Lean or Agile:
Lessons Learned from a Tech Startup

By John Johnson, PMP
Co-Founder, Second Nature Software LLC
Chief Technology Officer, Softek Enterprises LLC

Graduate Masters of Systems Engineering,
Clark School of Engineering,
University of Maryland, College Park

Abstract: “Lean or Agile” provides a model of how any businesses can discover, develop, and deploy innovative solutions under extreme uncertainty. This paper evaluates project management techniques by using a technology startup case study to explore the extreme conditions which clarify when Lean and Agile management are most successful at achieving the organization’s goals. The paper presents insights on the indicators of when and where it is most appropriate to use Lean or Agile project management techniques, and how to transition between the two. The case study uses real-world examples of success and failure in applying these techniques to enrich its proposed theory on Lean and Agile. The paper’s subject is a startup, Second Nature Software LLC, that began with neither a product nor a target market and that, within one year, developed a cutting edge application piloted by five large research institutes, including two institutes at the National Institutes of Health (NIH).

Influences and project management theories referenced include: Lean Startup, Customer Development, Product Development, Design Thinking, Theory of Constraints, Scrum, Disciplined Agile Delivery, Kanban, and Total Quality Management.

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 NIH (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]. 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:

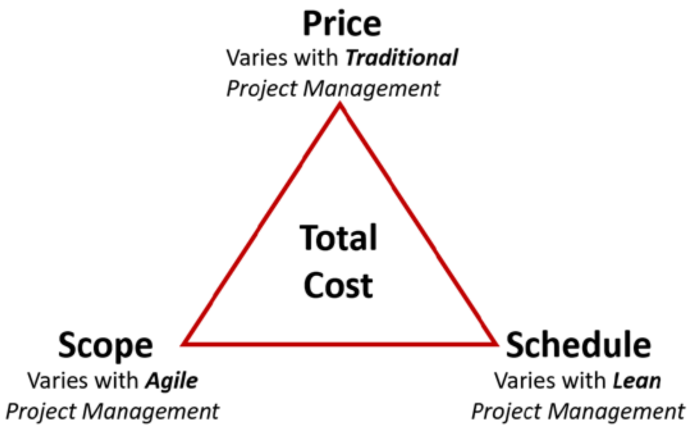

Price
Varies with *Traditional* Project Management
Total Cost
Scope
Varies with *Agile* Project Management
Schedule
Varies with *Lean* Project Management

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 of each other. Both methodologies “eliminate waste” and “vary output.” The history of two methods show a strong connection between Lean and Agile, which both have roots in Total Quality Management and its emphasis on continuous improvement.

The difference between Agile and Lean in a project management environment is the type of *uncertainty* they address.

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 [6] and the Opportunity Canvas by Dr. James Green [5]. Using these blueprints 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

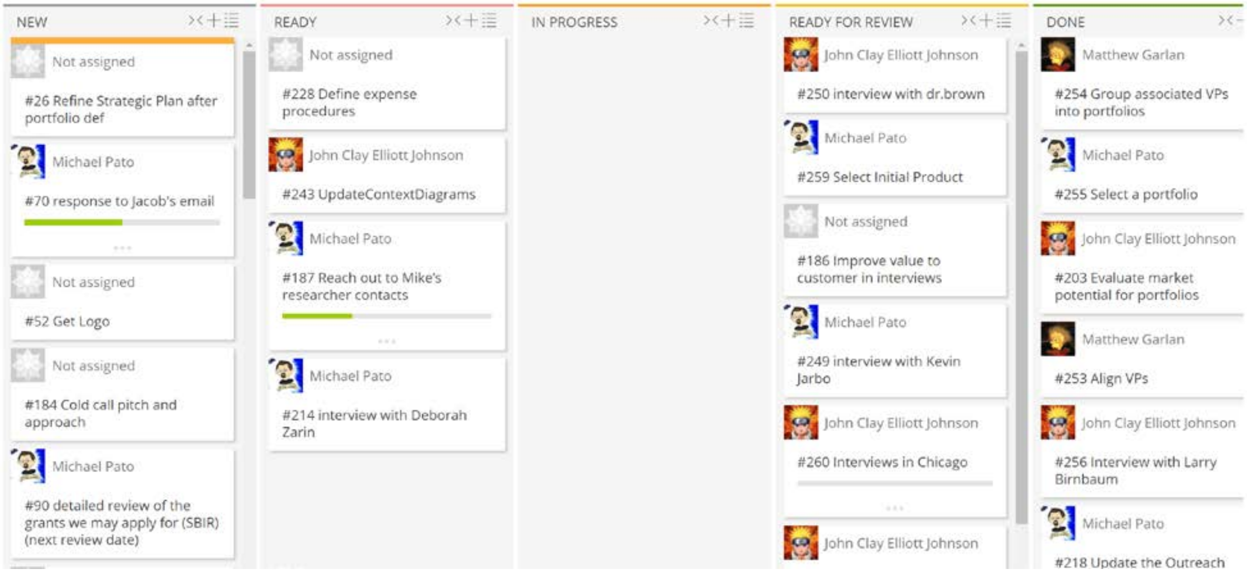
No one on the Second Nature team was truly qualified to execute any of these tasks as an expert. However, the team had systems engineering and management backgrounds, so they felt they could figure it out with “learning by doing.” At this point, the team decided to formally use Lean management to execute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


NEW

Not assigned
#26 Refine Strategic Plan after portfolio def

Michael Pato
#70 response to Jacob's email

Not assigned
#52 Get Logo

Not assigned
#184 Cold call pitch and approach

Michael Pato
#90 detailed review of the grants we may apply for (SBIR) (next review date)

READY

Not assigned
#228 Define expense procedures

John Clay Elliott Johnson
#243 UpdateContextDiagrams

Michael Pato
#187 Reach out to Mike's researcher contacts

Michael Pato
#214 interview with Deborah Zarin

IN PROGRESS

READY FOR REVIEW

John Clay Elliott Johnson
#250 interview with dr.brown

Michael Pato
#259 Select Initial Product

Not assigned
#186 Improve value to customer in interviews

Michael Pato
#249 interview with Kevin Jarbo

John Clay Elliott Johnson
#260 Interviews in Chicago

John Clay Elliott Johnson

DONE

Matthew Garlan
#254 Group associated VPs into portfolios

Michael Pato
#255 Select a portfolio

John Clay Elliott Johnson
#203 Evaluate market potential for portfolios

Matthew Garlan
#253 Align VPs

John Clay Elliott Johnson
#256 Interview with Larry Birnbaum

Michael Pato
#218 Update the Outreach

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.

