Ultimate Kanban: Scaling Agile Without Frameworks at Ultimate Software

PRATEEK SINGH, Ultimate Software
DANIEL S. VACANTI, ActionableAgile™

How did Ultimate Software, a fast-growing, Fortune 500 company, scale its Agile implementation to over 25 teams, reduce its Cycle Time across the organization, increase Throughput per team, and gain better overall predictability—all without the use of any scaling framework? This paper will outline the steps that Ultimate Software took to solve the dual threats to Agile of both growth and scale, it will present the data that demonstrates its success, and it will provide some details around how Ultimate Software plans to continuously improve. This paper's main intention is to show that an Agile implementation, based on Kanban principles, can be scaled across a large organization without the use of heavyweight, complex frameworks.

1. INTRODUCTION

Ultimate Software (Ultimate) is a leading provider of Global HR and Payroll software. The company has been ranked on "Fortune's 25 Best Companies to Work For" list for the past 5 years and was named "#1 Best Company to Work For in Tech" for 2016. The development organization at Ultimate is made up of 900 people spread across 25 teams—all 25 of which follow Agile principles. Agile practices are a necessity at Ultimate for two reasons:
1. A hyper competitive marketplace
2. The need to immediately react to the enactment of federal, state and local laws concerning payroll, taxation and human resources (e.g., The Affordable Care Act)

After several failed attempts to do Scrum at scale, Ultimate finally settled on Kanban as its scaled methodology of choice. The results of this move, as you are about to see, speak for themselves

2. BACKGROUND

Ultimate started experimenting with Agile principles (namely, Scrum) in 2005. This initial transition to Scrum provided Ultimate with better visibility into the progress of teams towards wider business goals. However, there were some common sources of interruption that the Scrum did not handle very well. Regulatory changes that required immediate attention often forced teams to throw out plans for their sprints and start work for the new requirements. The ideal small Scrum team size (7-9 members) led to arbitrarily small teams with a very high cross-team coordination costs. Most importantly, though, after trying our hand at Scrum for a while, we did not see any major improvement in productivity.

After struggling with Scrum for a while, Ultimate attempted a "reboot" by retraining the leads of the teams in Scrum principles. This time, however, the training was supplemented with practices from Lean and XP. To our dismay, this reboot still did not provide the hyper-productivity that the company was seeking.

At this point, the leadership in Product Development started to experiment with Kanban. As our first shot with Kanban, we selected the infrastructure team as that team was particularly problematic. Without explicitly changing methodologies from Scrum to Kanban, we overlaid the Kanban principles on top of the current Scrum process. Limiting WIP and visualizing work had an almost immediate effect on the team. The team was able to keep up with the tickets that were coming their way and were able to stay on track. Given our success, we

repeated this approach with another core Product Development team. This again, turned out to be a success. It wasn't long before all teams were moved to a Kanban system.

The impact of this move was felt immediately. As Kanban has no explicit limit on team size, it allowed us to collapse business lines into large teams to allow for lower transaction costs. The adoption of Kanban also marked the first time that we started to see the hyper-productivity that Agile promised. Kanban principles and flow metrics helped the teams begin to achieve the productivity results that we had been looking for since the start of our Agile journey. These results are discussed in detail in Section 3.

2.1   A Brief Word on Flow Metrics and Analytics

Before we go much further, we need to pause for a quick word of caution. The following section is going to detail improvements at both the team and organization levels after our move to Kanban. These improvements are going to be stated in terms of basic flow metrics: Work In Progress (WIP), Cycle Time, and Throughput. Further, these metrics are going to be visualized in basic flow analytics like Cycle Time Scatterplots and Throughput Histograms. A basic comprehension of these metrics and analytics is essential to understanding the success of the Kanban implementation at Ultimate. If you are not familiar with these concepts, then we highly recommend reading the book "Actionable Agile Metrics for Predictability" by Daniel Vacanti (see references below). This book represents the only comprehensive treatment of flow metrics and analytics currently on the market.

