Determine Cycle Times

# Discussion

Power BI is not able to determine the cycle times as it doesn’t have the ability to do a comparision against a prior row’s timestamp.

The goal of determining cycle times is to look how long it took from the point of starting the work and movement to different states in JIRA. Additionally, we would want to know how long items were blocked by various outside entities when those entities had indicated that they were ready to provide the needed service. By rule, we don’t start work unless we know that they have made the commitment.

The control chart report in JIRA does not accurately represent times. By example, [INFAOP-456](https://jira.t-mobile.com/browse/INFAOP-456) reports a 0 cycle time. The history table shows that the item went into the In Progress state on 10 Oct 2018 and was marked done on 23 Oct. As a matter of course, there wouldn’t be any issues which would complete in less than a half day. By rule, we don’t enter issues which would complete in less than a half day as it’s expected to be included as standard daily operations. The control chart also includes items which were not actually worked but were ‘resolved’ by any means – including canceled or rejected work.

The goal of this process is **not** to determine the cycle type by specific team members as this is not possible. Multiple team members may work on various parts or help each other out. If there were outliers then one could look at those records in JIRA or through other reporting mechanisms.

# Need

Create a service which polls the Jira Database in the NDW which does the initial load of data from the past 18 months and conducts the various calculations.

Create a database which stores the data (possible model included) which Power BI could reference. On a daily basis, this service would run and look at newly created issues and history records.

A service account and location to run the service.

Code review by a software developer to ensure that best practices are followed. This service could be completely written by a software development team, but this would be considered a low priority project. Therefore, requesting code reviews may be the best way to accomplish this task.

Database review and setup by a DBA to ensure that best practices are followed.

# Rules

1. Do not calculate cycle times for items which are **Done** but not in a **Done** resolution state. We don’t want to calculate times for things which were either rejected, cancelled, or otherwise determined that we would not conduct work. Delete any records which enter a non-tracked state.  
   **Note:** If something entered a work state and then went to a non-tracked resolution state, the record will be marked as such so we can still determine the amount of time spent.
2. Do not calculate the number of times items go into prior states.  
   Without building workflow definitions into the tool, we cannot determine the number of times things went into an incorrect state.  
   By example, a workflow generally goes from To Do -> In Progress -> Done. We may want to know how often something goes backward from In Progress to To Do as this indicates that we started and then had to hard stop a piece of work. This will be simply captured by the amount of time something spends in a specific state.
3. Time between the assignment of story points to when it enters a sprint which is either (currently) active of complete.   
   **Goal:** See how often we story point items far in advance of the work being scheduled. Items pointed too far in the past *may* not take into account changes to the environment or the customer’s current needs.
4. Determine the amount of time between:
   1. Entered in valid sprint to the In Progress state. Note: If configured it may be a specific state rather than a sprint. Some teams use the **Pending** state to indicate the work is scheduled but may not be a sprint yet.
   2. The amount of time in a ‘to-do’ state once it entered a valid sprint. This would include time when it goes from **to do** to any other valid state and then back to the **to do** state.
   3. The amount of time that something spends in the any state.
   4. The amount of time it takes an item to move from from a valid state to a done state – so that the BI tool does not have to do a calculation against an unknown number of states.
5. Calculate the number of times an item went into the blocked state from any other state.
6. Calculate the number of times an item’s expected unblocked date changed, with the exception of the first date and when the date is cleared when it enters the Done state.