Results of Phase 1

The Second Nature Software team quickly took stock of its strengths, including experience, skills, and social capital. These strengths were then aligned to markets using government NAICS codes and Census Bureau datasets. The final markets chosen as best alignments to the team were:

- **Medical Research** - highest on social capital and a large market at $70B annually
- **IT Consulting** - highest on skills and experience, but a very crowded market
- **Home Improvement** - highest on favorability to entrepreneurs with good skills alignment

The team ultimately decided to bank on social capital to open doors for interviews of decision makers. Here the team began to stumble as interviews initially took multiple weeks to set up, prepare for, and evaluate. The team also brought in stories of mapping business processes, archetypes, and customer needs to update and iterate in-between interviews. This saved down-time from waiting for appointments and responses from interviewees.

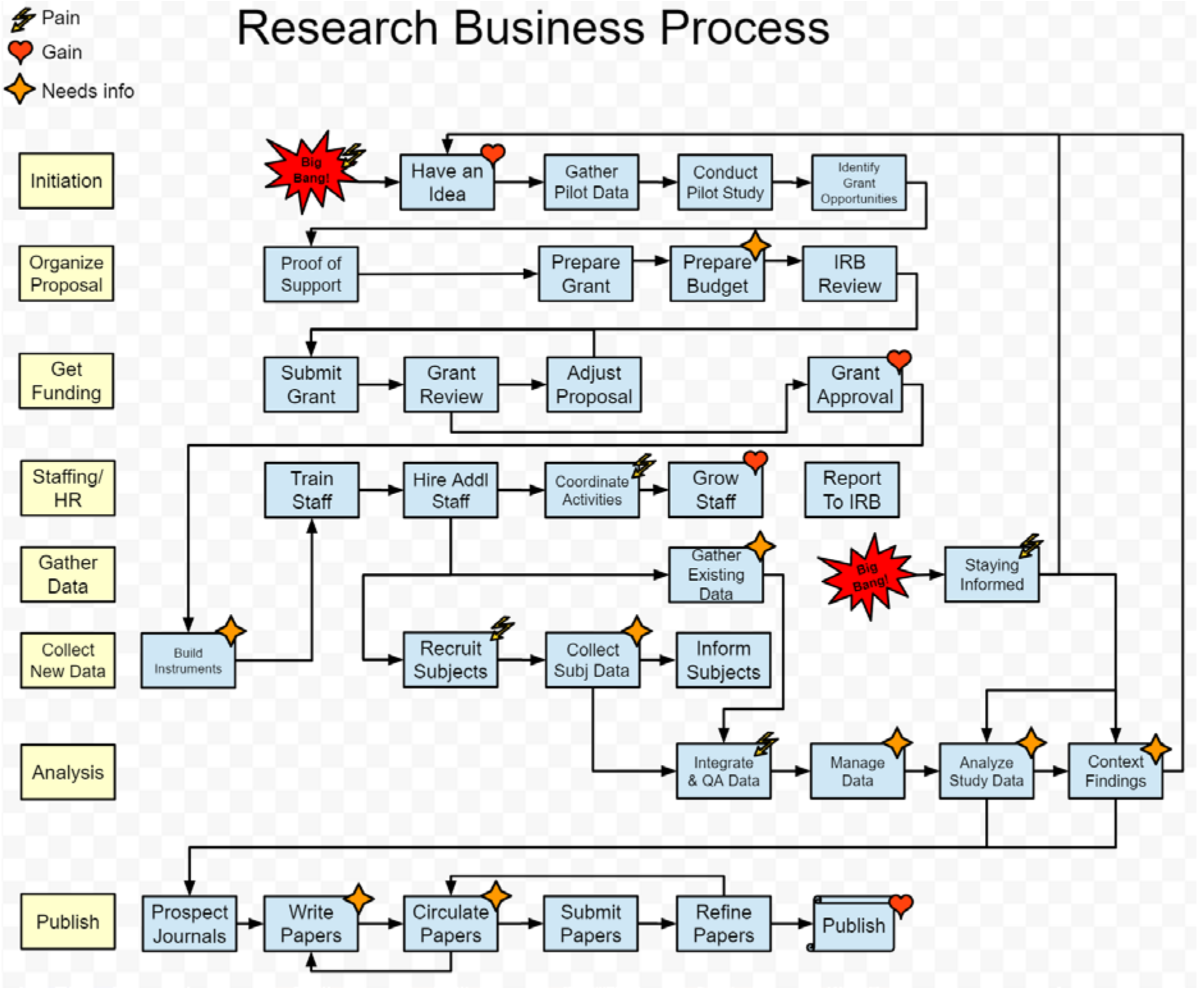


Figure 3: Business Process Map of Medical Research

Once interviews were complete, a quick decision making tool was developed to compare and prioritize customer needs. The top priority needs were then used as the basis for brainstorming product ideas that could fulfill one or more needs. The list was then systematically scored and reduced by the team to three potential products:

1) **Clinical Trials Recruitment App** - because 50% of clinical trials fail due to recruitment.
2) **Literature Review App** - staying informed is critical and time-intensive for researchers.
3) **Data Integration and Quality App** - researchers waste time working manually in Excel.