3.   RESULTS WITH KANBAN

It was late 2014 when we renewed our focus on Kanban. We retrained every team in the organization on the importance of Kanban principles in combination with flow metrics. We focused on the benefits of explicitly mapping out and visualizing process, limiting WIP and managing for flow. The results were far better than we expected and are detailed in the individual team cases outlined below.

3.1   THE ACES TEAM

- Team responsible for greenfield development of new Pay Calculation Engine
- 60% reduction in Cycle Time for stories from 35 days @ the 85th percentile (using Scrum) to 14 days @ the 85th percentile (using Kanban)
- Lost half the team yet recorded a 10% increase in story Throughput from 170 completed in five months to 186 stories completed in five months

The ACES team (ACES) started in 2013 as a 16-member Scrum team whose sole responsibility was the development of a new Pay Calculation Engine. In early days of Scrum, the team was widely considered successful because it delivered a consistent velocity (from a Scrum perspective). Upon examining the team's data in the light of flow metrics, however, we discovered that there were extreme inefficiencies in its development process. When remedied, the gained efficiencies resulted in higher productivity and greater predictability.

One of the best charts to demonstrate the predictability of a team is a Cycle Time Scatterplot (please note that for the rest of this paper, whenever we say the word "Scatterplot" what we mean is "Cycle Time Scatterplot"). The Scatterplots below contrast the performance of the team before and after adjusting their processes based on Kanban principles:
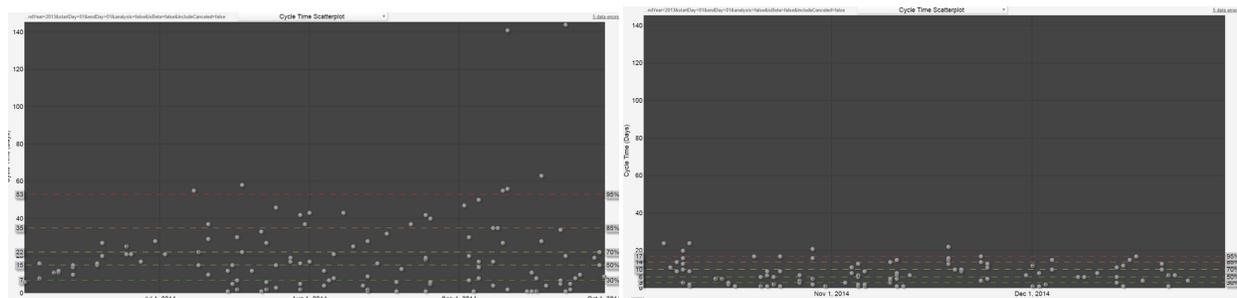

Figure 1: ACES Team Cycle Time Scatterplot Before Kanban (left) and After Kanban (right)

The left chart in Figure 1 above shows that when the team was using Scrum practices, 85% of its stories took 30 days or less to complete. A 35-day Cycle Time in and of itself is not necessarily bad unless you put it in the context of the fact that the team was running 14-day sprints. Further, 50% of the stories completed in that same time frame took 15 days or less to complete. What that means is that stories that started at the beginning of a sprint only had about a 50% chance of completing within that same sprint. This is not the picture of predictability that the Scrum velocity metrics would lead us to believe.

After taking note of the Scatterplot, the team began to dive into the reasons why stories were taking so long to complete. What they discovered was that most long-lived stories were sitting in the "Ready for QA" column for extended periods of time. That was a problem because "Ready for QA" is a queuing column where stories just sit and are not actively worked on. These "waiting" columns are the low hanging fruit of process improvement and so it was "Ready for QA" that the team decided to attack first by putting a WIP limit of 5 on that column. The team also chose to prioritize concentrating on work that has been in progress the longest to achieve a consistent rate of finishing work rather than allowing stories to age indefinitely.

The results of these policy changes were almost immediate. From that point forward (see the right chart in Figure 2) the team was able to get 85% of their stories done in 14 days or less. Throughput for ACES also increased from 1.07 stories per day to 1.41 stories per day. This was achieved in the same time period when the team size was reduced to half of the original size. It should be pointed out here these modifications did not include changing the size of stories or working overtime. The team continued to hone their flow-focused process by further lowering the WIP limit on the Ready for QA column and encouraging the various disciplines on the team to help each other out in order to make sure none of the items on the board age beyond their 85th percentile.

