums and continue motivating. Measure, improve and measure to ensure simplification yields quantifiable benefits to the organization. Get, set, go!

References

[1] Grebe M & Danke E, "Simplify IT: Six Ways to Reduce Complexity", BCG Perspective, (2013).

[2] Carbone JA, IT architecture toolkit; Enterprise computing series, Prentice Hall PTR, (2004).

[3] Michellod N, "System Standardization in Insurance: The Complexity of Simplification", Celent, (2015).

[4] Surendar A., Arun M &Periasamy PS, "Hardware based algorithms for bioinformatics applications-A survey", International Journal of Applied Engineering Research, Vol.8, No.6, (2013).

[5] Fauscette M & Perry R, "Simplifying IT to Drive Better Business Outcomes and Improved ROI: Introducing the IT Complexity Index", IDC, (2014).

[6] Premchand A & Choudhry A, "Industrialization of IT Services in Banking", International Journal of Computer Science & Engineering Technology, (2014).

[7] Dybå T & Dingsøyr T, "Empirical studies of agile software development: A systematic review", Information & Software Technology, Vol.50, No.9-10, (2008), pp.833–859.
[8] Ilieva S, Ivanov P & Stefanova E, "Analyses of an agile methodology implementation", Proc. of the 30th EUROMICRO Conference, (2004), pp.326–333.
[9] Karlstr¨om D & Runeson P, "Combining agile methods with stage-gate project management", IEEE Software, Vol.22, No.3,(2005), pp.43–49.
[10] Larman C, Agile and Iterative Development: A Manager's Guide, Pearson Education, (2003).
[11] Barros A & Kylau U, "Service delivery framework-an architectural strategy for next-generation service delivery in business network", Annual SRII Global Conference (SRII), (2011), pp.47-58.
[12] Erbes J & Motahari-Nezhad HR, "Sven Graupner-HP Labs, The future of Enterprise IT in the Cloud", IEEE Society, (2012).
[13] Terzidis O, Oberle D, Friesen A, Janiesch C & Barros A, "The internet of services and USDL", Handbook of Service Description, (2012), pp.1-16.
[14] Fagerberg J, Mowery DC & Nelson RR, The Oxford handbook of innovation, Oxford university press, (2005).
[15] Leverhawk Transformation Case Study–State Street, http://leverhawk.com/it-transformation-case-example-state-street, 2012.
[16] Abikhanova G, Ahmetbekova A, Bayat E, Donbaeva A & Burkitbay G, "International motifs and plots in the Kazakh epics in China on the materials of the Kazakh epics in China", Opción, Vol.33, No.85, (2018), pp.20-43.
[17] Akhpanov S, Sabitov R & Shaykhadenov R, "Criminal pre-trial proceedings in the Republic of Kazakhstan: Trend of the institutional transformations", Opción, Año, Vol.33, (2018), pp.107-125.
[18] Galindo GC, "Del Prometeogriego al de la era-biós de la tecnociencia Reflexionesbioéticas", Opción, Vol.33, No.82, (2017), pp.114-133.