In the end, the team choose the Data integration and Quality Assurance App, which would become the tool now known as “Rocketfish.” This standalone tool would provide simple data processing, integration, and quality assurance to medical researchers who were spending large amounts of time doing manual work in spreadsheets.

Analysis of Phase 1

Phase 1 took a total of about four months to complete, and required much more work than anyone anticipated. About half the time was spent on interviewing decision makers, which turned out to be much longer than other startups usually take. Although the Second Nature processes improved for sourcing and conducting interviews, the methods were clearly flawed since on average the team performed at best 5 interviews per week and teams at Stanford in Steve Blank's startup class had interviewed almost five to ten times that number with similar resources [7]. However, the team largely attests the slow interviewing process to the nature of the medical researchers who are typically overworked and hard to schedule meetings with.

Lean management was extremely efficient for assigning and executing work, which enabled the team to quickly assign tasks to meet the objectives. Stories could and often were adjusted or updated based on lessons learned. The key was having a vigilant team always looking for ways to improve repeated processes, such as setting up and performing interviews. For example. standard emails for introductions and follow-ups reduced originally page-long emails to just a few key sentences. It was learned that most researchers only respond to the second or third email you send, so follow-ups increased success rates with getting interviews; and shorter emails with questions also got better reply rates. The interviews initially took an average of three to four weeks, but that time was cut in half by the last batch of interviewees.

This type of learning didn't require any time to pass between members because everyone was sharing lessons learned and observing each other's work daily. The transparency and all-hands-on-deck approach led to fast organizational learning and improvement in how the team worked through the planned scope.

Phase 1 Lessons Learned

Despite the learning, Phase 1 was likely twice as long as it needed to be because of the lack of experience on the team. The team lacked the skills or realistic expectations to know if progress was moving quickly or slowly. Many lessons learned were hard-earned instead of studied and incorporated from the beginning. The two major lessons learned were:

1) **Use Comparable Benchmarks** - leveraging previous, similar projects to understand what would be reasonable to expect in terms of pace and performance. When working in a Lean environment with uncertainty o\in how the scope can or should be performed, benchmarks offer a means of validating production efficiency.
2) **Seek Expert Advice** - If the team does not have expertise in the work, then the team will waste enormous time in Lean project environments. Lean offers a way to incrementally improve production, but teams without expertise will repeat many beginner mistakes that can be avoided with expert oversight. This is why many startups benefit most at early company stages by working in incubators and programs where experts offer advice from multiple startup successes.

Given this experience, it's recommended that only teams with significant expertise use lean management techniques; and in all cases Lean teams should seek outside expert consultation for benchmarks and validation of task performance.

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 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 ensure adequate breakdown of work and the whole team's ownership of getting the work done.

There are many types of Agile, but the team was well versed in using 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

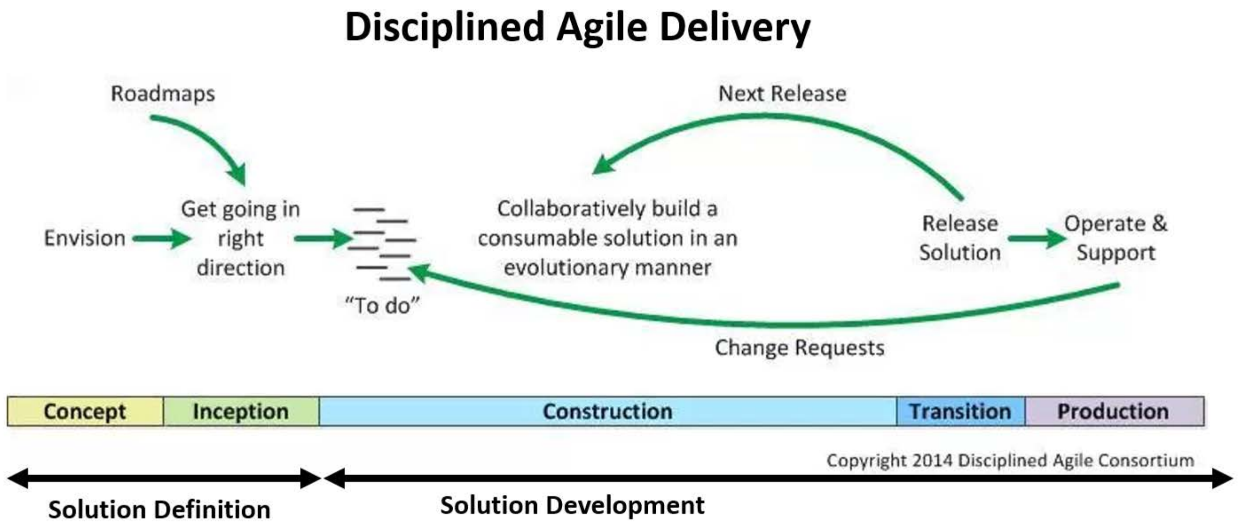


Figure 4: Disciplined Agile Delivery Stages [8]

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.

