Understanding ShiftWorx Data

**Definitions and Context**

* Data Sources
  + Fundamental data structure used in ShiftWorx. It is a tree structure that is used to represent many things, from the factories themselves down to individual data points tracked by our sensors.
  + Typically used in a structure like the following, but with more than 1 machine.
    - Company
      * Machine 1
        + Input 1

count

state

timer

* + - * + Input 2

count

state

timer

* + - * + Input 3

count

state

timer

* + - * + Input 4

count

state

timer

* + There is a **type\_id** column that will have one of the following values:
    - 1 => FreePoint itself
    - 2 => Legacy Box - our black box that connects to a machine. Used to represent the machine it is hooked up to.
    - 4 => Legacy Box Input
    - 5 => Company (represents the customer/factory)
    - 6 => Generic (user made DS mostly used for categorizing/organizing their Data Source trees)
    - 7 => Primitive/Data - These Data Sources are the only type allowed to store time series data sets under them.
    - 8 => Custom Expression – We haven’t given you any of these, but these can be used by our customers to create custom metrics using a set of available functions that can be arranged in a math expression like *count(123) / time\_range()* to get the parts produced per minute on a Data Source with the of ID 123. They can then hook up these expressions to KPIs or reports that determine the time range(s) for the calculations and some rule.
* Data Source Attachments
  + When users start Jobs, Status Codes, or Operator clock in/out it creates a Data Source Attachment.
  + We’ve separated them into 3 separate data dumps. Each will have a column containing the ID of the entity being attached (**job\_id**, **status\_code\_id**, or **user\_id**).
  + **attached\_at** is a UTC date time string representing when the attachment started.
  + **detached\_at** is a UTC date time string representing when the attachment ended. If this value is NULL, it means the attachment hasn’t been ended yet.
* Users (AKA operators)
  + The Users.csv file will contain a simple list of the users for the factory with an ID and name.
  + The IDs in this file will correlate to the **user\_id** column from the *Users Attachments.csv* file.
  + The attachments represent the time a person was working at the Data Source. Typically, these will be against the machines only.
* Status Codes
  + These are used to give context to operational state of a machine. These are also in a tree format where each Status Code has a **parent\_id** to point to its parent Status Code.
  + There are 2 “root” codes: Uptime, and Downtime. These are shared for all companies. The idea is that users can add additional context to basic uptime and downtime.
  + The primary use of these is to explain downtime, and uptime codes are rarely used.
  + The *Status Codes.csv* file contains the definition a customer’s Status Code Trees.
  + The *Status Codes Attachments.csv* file contains data of when and for how long codes were applied to a Data Source.
* Jobs
  + The *Jobs.csv* file contains a list of all of the company’s jobs.
  + The *Jobs Attachments.csv* contains Data Source attachments representing when jobs were started/stopped on the target Data Source. The **job\_id** column maps to an id from *Jobs.csv*
  + Fields:
    - ***target\_runrate*** / ***target\_runrate\_scale\_factor*** – If the target is to make 10 things per minute, **target\_runrate** would be 10 and **target\_runrate\_scale\_factor** would be 60. 10 things per hour would change the **target\_runrate\_scale\_factor** to 3600.
    - ***multiplier*** – multiplier to apply to part count for reporting purposes. Especially for presses, this is needed because 1 count from our sensor might mean the press closed once, but the mold may have made more than 1 thing.
    - ***expected\_cycle\_time*** / ***expected\_cycle\_time\_deviation –*** time in seconds that an ideal cycle (time between two consecutive state high events) should be, and an allowed deviation from that expressed as a percent. Example: if the expected cycle time is 12 seconds with a deviation of 50%, 6 – 18 seconds is within tolerance, but anything outside of that isn’t.
    - ***budgeted\_scrap\_percentage*** – Dictates a target to stay under for scrap creation. If you make 100 things in a given time frame, and your budgeted scrap % is 10, it is expected that the process should generate 10 or less scrapped parts.
    - ***standard\_operators*** – number of operators expected to be working on the task concurrently. If this value is 2, then they expect 2 operators to be able to hit any targets set. Which means that if 1 operator were to do the work within targets, they operated at more than 200% efficiency.
* Scrap
  + The *Scrap Types.csv* file contains a list of scrap types defined by the user.
  + The *Scrap Events.csv* file contains data about scrap made. The **scrap\_type\_id** column maps to an id from *Scrap Types.csv.* The **attached\_at** column is when the scrap event occurred, and the **count** column is how much scrap was made at that time.
  + Scrap should not be affected by a Job’s multiplier.
* State Mask
  + See screenshot below for example of structure.
  + The **previous\_value** property tells you what state the machine was in before the target range, and when it entered that state.
  + The **state\_mask** property is an array of state change events. Each event says what state it changed to, and what time it changed at.
  + The **time** values are unix timestamps in seconds.

