THE ROAD TO AGILE DATA SCIENCE

A Data Science Team's Journey from Scrum to Kanban to Data Driven Scrum

Data Science Process Alliance

Integrating data science process effectiveness research with industry leading agile training expertise
What is Agile, anyway?

Adored by many, hated by some, and misconstrued by most, agile is a set of principles that aim to align project and product teams with the broader organization. It espouses small, incremental deliverables, self-organizing teams, and adaptive planning.

Yeah, you’ve probably already heard all this and have seen the Agile Manifesto (right) before — most likely in a software context. But what about in a data science context?

What is Agile Data Science?

Simply put, it’s all about rapidly delivering meaningful incremental data science insights to stakeholders. These insights could be via reports, dashboards, data-driven strategy recommendations, or full-fledged machine learning systems.

Sounds simple, right? But, unfortunately, many data science teams struggle to achieve agility.

…but why is this so elusive?

Data science projects encounter numerous, unique headwinds that hamper a team’s ability to frequently ship meaningful deliverables. Some of the most common challenges are:

- Attempts to achieve agility using the same tactics as agile software teams
- The lack of defined data science-native agile frameworks
- The highly experimental nature of data science
- The complex layers of big data architectures
- The unknown data-related issues that will crop up in a project

How do you achieve Agile Data Science?

Like most questions surrounding data science, the answer is not straight-forward and is highly dependent on the team and project type. So to help answer this broader question, let’s hone into the voyage of one specific team as it evolved its processes in search for agility.
Scrum is a popular agile product development framework that defines values, roles, artifacts, and meetings. It is the most popular framework for software development. Other industries are also adopting it. Scrum teams break work down into small batches of tasks, deliver each batch of tasks, and revise plans based on feedback.

In Scrum, the product owner constantly updates a wish list of potential product features called the product backlog. The development team works to deliver top-priority items from the backlog in short, iterative fixed-length time periods called sprints. Meanwhile, the scrum master facilitates the overall process as a servant leader.

At the end of each sprint, the work output should be in a usable form. The team demonstrates this at sprint review and conducts a retrospective meeting to improve processes.

Kanban is a light-weight set of principles that was originally created for lean manufacturing. Software engineering, among other domains, has since started to adopt it. Kanban focuses on making work visible to help facilitate communication and collaboration. Unlike Scrum, Kanban does not define iterations, roles, or meetings.

Kanban's central focus is a highly-visual Kanban board which shows each life cycle phase of a development task as a column. The first column is typically a sort of to-do list, similar to the product backlog concept from Scrum. When the team has capacity to work on a new item, it pulls the highest priority unblocked work item into the next column (e.g. doing). The team continues to move this work item across each additional column (such as in test or done) provided that not too many items pile up into a single column per the work-in-progress (WIP) limits defined for that column. WIP limits help identify and resolve bottlenecks, minimize the adverse impacts of task switching, and reduce cycle times.

Data Driven Scrum (DDS) is a newer continuous flow framework designed specifically for agile data science teams. DDS integrates the structure of Scrum (e.g., adapt Scrum’s roles, artifacts, and meetings) and the continuous flow of Kanban. Specifically, DDS alters some key aspects of Scrum to address the typical challenges data science teams encounter when using Scrum.

DDS teams use a visual board and focus on working on a specific item or collection of items during an iteration, which is task-based, not time-boxed. Thus, an iteration more closely aligns with the lean concept of pulling tasks, in a prioritized manner, when the team has capacity. Each iteration can be viewed as an experiment to validate or reject a specific lean hypothesis (or answering a specific question).

Specifically, an iteration is defined by the following three steps:
1. Create something
2. Observe the outcome of that creation (e.g., via measurement)
3. Analyze those observables and create a plan for the next iteration.
Background Context
Organization, Team, and Projects

Organizational Context: The organization is a large profitable in the entertainment industry with over 10,000 employees located across the world.

Team Context: Known as the Big Data Science (BDS) team, the team had been growing for several years; and, at the time of the study, had 21 members distributed across two locations and on three sub-teams.

Data Engineering: Ten team members sourced and cleaned data and built software to maintain these systems. In many ways, the data engineering team was the most similar to a software development group – they were able to define modular tasks with well-defined requirements.

DevOps: There were six people in the development and operations group, where their focus was on running the IT environment as well as defining and creating the data and system architectures.

Data Science: There were five people in the data science group, who mainly focused on building models, reviewing the usefulness of potential data sources, and trying to ensure their models were useful to their (internal or external) clients.

