Data 604 - Final Project Presentaion

Hockey Stick Factory Manufacturing Simulation

Joseph Simone

7/15/2020

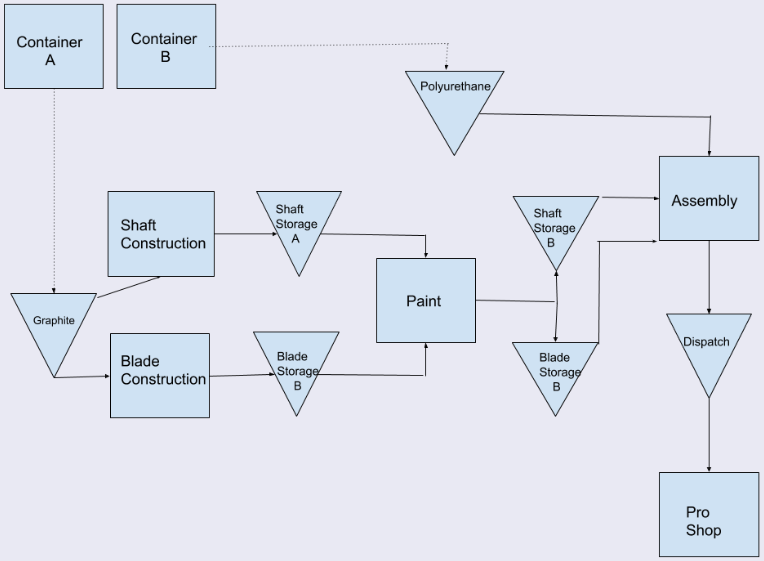
Hockey Stick Factory

Simulation

- The goal of this project was building a Manufacturing Simulation using the Python Package `simpy`.
- In simplistic terms, the construction of a hockey stick can be subsetted into the shaft of the stick and the blade piece elements.
- In this simulation, both the shaft and the blade are constructed seperately, however, from the same type of Graphite material.
- Then both elements are passed to be painted by the same Paint material.
- From there, both elements are passed to the assembly area where both pieces will be fitted together and a coated of Polyurethane will be applied.
- That is only a small look at this simulation from the surface.

Factory Continued

Flow Chart



Flow Chart

Explanation

- In the simulation there are two main `simpy` containers comprised of *"Graphite"* and *"Polyurethane"*.
  - Containing `N` amount of both graphite and polyurenthane materials that will be utilized for production.
- A hockey stick will be fashioned into two parts from material in the graphite container.
- Part A being the shaft of the stick, which will then be stored in the Shaft Storage A awaiting painting.
- The same process will occur with the blade end of the stick, however unlike the stick which is build from one piece of graphite, two blade can be with one piece of graphite.
- The painting process will collect both the shafts and blades from their respective storages then be moved to the Shaft and Blade Storage Containers B.

Flow Chart Explanation

Continued

- The assembly will then collect a shaft and a blade element, taking Polyurenthane from the Container and assembling the hockey stick.
- Then the completed hockey stick will be stored in the Dispatch container.
  - After an `N` amount of hockey sticks are stored after constuction, an alert will be sent for the Pro Shop to pick the sticks up.
- In addition, when either the Graphite and/or Polyurethane Containers reach below a certain critical level of material, an alert to call a Supplier go out.
  - After `T` amount of days, the Supplier will arrive at the factory and the critical levels of the material will be accounted for.

Constuctor Class

```
class Hockey_Stick_Factory:
    def __init__(self, env):
        self.graphite = simpy.Container(env, capacity=graphite
        self.graphite_control = env.process(self.graphite_stoc
        self.poly = simpy.Container(env, capacity=poly_capacit
        self.poly_control = env.process(self.polyurethane_stoc
        self.shaft_pre_paint = simpy.Container(env, capacity=s
        self.blade_pre_paint = simpy.Container(env, capacity=b
        self.shaft_post_paint = simpy.Container(env, capacity=
        self.blade_post_paint = simpy.Container(env, capacity=
        self.dispatch = simpy.Container(env, capacity=dispatch
        self.dispatch_control = env.process(self.dispatch_stic
```

Shaft Builder

Algorithm Construction

```
def shaft_builder(env, hockey_stick_factory):
    while True:
        yield hockey_stick_factory.graphite.get(1)
        shaft_time = random.gauss(mean_shaft, std_shaft)
        yield env.timeout(shaft_time)
        yield hockey_stick_factory.shaft_pre_paint.put(1)
```

Note: This loop will be used in all the “builder” function through this program creating: 2 shaft builders, 1 blade builder, 1 painter and 4 assemblers.