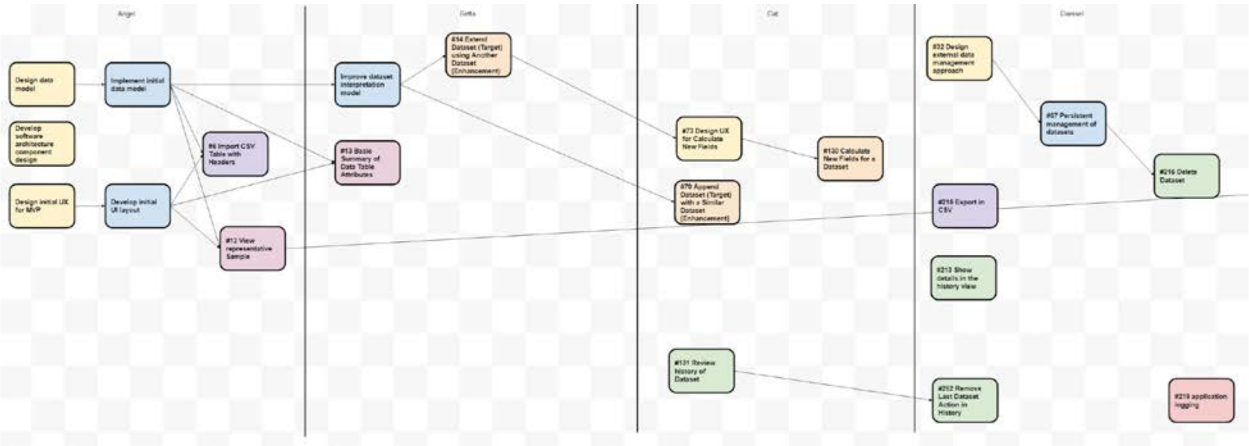

Angel
Design data model
Develop software architecture component design
Design initial UX for MVP
Implement initial data model
Develop initial UI layout
#6 Import CSV Table with Headers
#12 View representative Sample
Gotta
Improve dataset interpretation model
#14 Extend Dataset (Target) using Another Dataset (Enhancement)
#13 Basic Summary of Data Table Attributes
Cat
#73 Design UX for Calculate New Fields
#130 Calculate New Fields for a Dataset
#70 Append Dataset (Target) with a Similar Dataset (Enhancement)
#131 Review history of Dataset
Dataset
#32 Design external data management approach
#07 Persistent management of datasets
#216 Delete Dataset
#215 Export in CSV
#213 Show details in the history view
#212 Remove Last Dataset Action in History
#219 application logging

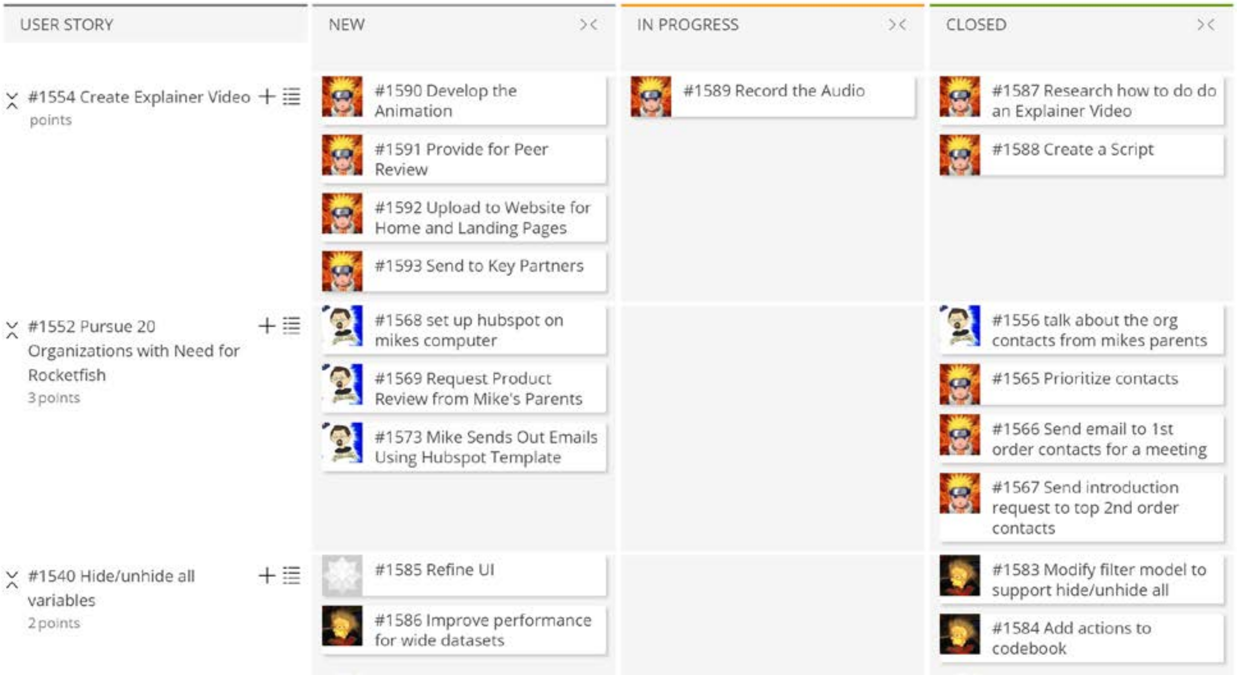
Figure 5: Sample of Rocketfish Alpha Story Plan

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.



USER STORY	NEW	IN PROGRESS	CLOSED
#1554 Create Explainer Video points	#1590 Develop the Animation #1591 Provide for Peer Review #1592 Upload to Website for Home and Landing Pages #1593 Send to Key Partners	#1589 Record the Audio	#1587 Research how to do do an Explainer Video #1588 Create a Script
#1552 Pursue 20 Organizations with Need for Rocketfish 3 points	#1568 set up hubspot on mikes computer #1569 Request Product Review from Mike's Parents #1573 Mike Sends Out Emails Using Hubspot Template		#1556 talk about the org contacts from mikes parents #1565 Prioritize contacts #1566 Send email to 1st order contacts for a meeting #1567 Send introduction request to top 2nd order contacts
#1540 Hide/unhide all variables 2 points	#1585 Refine UI #1586 Improve performance for wide datasets		#1583 Modify filter model to support hide/unhide all #1584 Add actions to codebook

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).