Each group had a group leader, each with more than five years of relevant work experience. The individual contributors also generally had significant knowledge and work experience (see the org chart below).

Project Context

At any given moment, the BDS team executed multiple projects. Most projects were internal but some were for external clients.

Clients, both internal and external, were often very engaged (sometimes trying to suggest what solution they wanted rather than focusing on explaining the problem / opportunity they were looking to solve). However, other clients were not engaged and would just want to know ‘how long and how much’.

In general, most of the projects were exploratory in nature and required a big data computing infrastructure. Projects typically lasted between 3 to 6 months.

The composition of each of the specific project teams varied based on the team’s collective view of what was needed for that specific project effort. However, most projects had 4 - 8 members with representation from each sub-team and each location.

While the data engineering group did much of data collecting / munging work, as appropriate, the other groups also helped.

Depending on the specific project, the team used a range of data analysis tools, including typical data science programming languages such as Python and R.
3 Challenges with Scrum

The team faced three major issues with Scrum:

First, many team members needed to respond to issues as they arose. So, as stated by one DevOps group member “if something went wrong with the platform, the team would receive a ticket on the issue, triage it, and address the issue according to its priority/time sensitivity”. But, “these unplanned tasks impacted the group’s ability to deliver the sprint on time”.

Second, task estimation was inaccurate. The goal was that each task would be completed in approximately 8 hours; however, these estimates often missed the mark. As one data scientist noted, “many tasks are exploratory in nature, and as such, can’t be accurately estimated”. Furthermore, since some of the data was new to the team, team members did not always know how much time a data exploration analysis task would require.

Finally, the team was much larger than the 3-9 development team members that Scrum prescribes, but they did not elect to use a scaling framework such as scrum@scale. This created communication challenges, which was apparent during their standup meetings. It was challenging for team members to keep track of the progress of each project, and stakeholders were confused as to what was going on with respect to their project. It was believed that were many additional meetings a result of poor standups.

First Destination
Scrum

The team initially implemented Scrum because many team members had positive experiences with it on software teams.

One Scrum Master supported the entire BDS team and each project typically had a product owner from the business.

The team executed two-week-sprints which consisted of the following sequential activities:

1. Sprint planning session — tasking and pointing for next sprint
2. One week of development / analytics work
3. Sprint backlog grooming session for the upcoming sprints
4. One week of development / analytics work
5. Sprint demo session — showing output of closing sprint
6. Sprint Retrospective — what went / did not go well

Work for each project had a typical product hierarchy. A product, or overall analysis, was comprised of one or more epics, which was a group of related stories. Stories were self-contained units of work agreed upon by the groups within the BDS team as well as by the stakeholders (product owners). Each story was further broken down into tasks which would be things like “contribute to building the data loading process”, “validate raw data” or “build a dashboard”. Each task had a build and a test component.

The project team used a web-based tool to assign tasks and track their status. Each task in the sprint backlog progressed in a Kanban-like visual board with the following sequence: “in waiting“, “in development“, “in testing“, “ready for sign off” and “done”. All the projects were on the same board, but it was noted which tasks were related to which project.

A typical day started with their daily Scrum (standup) call, which included all members of the BDS team, across all the projects, as well as any relevant and available business product owners. During this call, the team members would discuss the tasks they worked on the previous day, the planned tasks for the current day, and any potential barriers for task completion. The entire BDS team attended one daily call, as opposed to one call per project. The team did this to enable all BDS team members to help resolve any identified bottlenecks and roadblocks across all the projects.

There were, as needed, other project meetings during the day. Some would be planned in advance while others would be called as needed. These meetings were with other members of the team (either on-site or remote), representatives of the business to clarify project requirements, or with people outside the company for possible new data sources.
Due to the challenges with Scrum, the team migrated to Kanban.

In terms of the actual Kanban board, each group created and used a four-column board. While their column titles were “requested”, “in progress”, “validate” and “closed”, these were equivalent to the more traditional “to do”, “doing”, “test” and “done”. Note that there was a fifth step in their process, in that the task was removed from the “closed” column when it was confirmed that the task had been successfully completed (ex. confirmation by the client).

To help define and track work, the team used a web-based tool for creating and visualizing their Kanban board. Note that each group had their own board (and set of tasks to execute), and the different groups within the team could view the other group’s boards. The team did not use WIP (work-in-progress) limits for each column on the board, but they did have a focus of trying to limit their WIP.