The improvements achieved by the ACES team continue to this day.

3.2   Payroll Team

- Responsible for core payroll functionality in a context characterized by frequent interrupts of urgent customer requests
- 79% reduction in average queuing time for stories from 8.84 days to 1.88 days
- 69% reduction in story Cycle Time from 36 days @ the 85th percentile to 11 days @ the 85th percentile

The Payroll team maintains and develops the core payroll capabilities for Ultimate Software's flagship product, Ultipro. The current incarnation of this 30-person team was formed in 2009 by combining three separate smaller Scrum teams. It should be noted that this team had a consistent history of being interrupted by urgent customer issues (think about how upset you might be if your paycheck was not calculated correctly or distributed on time).

After Kanban re-training in 2015, the Payroll team immediately changed the way they worked. One dramatic change they made was to lower their WIP limits on their board below the total number of people on the team. The idea behind this change was to promote pairing and remove knowledge silos. This also left slack in the system to allow for the team to deal with emergency customer issues when they came up. The result of embracing these policies was immediately visible in how long it was taking the team to complete their stories. The amount of time stories spent in queuing states decreased dramatically over time and as a result the Cycle Times for the team went through a dramatic decrease as well. The table below shows the faster Cycle Times since the team's training at the end of March 2015.

MONTH	AVERAGE TOTAL CYCLE TIME (days)	AVERAGE QUEUING TIME (days)	TOTAL CYCLE TIME 85th PERCENTILE (days)
January	18.49	8.84	36
February	20.72	11.41	34
March	12.77	5.74	27
April	4.24	0.68	7
May	9.21	1.91	18
June	7.83	1.91	13
July	6.94	2.23	12

August	6.39	2.63	11
September	5.05	1.13	10
October	7.23	1.56	12
November	8.08	1.88	11

Figure 2: Payroll Cycle Times vs. Queuing Times

Notice from Figure 2 that the net active time that the team was spending on stories did not change. By limiting their WIP, the team was able to cut down the time the story was just sitting on the board. As the team got more efficient in the use of a Kanban system and started tweaking their process policies, they were able to gain greater consistency in story completion times (at the 85th percentile—again, see Figure 2).

The greater predictability of Cycle Times had two immediate effects. First, the team was able to deliver value to the customers faster and more regularly. Second, when an emergency issue did come up, the team could ask the question of "Can this wait until we finish one of the items we are currently working on?". As work items were getting done faster, there was a regular stream of people freeing up to pick up the next item. With a team member freeing up on a daily basis, the team could ask the requesting party to hold off for a couple of hours or till the next morning. In case of absolute emergencies though, the paired team members could break the pairs in order to deal with the escalations. The same question could not be asked if the team was taking upwards of 20 days on average to finish work items. In fact, the inability to ask the question and lack of slack in the system was one of the major reasons for driving up the time taken to finish stories.

The manager of the team had this to say about their experience with Kanban principles:

> *"At first we laughed at the thought of intentionally limiting our Work In Progress and simplifying our Kanban board. We truly believed that this approach would "never work for our team". That was before April's Kanban training. We transformed our board, changed the format of our Standup and implemented sensible WIP limits and the way we work changed forever.*
>
> *Before Kanban 2.0, we thought we must be "slacking" if we had fewer than 40 stories on the board. Today we rarely break 20. Much to our surprise we discovered that the ideas from our training really do work for us! …This lets us adjust our feature work more rapidly and deliver higher quality features. As a manager, it's now possible for me see to all of the team's work at a glance and pinpoint areas of concern before catastrophe strikes! Finally, having stable cycle-time and Throughput data allows us to truly predict our capabilities for future release planning and emergency requests from Production.*
>
> *Today we laugh, or cry, when we think about the way we worked before!"*
>
> *-Leighton Gill*
> *Manager of Software Engineering"*

3.3   Organization Wide Impact

The improvements outlined above were not limited to just these two teams. In fact, the advances shown here were largely exhibited by all teams across the entire development organization. There was a marked increase in both the number of stories completed and the number of features completed between 2014 and 2015 (see Figure 3). The shorter story Cycle Times translated into faster completion of features. Faster completion of features translated into a dramatic increase in the total number of features delivered to customers: from 176 in 2014 to 411 in 2015. Looking at the month over month comparisons, every month in 2015 was more productive than the same month in 2014.

