Lean or Agile: Lessons Learned from a Tech Startup[1]

Case Introduction

Which method is the best for managing uncertainty in startup-like environments - Lean or Agile?

This is the question that Second Nature Software LLC faced when beginning with one goal in mind: start, develop and build a successful product company. The three co-founders had over twenty years of software development experience, and about ten years' experience using Agile. However, they had no market, no product, and no identified customers - and only a small amount of savings. By the end of their first year, Second Nature Software had built a data science product that was in trial at the largest medical research organizations in the world; including three institutes at the National Institutes of Health (NCI, NIAID, and NCATS), Johns Hopkins University, and the University of Maryland School of Medicine.

There are many competing perspectives on what works best for success in organizations discovering and building new technology to sell in the marketplace. Startups, especially technology startups, work in such extreme conditions that they have only a 10% chance of succeeding [1,2]. With so much uncertainty, traditional business plans, schedules, resource matrixes, and requirements specs do not last long enough to write them down. Teams must be willing to change fast and leverage management processes that can ensure order in a chaotic environment.

In the case study of Second Nature Software LLC, both Lean, Agile, and Hybrid methods were used at different stages in the company's development. The case study offers details of the how the methods were employed, the tools used, the success of employing the techniques, and lessons learned for how to do it better. The Conclusion then judges which method is better for most startup and extremely uncertain project environments, based on these real-world experiences.

Lean vs. Agile - What's the Difference?

Most project managers today study Lean for process improvement and Agile for delivering projects. This general classification makes sense given the history of the techniques. Lean was invented for manufacturing cars, and agile was conceived for coding software. The goal of lean was to manage capacity and have the requirements for production "pulled" from the customer with *varying demand over time but defined orders* (e.g. scope and price) [3]. The goal for agile was to manage changes in what the customer asked for during development with *varying orders but defined demand over time* (e.g. schedule and price) [4]. If we map these concepts, as well as the traditional approach of varying price with fixed scope and schedule, we form what is known as the triple cost constraint:

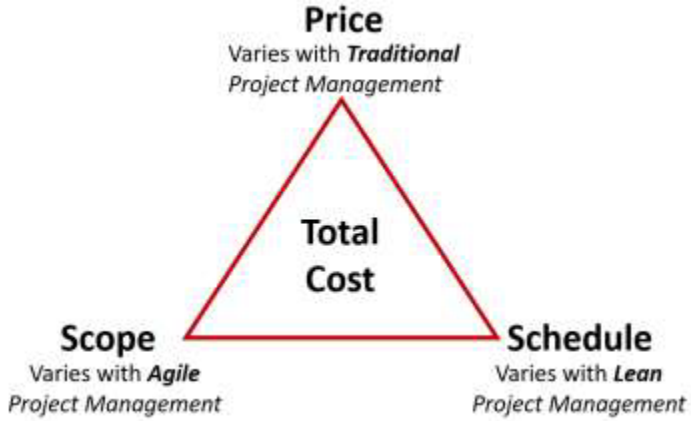

Figure 1: Triple Cost Constraint Alignment to PM Methodologies

In practice, however, it is difficult to understand the true differences between the Lean and Agile methods that are often considered variants

[1] Case adapted from Jj.Johnson: Lean or Agile: Lessons Learned from a Tech Startup

For Agile, one should have a certain customer and team fully capable of doing the work, but uncertainty which business needs will achieve the customer's business goal. Agile manages uncertainty by varying the scope of the project and maintaining a singular focus on the customer and their ability to express business needs. Therefore, Agile fits when the customer is not changing and the business needs are able to be prioritized at least for a set period of time. Agile's goal is to ensure that only the truly necessary business needs are satisfied to reduce the price and time to delivery.

In other words, Agile asks the question: *"What are the business needs that must be satisfied?"* Agile focuses on *business needs uncertainty.*

Lean, on the other hand, is best fit when one has a fixed set of resources and clear scope, but uncertainty in what the time and effort ("technical needs") really are to achieve that scope. Lean manages the uncertainty by prototyping, "learning by doing," and iteratively elaborating the technical needs to reduce time to delivery. Therefore, Lean fits when the business needs are certain but the technical needs are unknown. The goal is to learn what technical needs are required and achieve them as efficiently as possible.