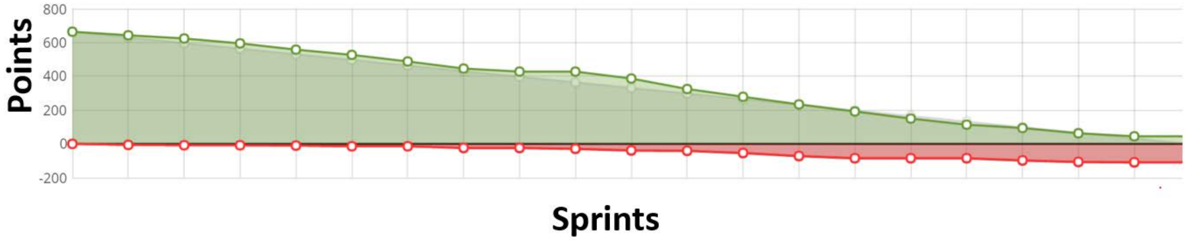


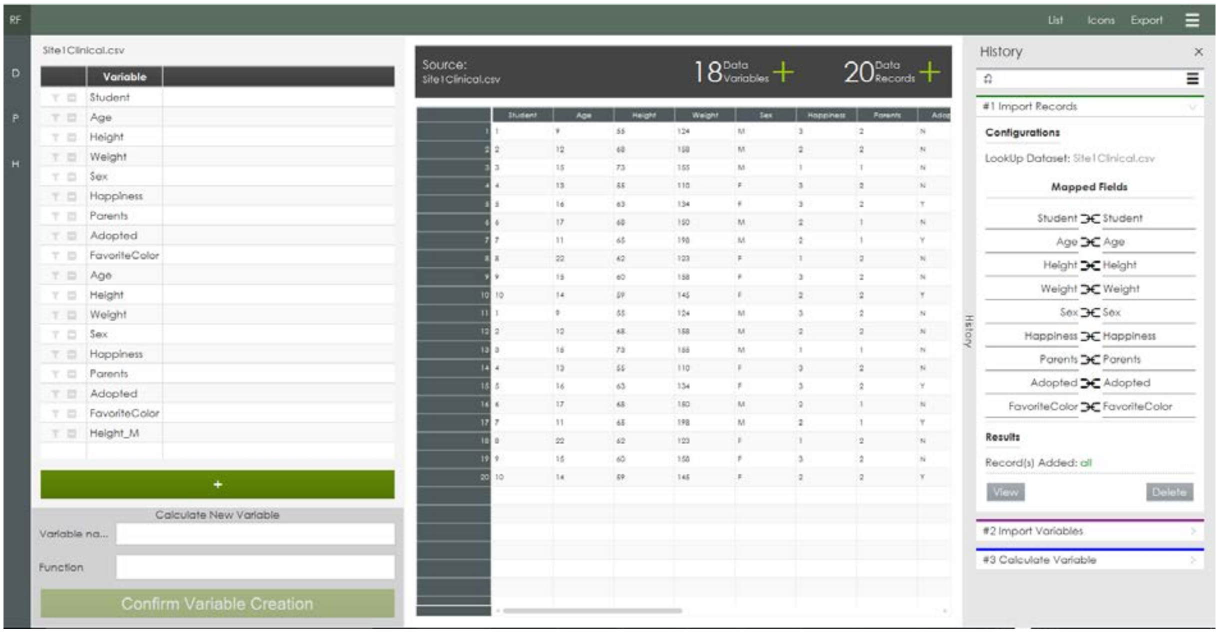
Figure 7: Sprint Burndown Chart

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.

Results of Phase 2

The Second Nature team finished the Alpha as designed on-time and with additional features that added data management and process history for each data set. The final product was release to 15 early evaluators of the tool that offered feedback on its design and use. The first release was such a success that one evaluator used it to complete her work that normally took days in just a few hours.

The Alpha Release was not without its hiccups. The user experience was generally overlooked in terms of installation, which required users to download the latest version of Java along with the tool. This flaw added barriers to getting started with Rocketfish and prevented some evaluators from being able to use to tool all together. However, in general reviews were very positive.



Site1Clinical.csv

	Variable
	Student
	Age
	Height
	Weight
	Sex
	Happiness
	Parents
	Adopted
	FavoriteColor
	Age
	Height
	Weight
	Sex
	Happiness
	Parents
	Adopted
	FavoriteColor
	Height_M

Variable na...

Function

Source: Site1Clinical.csv

18 Data Variables

20 Data Records

	Student	Age	Height	Weight	Sex	Happiness	Parents	Adop
1	1	9	55	124	M	3	2	N
2	2	12	68	158	M	2	2	N
3	3	15	73	165	M	1	1	N
4	4	13	55	110	F	3	2	N
5	5	16	63	134	F	3	2	Y
6	6	17	68	150	M	2	1	N
7	7	11	65	198	M	2	1	Y
8	8	22	42	123	F	1	2	N
9	9	15	60	158	F	3	2	N
10	10	14	59	145	F	2	2	Y
11	1	9	55	124	M	3	2	N
12	2	12	68	158	M	2	2	N
13	3	16	73	165	M	1	1	N
14	4	13	55	110	F	3	2	N
15	5	16	63	134	F	3	2	Y
16	6	17	68	160	M	2	1	N
17	7	11	65	198	M	2	1	Y
18	8	22	62	123	F	1	2	N
19	9	15	60	158	F	3	2	N
20	10	14	59	145	F	2	2	Y

#1 Import Records

Configurations

LookUp Dataset: Site1Clinical.csv

Mapped Fields

Student	Student
Age	Age
Height	Height
Weight	Weight
Sex	Sex
Happiness	Happiness
Parents	Parents
Adopted	Adopted
FavoriteColor	FavoriteColor

Results

Record(s) Added: all

#2 Import Variables

#3 Calculate Variable

Figure 8: Rocketfish Alpha Screenshot

A major oversight for adoption was requiring data to be pre-formatted and cleaned before being used in the tool. It was expected that users would clear their data first in spreadsheets; but most early users did not follow instructions and quickly gave up because using two tools for cleaning and integration was too cumbersome. Data cleaning had been delayed in being added to the tool, and the result was that the Alpha was burdensome to prepare for as a data prep tool.

Lastly, to ensure on-time delivery the level of innovation in the design was limited, especially on the aesthetics and user experience. This was because achieving functionality on-time was prioritized over delighting the customers with a visually appealing and high-quality experience. The final design felt like what some called an “Excel-Light” user interface, which hurt branding the tool as something truly transformational.