Figure 3: Month Over Month Comparisons of Features Completed

These organization wide improvements had the ultimate effect of streamlining our release planning process. That Cycle Times were so predictable and Throughput was so stable that it allowed us to experiment with more sophisticated planning techniques—the most important of which was Monte Carlo Simulation.

4.   MONTE CARLO SIMULATION AND PROBABILISTIC RELEASE PLANNING AND TRACKING

Monte Carlo Simulation (MCS) is a forecasting technique where a process's past data is used to simulate a system's future performance. The simulation technique produces a summary of risk levels that the business can use to determine how much risk it is willing to accept. We don't have space to go into too much detail about what MCS is and how to use it, so we invite you to explore the method on your own.

4.1   Release Planning

MCS is particularly useful to figure out the probability of meeting a certain delivery date given the number of stories needed to be done by that date. The results can be used to both plan a release and to track the progress of a team towards its release goals. The simulations, run at different points in the release can tell us if the team is falling behind on its commitments, is on track to meet its date, or can pull more work into the release.

At Ultimate, each team's release is independent which means that each team has its own release dates. Using the story data from the teams as outlined in Section 3, and feeding that data into an MCS, we have put together a release dashboard that tracks each team's progress toward its target dates. Below is a screenshot of the Monte Carlo release tracking dashboard that gets updated every hour each day to reflect the completion likelihood for every release currently in progress. The information here also includes the code freeze date for the release, stories remaining to be closed and the date where we can say with 85 percent confidence that the team will be done with the stories in the release.

MonteCarlo

Team	Release	Code Freeze Date	Stories Remaining	85% Completion Date	Completion Likelihood
Recruiting	June 2016	06/30/2016	9	06/29/2016	93.15%
Hiring Integration	June 2016	06/30/2016	5	07/29/2016	4.95%
Onboarding	June 2016	06/30/2016	13	07/11/2016	22.65%
TWIST	June 2016	06/30/2016	9	07/01/2016	76.70%
Platform Services	June 2016	06/30/2016	15	07/27/2016	0.05%
UTA	UTA-6.1.3v15	07/01/2016	5	07/05/2016	80.50%
UTM	UTM-2.7	07/01/2016	16	07/08/2016	32.70%
Talent Management	July 2016	07/08/2016	18	07/12/2016	79.90%
Recruiting	July 2016	07/15/2016	36	07/29/2016	6.85%
Business Intelligence	August 2016	07/28/2016	30	08/11/2016	19.50%
Workforce Management	WFM-Beta	07/29/2016	33	07/06/2016	99.99%
ACES	ACES-ENTCR	08/01/2016	16	07/13/2016	99.95%
Hiring Integration	August 2016	08/11/2016	11	10/11/2016	2.80%
SPS	SPS-Beta	08/31/2016	65	08/17/2016	99.80%
Compliance	V12.1.2 (R2 - Fall/YE 2016)	09/07/2016	216	08/10/2016	99.99%
Hiring Integration	V12.1.2 (R2 - Fall/YE 2016)	09/07/2016	1	10/18/2016	24.55%
SPS	V12.1.2 (R2 - Fall/YE 2016)	09/07/2016	1	08/17/2016	99.99%
Global Benefits	V12.1.2 (R2 - Fall/YE 2016)	09/07/2016	30	08/18/2016	99.70%
Payroll	V12.1.2 (R2 - Fall/YE 2016)	09/07/2016	57	07/29/2016	99.99%
Talent Management	V12.1.2 (R2 - Fall/YE 2016)	09/07/2016	40	08/30/2016	97.00%
Global HR and Compensation	V12.1.2 (R2 - Fall/YE 2016)	09/07/2016	110	09/01/2016	96.15%
TWIST	V12.1.2 (R2 - Fall/YE 2016)	09/07/2016	1	07/05/2016	99.99%
Platform	V12.1.2 (R2 - Fall/YE 2016)	09/07/2016	99	09/08/2016	82.80%

Figure 4: Monte Carlo Dashboard

This dashboard gives us a single point where the organization can look and see the risk of any given release completing on time. This dashboard is of such importance to our Agile practice at scale, that it becomes the focal point of an organization-wide daily tactical meeting called the Daily Product Review.