In other words, Lean asks the question: *"How can the business needs be satisfied efficiently?"*
Lean focuses on *technical needs uncertainty.*

However, it should also be said that every method will in fact manage change in scope, schedule, and price throughout a project. Lean often leads to changes in scope driven by learning what is needed to deliver. Agile leads to changes in schedule by learning what not to deliver. And every Traditional project manager will use systems engineering to reduce overlapping scope and hasten delivery to be on time and budget. But it is the emphasis of either varying price, schedule, or scope that differentiates project management methods and the approach to managing uncertainty.

Case Study Background and Overview

Second Nature Software LLC was founded by three former IBM employees: Matthew Garlan, John Johnson, and Michael Pato. At IBM, the team delivered an award-winning project for the National Archives, where Michael and John served as the two Agile Project Managers for the 50 team members, and Matthew Garlan served as Development Lead and Architect. The project called ERA 2.0 implemented a cutting-edge, cloud-based platform for the National Archives to process and store petabytes of government records; and was awarded Project of the Year in Project Management for all IBM projects (the company employees over 400,000 people).

After completing the ERA 2.0 Pilot, the startup co-founders decided that they wanted a change from services to build products that could sustain them financially and do good in their communities. At first, the startup was named "One Year LLC" because the team gave itself one year to be successful. However, this name was quickly changed to "Second Nature Software" as the startup did not want any customers thinking the company was unstable. The team had no product or market, but managed to navigate its way to becoming a data science products company for the medical research community.

Second Nature Software's primary product, "Rocketfish," is a data management automation tool that is faster than Excel, while being more traceable than scripting languages like Python or R. This case study covers the first year of the startup's history and its first three phases of selecting, building, and distributing the Rocketfish to the DC-Baltimore medical research community:

- ***Phase 1 - Going Lean on Product Selection***. This phase covers the process of selecting product, Rocketfish, by evaluating team strengths and market needs. The team used Lean project management to execute this phase of startup.
- ***Phase 2 - Build the MVP with Disciplined Agile***. Here the founders work to build the Alpha version of Rocketfish to get a minimum viable product (MVP) in the hands of medical researchers. The team used Agile for this phase.
- ***Phase 3 - Distributing Rocketfish with a Hybrid Model***. With the Alpha built and in use, the co-founders worked using Lean and Agile methods to both respond to feedback and build new features needed to attract major institutes into product trials.

Second Nature Software LLC is still operating today and has almost medical researchers in trial with Rocketfish across institutes at NIH and major medical research universities globally.

Phase 1 - Going Lean on Product Selection

Second Nature Software started with a small team and a simple goal: discover, develop, and distribute a product for sale within one year. The team had experience starting companies before, but never without a predefined service or product. Therefore, a few of the initial steps were already known but mostly the plan for Phase 1 existed just as high-level objectives:

1) Establish the Company Formally (charter, bank accounts, operating agreement, etc.),
2) Set up Business Systems (Email, Website, Business Cards),
3) Determine Process to Discover Product Options,
4) Execute Process to Discovery Product Options, and
5) Select a Product.

The first two objectives were quickly achieved in the first week. The third, determining a process to discover product options, drew heavily on Customer Development by Steve Blank [5]. Using this blueprint, the Second Nature team could map out a scope of how to identify a product that best fit the team:

1) Evaluate Team Strengths (Experience, Expertise, and Social Capital)
2) Evaluate Markets that align to Team Strengths
3) Select a Target Market
4) Interview 30 Decision Makers in the Target Market
5) Map Business Process, Archetypes, and Customer Needs (Pain Points and Gain Points)
6) Generate Product Lists
7) Evaluate Products against Team Strengths and Market Needs
8) Select a Product

No one on the Second Nature team was truly qualified to execute any of these tasks as an expert. However, the team had systems engineering and management backgrounds, so they felt they could figure it out with “learning by doing.” At this point, the team decided to formally use Lean management to execute the plan efficiently and quickly get to building software products.