Analysis of Phase 2

Phase 2 was efficient and productive, with a working product that truly added value to its users. There were many issues to be overcome before a true Beta could be released and sold, but the primary functionality was built and in the hands of users. The feedback was very positive and delivered recommendations from the decision makers in the National Institutes of Health (NIH) for Rocketfish to piloted institute-wide at both the National Cancer Institute (NCI) and National Institute of Allergy and Infectious Diseases (NIAID).

There were some issues with implementing Agile according to best practices. Although the key planning documents were developed and the researcher process understood, the team chose poorly in defining the MVP. Cleaning was an essential first step that was needed much more than integration or calculation (as the team would find out), and came first in the user's workflow. Without cleaning within the tool, the burden to the user was too great to get started. This could have been avoided by features into the tool in the order the features would be used in the researcher's workflow.

The detriment of using Agile was a miss on aesthetics and user experience because the push to deliver functionality within the timebox de-prioritizes polishing features. In some cases, this is not just acceptable, but preferable. However, when selling new data tools there was clearly a need for ease of use and tool attractiveness to gain first-time users that believed Rocketfish could be transformational.

Phase 2 Lessons Learned

Agile is focused on delivering features fast, and while that can be a great strength, it's easy to get caught up in what is known in startups as "featuritis." This occurs when features and capabilities overcome the true customer needs and selling potential of the tool. Users must be considered not just as part of a business process, but as psychologically-driven and incentivized people who must *want* to use the tool. Therefore, the key lessons learned were:

1) **Emphasize Front-to-Back Design** - start with the user experience in mind and adapt technology to achieve it. There should be no contempt for users, only a focus on delighting them.
2) **Build Features in Workflow Order** - do not build the most important feature first. Instead build the first feature a customer will use first. Know the true bounds of the user's workflow and inject the tool as far upstream as possible to maximize adoption.

Because speed was the focus, the team avoided hard features like data cleaning and building powerful user interfaces. Just data cleaning could have arguably been its own product, but in a rush to ensure delivery in the timebox the team chose a safer, less transformation feature set for the MVP. Agile Teams need to set bold feature goals that will be truly transformational to offset a natural tendency to play it safe with delivery inside the sprint timebox.

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 canceled 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.

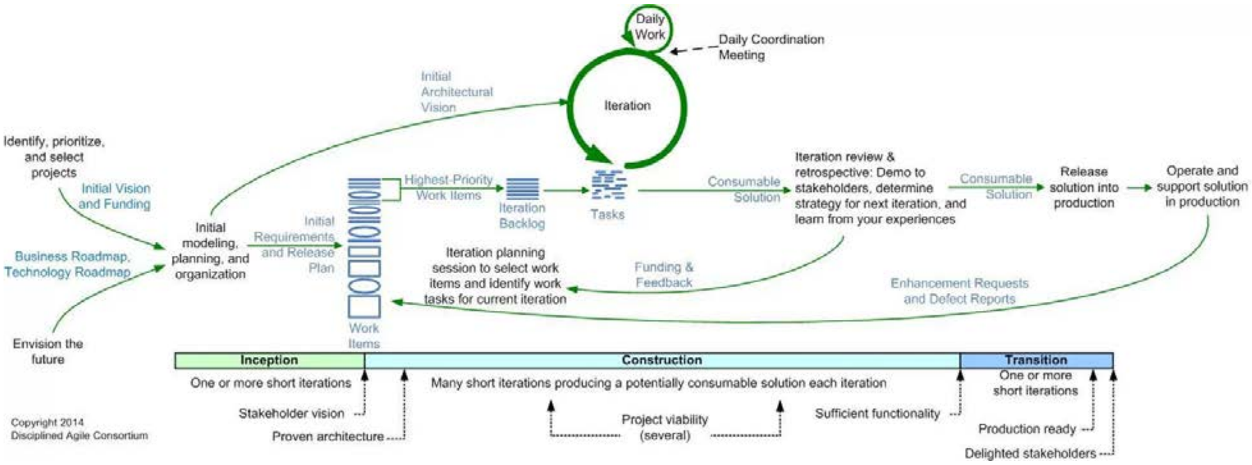

Daily Work
Daily Coordination Meeting
Initial Architectural Vision
Iteration
Identify, prioritize, and select projects
Initial Vision and Funding
Business Roadmap, Technology Roadmap
Envision the future
Initial modeling, planning, and organization
Initial Requirements and Release Plan
Highest-Priority Work Items
Iteration Backlog
Tasks
Work Items
Consumable Solution
Iteration review & retrospective: Demo to stakeholders, determine strategy for next iteration, and learn from your experiences
Consumable Solution
Release solution into production
Operate and support solution in production
Iteration planning session to select work items and identify work tasks for current iteration
Funding & Feedback
Enhancement Requests and Defect Reports
Inception
Construction
Transition
One or more short iterations
Many short iterations producing a potentially consumable solution each iteration
One or more short iterations
Stakeholder vision
Proven architecture
Project viability (several)
Sufficient functionality
Production ready
Delighted stakeholders
Copyright 2014 Disciplined Agile Consortium

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.

