TREXIN INSIGHT PAPER

February 28, 2018

AGILE PRACTICES FOR DATA ANALYTICS

An iterative framework for exploring data.

We work with a lot of organizations that analyze data to answer big questions about their business but whose stakeholders offer insufficient guidance and direction to the people doing the analysis. Very often, data analysts are chartered to find answers to a loosely-connected set of questions with no relative priority, importance, or interdependency. The analysts have an unrealistic schedule, or they don't negotiate one. And, they don't schedule a time to review their findings with their stakeholders at regular intervals to obtain feedback. Making matters worse, the analysts often do not fully understand the end-to-end processes in which the data they are analyzing gets created or modified.

Like anyone doing software development without guidance or direction, data analysts can go adrift, producing findings that they may find interesting, but are neither actionable or provide valuable insight to their stakeholders.

More than application development, data analytics is an exploration, or testing of hypotheses, with few requirements determined up front. Rather, the course of a data analytics project must respond quickly to stakeholder feedback from discoveries or findings in the data. We found that applying Agile Framework allows for this quick response from regular feedback and thereby enables data analysts to deliver better outcomes.

BUILD A BACKLOG

To begin to apply Agile, build a backlog. A backlog, in Agile, is simply a list, or work queue, in which pending work items are refined and assigned priority. In data analytics, a backlog should be a place to prioritize, break down, and sequence the questions stakeholders are asking. Using a backlog enables data analysts to align their work to stakeholder's needs, plan their work, and set realistic expectations for delivery.

To start the backlog, data analysts should capture the questions that their stakeholders have about their business and put them in a list. Grouping them by relevance is helpful.

However, a simple question does not tell the analyst why the question is being asked in the first place. Answers to large, complex questions are composed of a network of answers to interdependent questions. For example, the question "What is the ratio of widgets used to widgets shipped?" depends on the answers to two child questions:

- "What is the number of widgets being used during the period?"
- "What is the number of widgets that were shipped in the period?"

The analyst will need to supplement the questions with information that allows them to break down the question into its dependent questions by answering the why to each question until they get to the bottom-most, base elements. The format the analyst should use to do this is the User Story.

In Agile, the basic unit of work in a backlog is a User Story, which describes the function that a system must provide a user. In data analytics, we can modify the User Story format to identify what we are looking to answer and why. The User Story in data analytics would have three clauses:

> *As a <role of person needing the answer>*
> *I need to know <question>*
> *So that I can <solve problem X, answer question Y>*

The first clause identifies the stakeholder needing this information, and it may be optional if only one stakeholder is asking all the questions. The second clause contains the question and is not optional. The third clause identifies the reason for this question. The third clause is critical if the user story is a dependency of another question or set of questions.

To complete the backlog, the analyst should break the stories down into child stories until they contain only one element with no dependencies.

PRIORITIZE THE BACKLOG

With the questions framed up in user stories and decomposed into their dependencies, the next step for the analyst is to engage their stakeholders in prioritizing the user stories.

Using the list of user stories, the analyst should first size the effort needed to resolve answers to each story and its dependencies. If the data needed to resolve an answer to a story is not readily available in the analyst's environment, and will need to be requested, extracted, transformed, and loaded, then the analyst should create a user story to identify this data requirement. It should be a dependency of any user story that needs this data. Like other stories, the analyst should estimate the size of the effort needed to get the additional data.

Estimating the size of each backlog item before asking the stakeholders to prioritize them will help identify those stories which, if answered, can provide the greatest value for the least amount of effort. This "bang for the buck" metric may help prioritize the user stories.

The analyst asks their stakeholders to assign an index of relative value or importance to each of the major user stories. The stakeholders should understand that by indicating their priority, they are communicating which user stories they want to have resolved before others. Hence, not everything can have a maximum priority.

PLAN THE WORK

Stakeholders seldom understand how much effort it takes to answer complex questions for which data may not be available or complete. Unless the analyst engages their stakeholders in planning out the analysis work, stakeholder expectations around delivery timeframes are likely to be unrealistic.

We recommend that analysts and their stakeholders plan to complete as many prioritized user stories as they can within five two-week sprints. Schedule reviews of findings with stakeholders every two weeks. Incorporate feedback at these reviews and re-prioritize the user stories in the backlog when necessary.

The ten-week planning horizon affords a sufficient amount time to complete the larger and more complex user stories, especially those that require adding data to the analysis set. Reviewing the findings every two weeks maintains stakeholder engagement and a level of guidance and direction.

Near the end of the ten-week period, the analysts should summarize findings from the entire ten-week period, and plan for the next ten-week period.

UNDERSTAND THE END-TO-END PROCESS

Too often, data analysts do not understand the process from end to end in which the data they analyze gets created or modified. Not having this understanding introduces a risk that analysts will go off-track and produce findings that are not based on what the data actually represents.

Whenever this understanding is missing, the analysts should plan an effort to gain this understanding and create a document that clearly captures and communicates the activity flow and the data that gets captured along the way. Analysts should create user stories to complete the process analysis, decompose them into child stories, and add them to the backlog, making them dependencies of any user story that seeks to resolve answers to questions about this end-to-end process. Stakeholders should give these process analysis user stories sufficient priority in order to ensure meaningful and actionable findings.

SUMMARY

We have seen analysts and their stakeholders achieve closer collaboration, realistic expectations, and actionable findings when they put the structure of an Agile framework around their efforts. We found that by using this iterative framework to structure a process of exploration, organizations can yield the maximum valuable insights from data. We recommend it.

This TIP was written by David O'Brien, who specializes in Financial Services and Agile Frameworks. David welcomes comments and discussion on this topic and can be reached at david.obrien@trexin.com.