4.2 The Daily Product Review

The Daily Product Review (DPR) is Ultimate Software's successor to the Scrum of Scrums. The DPR, which is a 15-minute daily meeting, brings together the key metrics of Cycle Time and release completion likelihood in one place to provide the overall scorecard for the development organization. It reinforces the metrics and practices we care about on a daily basis. Below are some pieces of the DPR board that help us reinforce and scale these practices.

A slightly modified version of the Monte Carlo dashboard in Figure 4 finds its way to the DPR board. The main difference between the two Monte Carlo views is that the DPR view includes deltas from the previous day. This view is updated only once a day in the morning and for the Stories Remaining and Completion Likelihood columns contains the changes since the same time on the previous day. When a team's release starts to go red or starts slipping further into red they usually respond with any combination of the following strategies –

- Reducing scope of the release.
- Moving the date for the release.
- Working extra hours to bring the remaining stories count down.
- Or some combination of part or all of the above.

Another part of the DPR board is the individual Team Updates tiles. These tiles are color coded green, yellow or red based on the number of stories that the team has above the 95th percentile of the Cycle Times for their stories. The team can add notes to their tiles with the dependencies that are causing the stories to take a long time and the course of action they are taking to address the long running stories. The assumption here is

that anything exceeding the 95th percentile is probably something out of the team's control. As can be seen in the updates from the Payroll team below, there first story is blocked due to an external dependency and the second was blocked due to the lack of proper builds. They have also noted the strategies they have moving these stories forward. Teams can also update the tiles with other announcements that don't necessarily fall under the umbrella of General Announcements and are more team specific.
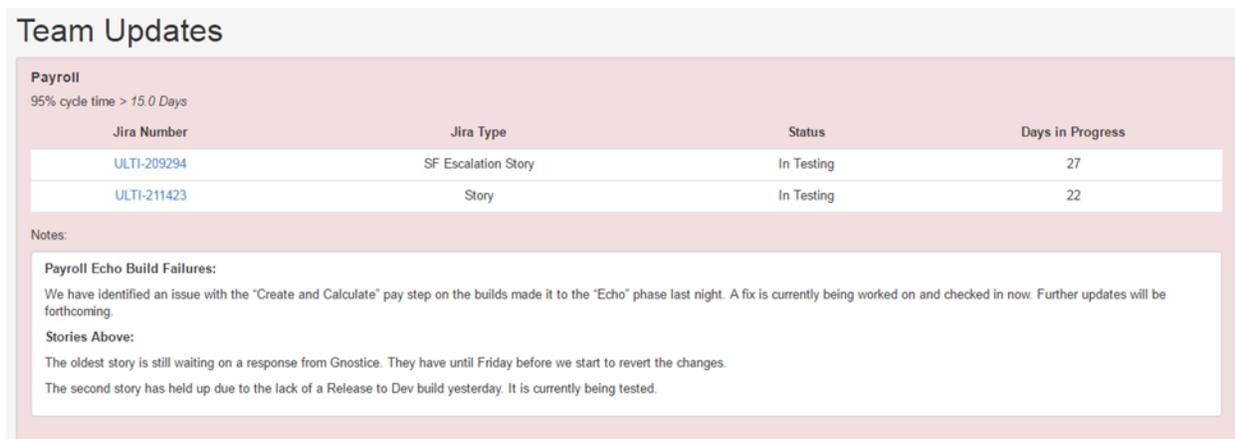

Figure X: Team Updates in DPR

5. MOVING BEYOND DEVELOPMENT

Adopting Agile techniques has provided the benefits of increased productivity and predictability. For an overall perspective though, Ultimate Software is in a waterfall sandwich. The Agile development organization sits in the middle of traditional sales and support organizations and traditional deployment and activation organizations. As a part of the next evolution of Agile and flow based thinking at Ultimate Software, we are expanding out to organizations that flank development. We are bringing to these organizations the same Agile principles that have worked for development.