Lean Planning and Mechanics

Lean project management is actually very simple in terms of planning and mechanics. First a charter is written detailing goals and a plan that includes clear a set of milestones. Then the first major milestone is broken down into component work elements, or “stories.” These stories are written to ensure the business need is clearly understood in terms of what the work is and what they expect output should be. The stories are designed to take no less than one day and no longer than one week. Stories are then put on a simple board showing progress called a “Kanban,” which means “billboard” in Japanese. There are five positions for each story:

- New - these stories are written as potential work, but not yet possible to deliver
- Ready - these stories are ready to be executed, but not yet being worked on
- In-Progress - these stories are currently being worked on by the team
- Ready for Review - stories are complete and ready for team verification
- Done - stories are verified complete by the team

Each story carries its entire information, updates, and attachments for documentation posterity and reference. Every story is assigned to one team member as the owner, but can be worked on by any team member. Every team member can see every story on the Kanban. This creates a fully transparent Lean Plan for accomplishing the scope for the first major milestone.

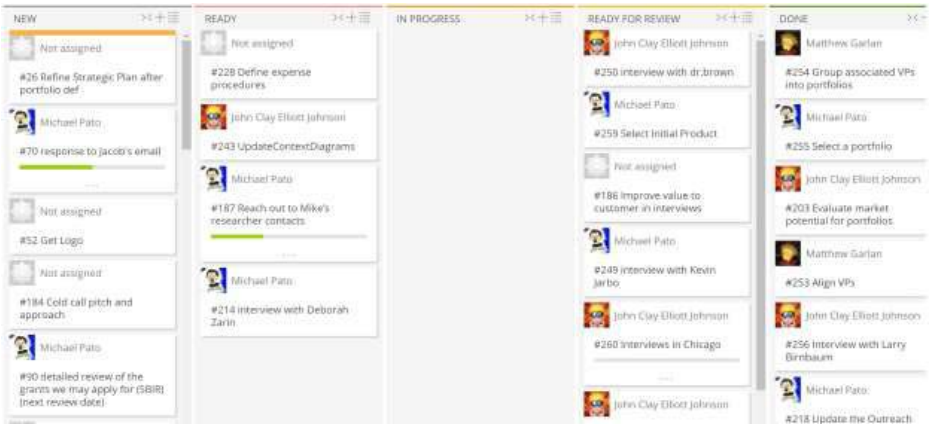

Figure 2: Example Kanban Board from Second Nature Software

The mechanics for operating a Lean team are also simple. Every morning there is a standup meeting where the team reviewed the board together and said what they planned to do. Team members can ask for help or offer it. Team members then continue work on the story they already "opened," or if they have no story then the team member selects a new story to begin. If there are enough stories waiting for Review (usually three to five) or a critical lesson learned, then the team sets up a Review meeting. During the Review meeting the team conducts story verification and planning to adjust, add, or delete New and Ready stories. This enables feedback to adjust the work and processes to become a more efficient team and meet the milestone as quickly as possible. Once it is determined that the milestone is complete, then a Milestone Review and Planning session begins the process again.

Phase 2 - Building the MVP with Disciplined Agile

Selecting a product provided the Second Nature team with a clear vision to execute against. The team began by exploring potential technology architectures that could speed up Mimimum Viable Product (MVP) development and fit the technical constraints of the medical research community. It was quickly identified that a small, standalone application would be best instead of developing a website or server-based tool. A standalone application is like Microsoft Word or Excel, where it runs on a user's personal computer (PC). Standalone tools are simple to build and add the benefit of inheriting security of the user's PC, which was the main technical constraint of medical researchers who work with personal identifying information. With a technology stack identified, Second Nature began investigating reference architectures and developing first round requirements. Tools built for cloud environments and used on large financial datasets, such as Trifacta and Paxata offered a service paradigm called "data preparation," which was exactly what the research community needed. Using these tools as reference, further prototyping and interviews of researchers helped to finalize technical designs and business requirements. Second Nature was now able to finalize its first set of features for the Alpha release of Rocketfish. The target early adopters were lab-based researchers who needed to process data collected using assays (test tubes arrays). The features included:

- **Link** - ability to merge datasets with appends (add records) and joins (add variables)
- **Derive** - ability to calculate new variables with existing variables and constants
- **Format** - ability to summarize, filter, and export datasets for analysis

With the direction set, the team leveraged Agile planning and mechanics to quickly deliver a working Alpha in less than two months, from product idea to product pilot.