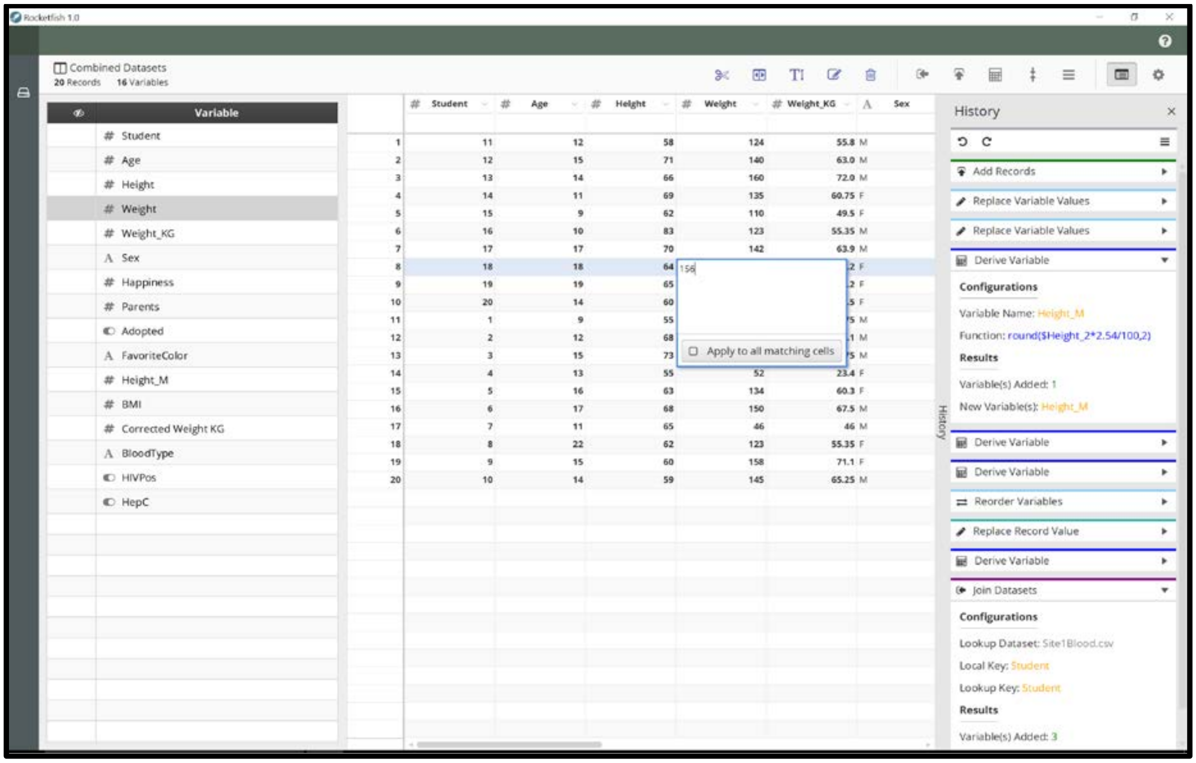
Agile processes are used to manage the day-to-day Mechanics, with set sprints that include Sprint Planning, Execution, and Review. This enables limits the planning churn impact on the team, and allows for managing the scope change in the Release Plan driven by customer feedback.

Results of Phase 3

Phase 3 was a partial success, with the Product Development team building a tool that had all the features needed for a trial, but not actually achieving the features planned in the Release Plan for Rocketfish Beta.

The final product was (and is) a very powerful data preparation (management and processing) tool for researchers with all the essential features included:

- **Data Management** - saving, copying, deleting, and exporting data files.
- **Dynamic Data History** - tracking and rollback for all data processing actions.
- **Data Cleaning** - variable and value cleaning, as well as row and whitespace removal.
- **Data Formatting** - one-touch splitting, filtering, and capitalization of values
- **Data Integration** - enhanced data merging with auto-matching for linking variables.
- **Advanced Calculation** - text, date, number, and logical expression evaluations
- **Aggregations** - roll-up of record-level information to summary-level observations.



Combined Datasets
20 Records 16 Variables

Variable
Student
Age
Height
Weight
Weight_KG
A Sex
Happiness
Parents
Adopted
A FavoriteColor
Height_M
BMI
Corrected Weight KG
A BloodType
HIVPos
HepC

	# Student	# Age	# Height	# Weight	# Weight_KG	A Sex
1	11	12	58	124	55.8	M
2	12	15	71	140	63.0	M
3	13	14	66	160	72.0	M
4	14	11	69	135	60.75	F
5	15	9	62	110	49.5	F
6	16	10	83	123	55.35	M
7	17	17	70	142	63.9	M
8	18	18	64	156	.2	F
9	19	19	65		.2	F
10	20	14	60		.5	F
11	1	9	55		5	M
12	2	12	68		.1	M
13	3	15	73		5	M
14	4	13	55	52	23.4	F
15	5	16	63	134	60.3	F
16	6	17	68	150	67.5	M
17	7	11	65	46	46	M
18	8	22	62	123	55.35	F
19	9	15	60	158	71.1	F
20	10	14	59	145	65.25	M

□ Apply to all matching cells

History

- Add Records
- Replace Variable Values
- Replace Variable Values
- Derive Variable
  - Configurations
  - Variable Name: Height_M
  - Function: round($Height_2*2.54/100,2)
  - Results
  - Variable(s) Added: 1
  - New Variable(s): Height_M
- Derive Variable
- Derive Variable
- Reorder Variables
- Replace Record Value
- Derive Variable
- Join Datasets
  - Configurations
  - Lookup Dataset: Site1Blood.csv
  - Local Key: Student
  - Lookup Key: Student
  - Results
  - Variable(s) Added: 3

Figure 10: Rocketfish version 1.0 Screenshot

Although not all essential features or differentiating features were built, many organizations showed interest in Rocketfish. The list of organizations in or starting trials as of the end of the Phase 3 included:

1. National Cancer Institute (NCI)

2. National Institute of Allergy and Infectious Diseases (NIAID)
3. National Center for Advanced Translational Sciences (NCATS)
4. Johns Hopkins University, Bloomberg School of Public Health
5. University of Maryland, School of Medicine, Institute of Human Virology
6. Rutgers University, Division of Infectious Disease

This constituted a true success for the first year of development. A fully functioning product in pilot with the top research organizations in the country. However, the company still did not have a sale and that meant that the team had missed somewhere on requirements, development, and especially distribution.