Our closer engagement with Product Strategy and the ability to give them higher degree of predictability has vastly improved Development's ability to assist with support issues without interrupting active work. Tier 3 support has also adopted Kanban practices in order to improve their ability to support our customers. Product Strategy is able to utilize the predictability and productivity gains of Development to provide better guidance to Sales on upcoming products and features. As we continue to improve the predictability that we can provide Sales, we can start creating feature requests and priorities in conjunction with Sales. Features can then be pulled all the way through the value stream and tracking of cycle time and throughput can allow us to make and keep more accurate commitments to our customers.

While the upstream expansion helps us get better at the creation of value, expanding downstream to deployment and activations is where we can improve the delivery of value to our customers. As Ultimate Software has started working on new products, we have pulled deployment activities onto the teams. For our older products, we have always done a handoff to our Sass deployment group. We broke the "over the wall" mentality by embedding deployment engineers on the production teams for new products and helping them educate the rest of the team on maintaining their own deployment pipelines. These teams are supported by three groups outside of Product Engineering. These are operations groups that manage the Build and Deployment infrastructure for the products being developed. These groups have also adopted Kanban principles and started measuring cycle times for making infrastructure available to teams and making new features available in production. They have established SLAs for different types of requests and have become predictable with these metrics.

We can now see a feature make its journey all the way from a request generated in Sales to Product Strategy, to Development and finally to Production. Once we are able to track the progress of a feature in this manner, we can start identifying opportunities for improvement in the inception-to-delivery cycle. We can identify where features get stuck and apply our understanding of flow to eliminate the time features have to wait in queues across the entire organization.

Another aspect that is downstream from the development and even the deployment group is activations. Activations is the group that helps a new customer go live with Ultimate Software's products. The activation process can take up to a year and can involve multiple teams. Every day that a customer is in the activation phase, Ultimate Software is investing time, but not receiving any revenue. This is an area that can use the benefits that the Development Organization has gained from flow and Agile practices. Development has started working with Activations to share the principles and practices that have made a positive difference in the predictability and speed of completion for deliverables.

Moving Kanban outside the lines is the next large step for Ultimate Software. We have already started moving in this direction through our work with support and deployment teams. We also continue to reach out and get welcoming responses from other parts of the company that want to see the same kinds of results that teams within Development have seen. Ultimate continues to scale out its Agile implementation without using any established frameworks. Setting up the right channels of communication and visualizing our work in a manner that is easily understood by all is at the crux of how Ultimate has been able to successfully adopt and evolve Agile at scale. We continue our efforts to provide features to our customers at a predictable and faster pace thorough our practice of Kanban.

6. CONCLUSION

Through the innovative use of flow practices and principles Ultimate has been able to achieve many of the benefits of a Lean-Agile implementation without the use of a heavyweight framework:

- **Improved Productivity**: More features released to customers more quickly means higher overall customer satisfaction
- **Streamlined Planning**: Using techniques like Monte Carlo Simulation, the time it takes to plan a release has been reduced from days to minutes
- **Early Warning Signals**: Signs that a given story or a given release may be going off track are observed much earlier in the process allowing us to react and adjust
- **Easily Pivot**: Without a detailed understanding of our true capacity we would not be able to pivot to handle new customer requests and/or government regulations

We have been able to recognize these benefits much more quickly and at a fraction of the cost of a more traditional scaled Agile implementation. The practice outlined here are ones which any organization—regardless of size—can easily pick up and see immediate results.

7. ACKNOWLEDGEMENTS

REFERENCES

Vacanti, Daniel, "Actionable Agile Metrics for Predictability" ActionableAgile Press 2015 https://leanpub.com/actionableagilemetrics/