As an example of how the boards were used, the data science group used the Kanban board to intake project requests from different business units. Then they prioritized the tasks in the requested (to do) column. Once they started on a specific task, they used the Kanban board to track its progress. Specifically, when a task was completed within a column, that task got moved to the next column (ex. “closed”). Since Kanban did not have a specific target task completion date, creating additional tasks in the to do did not cause concern nor was it seen as an issue.

In other words, when using Kanban, the process could be thought of as a pipeline with requests entering one end and improved data insight coming out the other end. Each group worked through their pipeline without a defined schedule. The goal was to make sure that there was not a lot of time spent on an effort that did not complete, since as stated by a team member, it was “better to get a fewer number of tasks all the way through the pipeline”.

Since Kanban does not define specific meetings, meetings were held as needed. As appropriate, these meetings were with team members (either on-site or remote), representatives of the business to clarify project requirements, or with people outside the company for possible new data sources. Some of the meetings would be planned in advance while others would be called for that day.

Note that, the team did not define any metrics to analyze the flow of work. This was partially due to the organization being new in the use of Kanban, and partially due to the team having a high-performance culture, where the team members were trusted to work as efficiently and effectively as possible.

Second Destination

Kanban

3 Challenges with Kanban

The team faced three major issues with Kanban - all focused on the fact that Kanban does not define any process specifics.

The first issue was the lack of focused, well-structured meetings. Initially, the team liked the fact that there were no regularly-defined meetings, in that they could focus on “doing the work”. However, this started to create problems. For example, everyone was not on the same page with respect to how to prioritize the tasks in the to do column or where were the project bottlenecks. Equally concerning, the team’s communication with the stakeholders was becoming less frequent and less effective, which was causing issues such as the team not understanding the value (or lack of value) in their completed tasks.

Second, there was no focus on improving their process. While Kanban has a focus on trying to always reduce WIP, and team members did try to identify ways to reduce WIP, there was no process improvement effort defined, nor a way for team members to brainstorm potential process improvements.

Finally, many of their clients complained that there was a lack of specific deliverables, which created questions such as ‘when should I test the work done for my project’. In other words, the continuous flow made it difficult to structure the dialog with their stakeholders.
What about CRISP-DM (or other life cycles)?

CRISP-DM describes six iterative phases (business understanding, data understanding, data preparation, modeling, evaluation, deployment).

Typically, when using CRISP-DM, the team progresses through the different phases as they deem appropriate. When using CRISP-DM, as needed, the team can “loop back” to a previous phase (ex. data prep).

The BDS team defined a CRISP-DM like series of phases (but added a new phase ‘monitoring’ as the last step in the life cycle).

For each iteration, the team followed their defined life-cycle. Specifically, as an iteration was being broken down into tasks, each phase of the life cycle was reviewed (to ensure tasks were not forgotten). Furthermore, each task was noted as being part of a specific life cycle phase.

The use of a life cycle within an iteration (experiment) helped the team members think about the work to done for a specific iteration.

Note that for some iterations, some of the phases were not applicable. For instance, effort for business understanding or data preparation might not be required. In this case, the team just skipped that phase (but the team never just assumes a phase could be skipped).

The team migrated to Kanban prior to their awareness of Data Driven Scrum. However, once the team knew about DDS, they quickly realized it addressed many of the team’s concerns with Scrum as well as their recent concerns with Kanban.

When using DDS, the team worked collectively to determine what specifically needed to be done during an iteration, what data should be observed and analyzed, and what would be required to collect and analyze the information generated from that iteration. Then, during backlog selection, the team collectively reviewed their backlog items to come up with a specific experiment to run.

Task estimation, which was needed to help prioritize their iterations, the effort required to run the experiment (i.e., perform one cycle of create, observe and then analyze), was done at a high level (with only high, medium and low estimates).

An example backlog item was to explore customer satisfaction by age. This task was broken down to explore age via overall customer satisfaction, as well as satisfaction by geography (e.g., per each state in the United States). The team determined that the item required four tasks on the board, two related to data munging, one to calculate customer satisfaction across different loyalty levels by age, as well an effort to explore customer satisfaction by age from a geographic basis.

This experiment (item) was prioritized as important because the team hypothesized that age might be an important characteristic of customer satisfaction. Once it was clear how the team was going to create, analyze and observe the experiment, the team began their iteration. During the iteration (and all other iterations), the team’s board was defined with “to do”, “in progress”, “validate”, “observe”, “analyze” columns. The teams used these columns since there was the belief that there were not specific observe / analyze tasks, but rather, each task should be observed and analyzed.