Analysis of Phase 3

The Lean/Agile methods of Phase 3 were hard to balance with competing demands from customers for must-have-now features and the planned features to achieve the assessed market needs. The most critical features planned for Phase 3 that involved innovating new technologies were designed, but never built. The feature, an advanced cleaning capability using visuals, was delayed as mounting feedback came in demanding modifications for specific customer needs.

For example, the University of Maryland needed better calculation features, which would also enable more cleaning and validation. This feature, along with others, was implemented instead of the planned data cleaning using visuals. So, the tool met the needs of the customers but lacked a truly differentiating component.

Another phenomenon occurred, which is that the tool began to reach its limits as a standalone tool built with open-source Java components. This meant that the team began to hesitate to put in features because the team knew the features would be rebuilt once Rocketfish inevitably migrated to new JAVA software components. This is typical in Agile development, but was also likely exacerbated by the many aesthetic updates since most of the trouble was with the Javafx components used for the user interface.

Lastly, with the hope that the tool would sell quickly the team began to plan less and less for transformational capabilities. Pushing the tool forward technically was no longer priority. This led to almost no new planning for the second half of Phase 3 when it came to Story Plans and Release Plan modifications. It was clear at the end of the Phase 3 that with better planning, more features of higher importance would have been completed.

Lessons Learned from Phase 3

There was a general lack of new development because of trying to integrate Lean planning and Agile execution. Lean Planning resulted in many features being caught up in design because the team was uncertain what the total scope would be for the Release. This meant there wasn’t enough time in a timebox to complete complex stories and still respond to defects and enhancements. The Second Nature team didn’t want to risk building features without enough space in the Sprint due to growing tool instability which would create more rework for later, or “technical debt.” The result was a lack of true progress towards building features that could differentiate Rocketfish from its competitor tools.

Without a stable plan for feature development and clear deadlines for feature completion, the Lean/Agile approach falls into incrementalism. Simply building one small improvement at a time without vision or confidence to risk.

1) **Maintain a Story & Release Plan** - for Lean/Agile to work effectively, there must be continuous master planning (Release Plan) and mid-range planning (Story Plan). Without it, the team will lose direction and motivation.
2) **Plan a Hardening Sprint** - Hardening Sprints are used to catch up on and clean up technical debt. Knowing the Hardening Sprint is coming will ensure the team can confidently build complex stories in packed Lean/Agile sprints.
3) **Build Differentiating Features First** - as a startup, one cannot just “meet the requirements,” one must also sell the tool while building it. The delay of building the differentiating capabilities, or “star feature,” of the tool hurt sales although it made development easier. Therefore, plan to build the most important features early when defects and enhancements requests are low.

Conclusion

For Startups, the recommendation is to use Agile for speed and adaptability as the *team learns what the business needs truly are*; and to only go with Lean or Lean/Agile hybrid approaches for sustainment and improvement of a working, selling product. Even at the beginning of Second Nature’s journey, when the team was following roadmaps to discovery a product, Agile would have been more effective. The methods would have been questioned, interviews sped up, evaluation methods questioned, and a lot of wasted effort (scope) avoided. Without significant expertise on the team, operating as “Lean Startup” is recipe for incrementalism and scope creep that consumes resources while adding frustration.

If the team has little to no control or concern over what scope must be accomplished, and if the goal is to deliver as fast and efficiently as possible, then Lean is more effective. Lean is especially appropriate when the team is highly experienced with the technical work that is being performed. In many scaled Agile frameworks the architects respond in a Lean process to the Agile teams -- subordinated to the scope determined by those Agile teams. This works for Lean teams because instead of focusing on what to do, the team can learn how to predict, build, and test designs that will meet the needs of the customer(s) as efficiently as possible. Lean is the most responsive to customer demands, if the customer knows what they want.

However, if there is a possibility the business needs will change or should change, either by the ability to target customers or include the customer in the development process, then Agile is far faster and more stable. The timeboxing and planning efforts are clearer and provide a consumable vision for developers and business analysts to focus on. The focus and planning makes Agile much more effective than Lean at managing changing or uncertain business needs.

The question of “Lean or Agile?” in theory comes down to a question of whether the area of uncertainty is in the *how* to deliver the technology or *what* technology to deliver. But based on the real-world experiences of using Lean, Agile, and Hybrid methods - Lean and Hybrid models lack the dedicated time and process to address *why* the technology is even needed. This doesn’t mean the team stops asking the question -- it just stops answering it as a team. Agile provides this time for team planning with Release Plans, Story Plans, and Sprint Planning days. All that planning challenges, shapes, and reinforces a shared vision. Lean methods lack this time for the team to think together as a team, and, therefore, is not conducive for rough and tumble challenges of an uncertain project, like a startup company.

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-what-you-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. https://s3.amazonaws.com/accredible_card_attachments/attachments/54661/original/TheOpportunityAnalysisCanvaspdf.pdf
6. https://web.stanford.edu/group/e145/cgi-bin/winter/drupal/upload/handouts/Four_Steps.pdf
7. https://www.youtube.com/watch?v=6t0t-CXPpyM
8. http://www.disciplinedagiledelivery.com/lifecycle/
9. https://i2.wp.com/www.disciplinedagiledelivery.com/wp-content/uploads/2014/05/disciplined-agile-lifecycle-basic1.jpg
10. http://www.leanproduction.com/theory-of-constraints.html
11. https://en.wikipedia.org/wiki/Toyota_Production_System
12. http://www.industryweek.com/companies-amp-executives/theory-constraints-tapped-accelerate-bps-gulf-mexico-cleanup
13. http://www.manufacturingglobal.com/top10/38/Top-10:-Lean-manufacturing-companies-in-the-world
14. http://www.forwardfocusinc.com/jumpstart-change/the-importance-of-empowering-employees/
15. https://www.bloomberg.com/news/articles/2013-09-19/forget-employee-engagement-u-dot-s-dot-companies-need-passionate-workers
16. http://www.skymark.com/resources/leaders/deming.asp
17. https://en.wikipedia.org/wiki/W._Edwards_Deming
18. http://agilemanifesto.org/