Agile Planning and Mechanics

Agile planning and mechanics follow simple general practices that are found in every Agile approach. These practices adhere to the Agile Manifesto of emphasizes people, change, collaboration with customers, and delivering working software:

- **User Stories** - like with lean, every work feature component is described from the perspective of the primary user, what the user wants to do, and why they want to do it.
- **Timeboxes** - in Agile there is not task-level schedule, work is listed as a backlog

and performed within a set period of time, or “timebox.” Timeboxes enable Agile teams to innovate how to accomplish tasks while avoiding the stress of deadlines or the procrastination of start dates.

- **Colocated Teams** - Teams work colocated to ensure everyone can communicate efficiently face-to-face. This also improves team unity and opportunity for inspiration from working near people who are thinking and dealing with similar work challenges.
- **Whole Teams** - the team also works through the whole lifecycle of a work item together, including design, development, and testing. This ensures adequate breakdown of work and the whole team’s ownership of getting the work done.

The team used Disciplined Agile Delivery to design and implement new software products, so that Agile paradigm was used. Disciplined Agile Delivery recommends two stages to agile projects to ensure a shared vision and prioritized set of business needs before development begins. The goal is to exit the second stage with a Minimum Viable Product (MVP) ready for market:

1) *Solution Definition* - define the requirements of the MVP and explore technical solutions
2) *Solution Development* - teams select and implement the planned and designed MVP

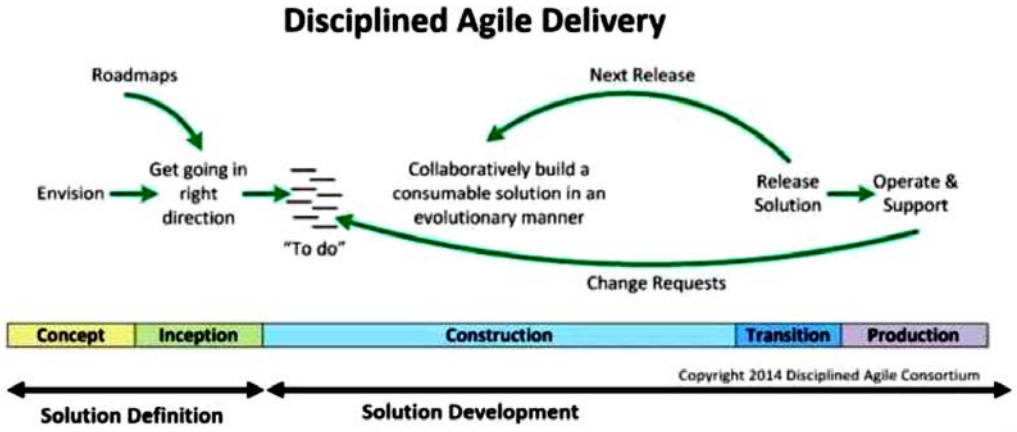

Figure 4: Disciplined Agile Delivery Stages [8]

Solution Definition

Solution Definition is a chance for the team to perform Agile planning for essential features. Agile planning starts by identifying a feature-driven breakdown of the business needs and alignment of those needs to a business process. Everything in Agile planning is based on the business, not the technical requirements (except when technology skills are lacking or solutions uncertain). The first release is determined to be the minimum set of features that will provide a “shippable product” that can be used by the business. This is called the Minimum Viable Product or “MVP.” Once the MVP is determined for the release, then stories are generated by whole teams of business and technology experts. These stories are then planned to be accomplished in fixed periods of time called “timeboxes” or “sprints” that are usually two to four weeks long. The stories are ordered based on testing dependency, risk, and priority. The resulting is a Story Plan forms a backlog of work that is testable by the team and users who support development. With the initial planning complete, the team is ready to progress from Solution Definition to Solution Delivery.