Simpy

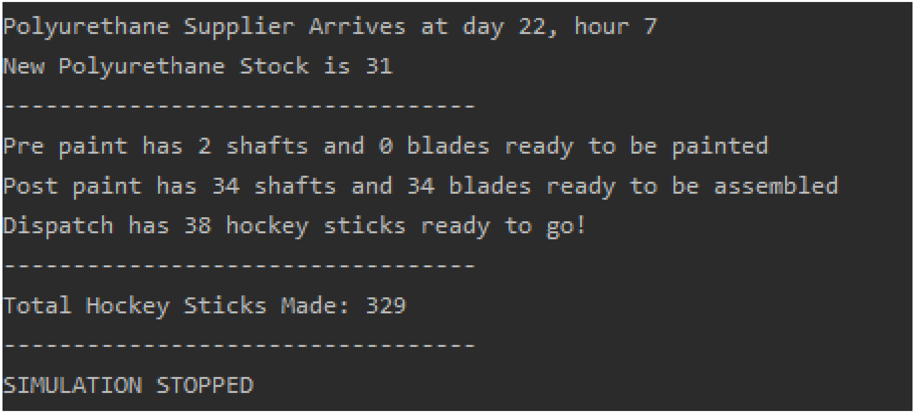
Enviroment Construction

```
env = simpy.Environment()
hockey_stick_factory = Hockey_Stick_Factory(env)

shaft_gen = env.process(shaft_builder_gen(env, hockey_stick_fa
blade_gen = env.process(blade_builder_gen(env, hockey_stick_fa
painter_gen = env.process(painter_job_gen(env, hockey_stick_fa
assembler_gen = env.process(assembler_job_gen(env, hockey_stic

env.run(until=total_time)
```

Month Simulation



```
Polyurethane Supplier Arrives at day 22, hour 7
New Polyurethane Stock is 31
-----------------------------------
Pre paint has 2 shafts and 0 blades ready to be painted
Post paint has 34 shafts and 34 blades ready to be assembled
Dispatch has 38 hockey sticks ready to go!
-----------------------------------
Total Hockey Sticks Made: 329
-----------------------------------
SIMULATION STOPPED
```

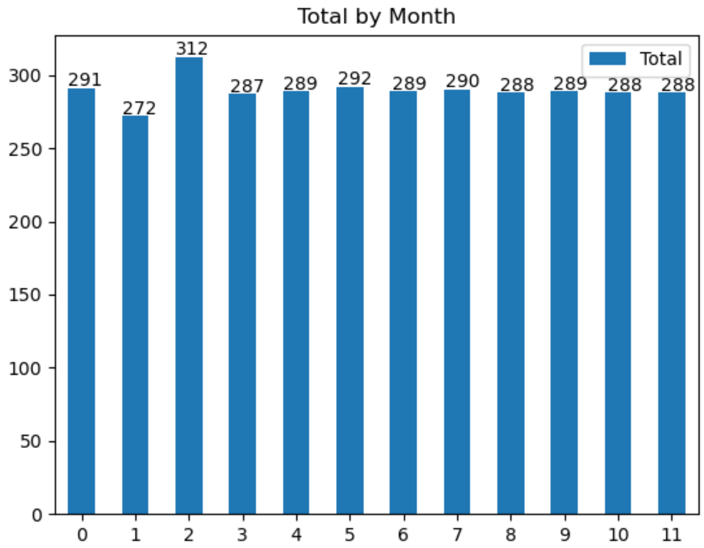
Year Simulation

```
for i in range(12):
    env = simpy.Environment()
    hockey_stick_factory = Hockey_Stick_Factory(env)
    shaft_gen = env.process(shaft_builder_gen(env, hockey_stic
    blade_gen = env.process(blade_builder_gen(env, hockey_stic
    painter_gen = env.process(painter_job_gen(env, hockey_stic
    assembler_gen = env.process(assembler_job_gen(env, hockey_

    env.run(until=total_time)
```

Year Simulation

Production by Month



Conclusion

There are many advantages for running a Manufacturing Simulation, from estimating productivity to projecting growth. Today, given the Supply Chain issues that the world is facing, it is more crucial than ever to keep track of inventory. This allows for companies or manufacturers not only to check if their inventory is low but if it is over stocked as well. Inventory is money wasting away, however, running out of stock is an entirely seperate issue.

`Simpy` is a great package for create Discrete Event Simulation and would highly recommend it to anyone in the field.

Looking Forward

- I created this simulation to work well within a Python Environment and print out the variable counts very well. However, this lead to some difficult retrieving the hockey stick total that were store in the Dispatch `simpy` container.
- I would like to rework this simulation to be able to store this data in additional `pandas` DataFrame for easy analysis.
- In addition, I would like to make this more User Friendly, creating a Flask Application with interchangeable variable conditions.

Appendix

The source code for this project can be found on Github

References

SimPy