Each day, the team had their daily standup to identify issues and roadblocks. This iteration took 1.5 days. Once the iteration completed, the team started the next iteration. The team discussed their findings in their next scheduled iteration review meeting, where they agreed on new experiments to be added to the backlog. Their process improvement was done via their monthly retrospectives.

Finally, with respect to metrics, the team focused on tracking:

- **Top Item time**: % of time focused on the top 2 iterations
- **Cycle iteration time**: How long it takes an iteration to complete after the item work is started.
Summary

This case reports on the evolution of the process used by an organization for data science projects, where three different agile process frameworks were used (Scrum, Kanban, and then Data Driven Scrum). The case helps to clarify the concept of agility within a data science project, as well as the key process challenges a team might encounter when executing a data science project.

Within this organization, the team’s desire for agility was driven by the need to iterate through the many possible analytical avenues, since there was no clearly defined path to follow with respect to what were the key questions to address nor the actionable insight to generate. The team wanted to focus on maximizing the number of empirical experiments/iterations that they could achieve, weighted by the value of each experiment / item to be explored.

In short, the use of Data Driven Scrum provided the team with some structure (ex. meetings to prioritize work and refine their process), while at the same time, still allowing the benefits noted in their use of Kanban (no time-box sprints, but rather, a lean pull-based approach to task execution).

Key Takeaways from this Case Study:

- Reflecting / improving a team’s process is key to achieving an effective process
- Data Driven Scrum (DDS) leverages key aspects of Scrum, but addresses some of the challenges in using Scrum for Data Science Projects
- DDS can be viewed as one implementation of Kanban, but with a defined process framework to help teams execute the Kanban focused project.
- Agility is useful for data science and can be achieved by focusing on a lean iterative framework

Key Tenets DDS and agility within data science projects:

- Agile is intended to be a sequence of iterative experimentation and adaptation cycles.
- The goal of each cycle should be to have an idea or experiment in mind, to build it, to then observe the analysis, and then to analyze those observations - helping to define the next idea or experiment.
- Going from an initial idea, through implementation, and the analysis of the results should be the basis for an iteration (not a predetermined number of elapsed hours).

With these tenets in mind, teams should following these three principles:

- **Allow capability-based iterations** - it might be that sometimes it makes sense to have an iteration that lasts one day, and other times, for an iteration last three weeks (ex. due to how long it takes to acquire / clean data or how long it takes for an exploratory analysis). The goal should be to allow logical chunks of work to be released in a coherent fashion.

- **Decouple meetings from an iteration** – since an iteration could be very short (ex. one day for a specific exploratory analysis), meetings (such as a retrospective to improve the team’s process) should be time based on a logical time-based window, not linked to each iteration.

- **Only require high level item estimation** – In many situations, defining an explicit timeline for an exploratory analysis is difficult, so one should not need to generate accurate detailed task estimations in order to use the framework. Rather, high-level “T-Shirt” estimates are required to help prioritize the product backlog (but there is no more detailed estimates required during an iteration).
Data Science Project Management Training

Data Science Process Alliance (DSPA) empowers individuals and teams to successfully set up, execute, and deliver data science projects. Members achieve this by applying practical knowledge gained through courses dedicated to data science project management.

Better Process Delivers Improved Results

GAIN EFFICIENCY
Repeatable processes drive team efficiency and help ensure the highest value analyses are explored.

AVOID COMMON PITFALLS
Understand how to recognize common project management issues and how to quickly address those issues.

ACTIONABLE INSIGHTS
Ensure insights that are actionable and understood by stakeholders via better communication and coordination.

Training and Certification

The Data Science Team Lead™ courses provide in-depth, comprehensive, and actionable training in data science project management.

Unlike other organizations, all we focus on is data science project management. Contact us to learn what better project management can mean for your organization.

Course Components

- Individualized mentoring (DSTL and DSTL+ courses)
- Access 6+ hours of on-demand video content
- Examine a real-world case study
- Discuss trends, blogs and other emerging items
- Get exclusive Training on Data Driven Scrum
- Access a library of curated resources (white papers, videos, etc)

Private Group Training

We offer group training for your team and can easily create a training program for your specific organizational needs. Contact us to learn more.

About the Data Science Process Alliance

Combining data science process research with industry leading agile training, the Data Science Process Alliance (www.DataScience-PM.com) is the only organization dedicated specifically to improving data science project management.

www.DataScience-PM.com  © Data Science Process Alliance  2022