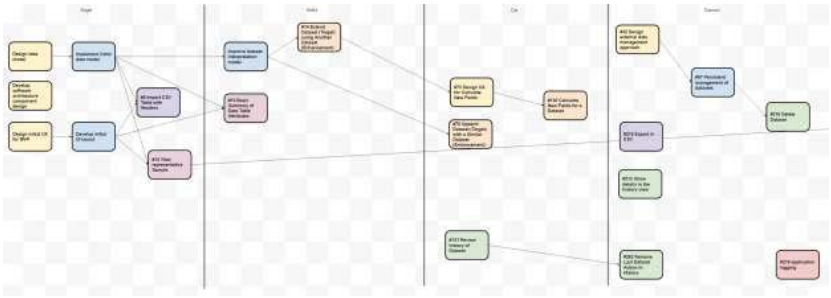

Figure 5: Sample of Rocketfish Alpha Story Plan

Solution Delivery

Solution Delivery begins with “Spring Planning” event where the whole team, product and customer team members, collaboratively selects stories for elaboration. Each story is first clarified in terms of the user, what the user wants to do in the app, and the business need satisfied by the story. The team then deconstructs these stories into tasks that include requirements, development, and testing of the solution. Once the story is fully understood, the team votes on the size of the story. This process continues until all stories are fully sized in detail. Once all stories are sized, the team re-evaluates and selects the stories for the sprint. The team then commits to accomplishing these stories within the sprint timebox (usually two to four weeks) with no plans to change the stories. After Sprint Planning the team begins working on the stories with each team member choosing the story and task to begin working on. Every day a standup meeting is held where team members say what they did, what they are planning to do, and if they need any support. Tasks are managed either as checklists or on a Kanban board; and when the final testing task is closed and approved by the whole team the story is complete.

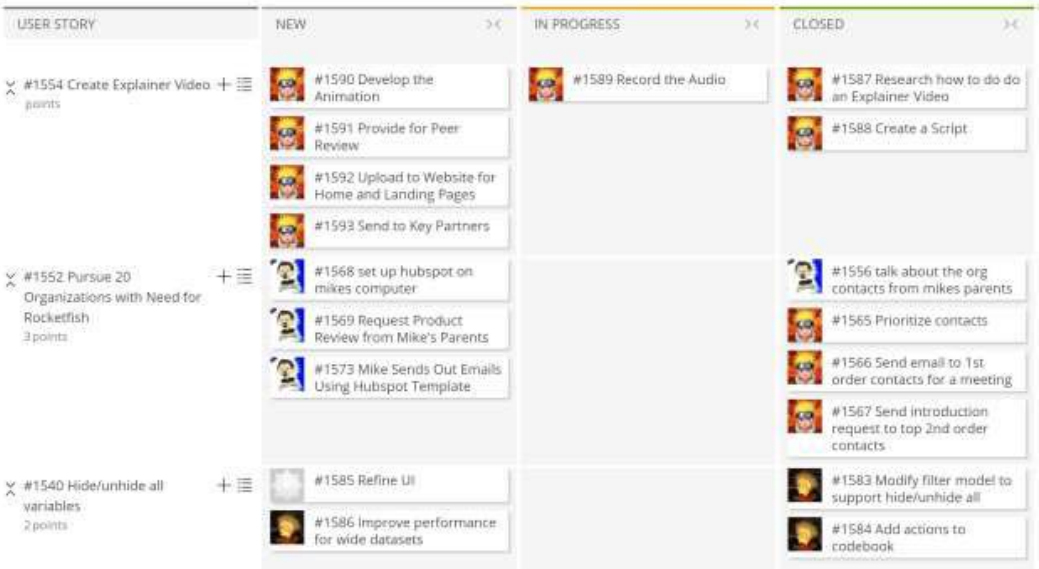

Figure 6: Sprint Execution Board

At the end of the sprint if the team was not able to complete the stories, any remaining stories are moved to the next sprint. The team then holds a Sprint Review to see what work was completed and provide feedback. Following the Sprint Review, is a Sprint Retrospective where the team discusses what went well, went poorly, and actions to be taken to make the next sprint better. This review process offers a means to continuously improve the team and hone in on a sustainable pace for development or “burn rate” (see Burndown Chart below in Figure 7).

Figure 7: Sprint Burndown Chart

The next Sprint then begins again with a Sprint Planning session and the cycle continues until the Release date. In each sprint the team re-evaluates stories, prioritizes the stories, and builds the features and enhancements determined by the whole team. At Release the team performs a Release retrospective, and then depending on the backlog will either go into another Release in Sprint Development or back to Solution Definition.

Phase 3 - Distributing Rocketfish with a Hybrid Model

Rocketfish was now in the hands of early evaluators, but the feedback was that the tool was incomplete with data cleaning and some better user interfaces for understanding the data. Meanwhile, the team at Second Nature needed a sale and feedback. Although one purchase order was made based on the Alpha, it was cancelled when the customer didn't receive the data they were going to use in Rocketfish.

The team decided it had enough information for two sprints using Agile and would then build both planned and requested features from current users in the following six sprints using a Lean/Agile approach for continuous releases every month. The result was a four-month release plan to generate the "Full Beta" with enough features for trials in NIH at the National Cancer Institute (NCI) and National Institute of Allergy and Infectious Diseases (NIAID).

Along the way, the team would learn that certain additional features were needed for distribution and managing trials. There was a need for licenses and license management for tracking users. A need for enabling persistence across not just tool sessions, but also version updates for continuous delivery.

Lean/Agile Hybrid Mechanics

Lean/Agile Hybrid planning sets up a Release plan, but lowers the pace of delivery for major feature development because of expected enhancements and defects. A Release Plan and Story Plan are needed to ensure that continuous improvement is made to the product to achieve the vision. However, that new feature scope must be limited or decomposed to enable the development team to respond to customer requests for improving the tool.

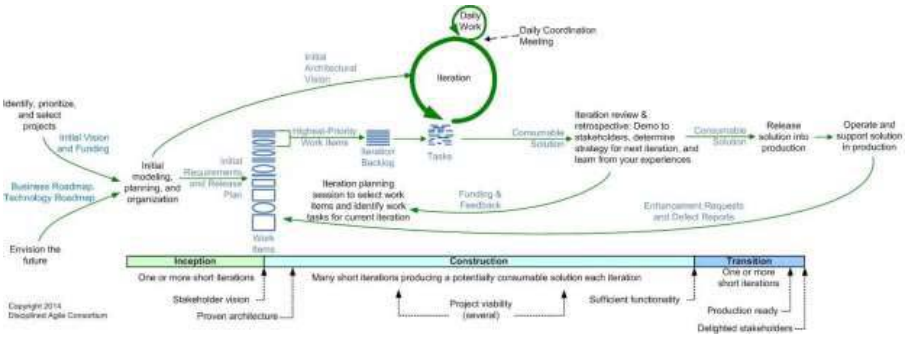

Figure 9: Lean/Agile Framework for Continuous Delivery [9]

Therefore, Release planning should therefore guide development, but not each sprint must prioritize the customer demands and feedback. In this way, the planning is much more Lean than Agile because it is driven by learning how to meet the needs dictated by customer feedback and still accomplish essential new product features efficiently and quickly. Agile processes are used to manage the day-to-day Mechanics, with set sprints that include Sprint Planning, Execution, and Review. This enables limits the planning churn impact on the team, and allows for managing the scope change in the Release Plan driven by customer feedback.

References

1. http://fortune.com/2014/09/25/why-startups-fail-according-to-their-founders/
2. https://www.forbes.com/sites/neilpatel/2015/01/16/90-of-startups-will-fail-heres-whatyou-need-to-know-about-the-10/#510637c96679
3. https://en.wikipedia.org/wiki/Lean_project_management
4. https://en.wikipedia.org/wiki/Agile_management
5. On Brightspace: Why the lean Startup changes everything, by Steve Blank