Chris Hipple
Project 1 Design Document
10/17/2016
W18 MIDS Python Bridge Course

Goals and Purpose of the Project:

This program will simulate a production line in which a variety of components are produced sequentially. Products are batched into production orders which tell the production line how many of which type of product to build. Each of the different product types take a different amount of processing time on each station, and some products will be built significantly more than others. This is a simulation of the classic "high variance, low volume" manufacturing scenario. The other main manufacturing paradigm is "high volume, low variance." When building products with a wide variance, the sequencing becomes very important for line balancing. For example, a product which takes a very long time to process on station 2 should be produced after a product which takes a long time to process on station 3. This way, none of the stations are waiting for each other.

The output of the simulation will be a total running time for the production line, an overall line efficiency, and the efficiency for each station. The program will also be capable of determining the most efficient sequence of orders.

General Design and Flow of the Program:

The main object in this program is the production line. The production line is made up of stations, which are consecutive objects for processing products. Stations are identified by their station number. Products are grouped into production orders, which are a batch of the same type of product. Production Orders are identified by an order number. Each individual product is identified by it's product type and a unique serial number. The product order sequence is simply the sequence in which the production orders are processed on the line.

The general flow of the program will be that when started with a defined production order sequence, with defined production orders. The line will pick the next order, and begin to process it at the first station. At this station, the specific products within the order will be assigned, as the serial numbers are generated. Once a product is finished being processed at this station, it is passed onto the next station. When all the products from this order are completed at the first station, it will grab the next order and begin processing that one. This cycle continues until all the orders have entered the line. When a product is finished on the last station, it's marked as completed, and removed from the line.

Each station is only capable of processing one part at a time, and if a station finishes, but the next station is still working, they will not be able to pass it on. This can create a traffic jam that will back up the process, hurting efficiency.

In order to determine the most efficient production order sequence, the line will simulate running through each of the sequences, calculating an efficiency and an overall completion time

for each one. When calculating efficiencies, this simulation assumes that there's an operator at each station, and they do not move between stations. If a product is currently there, they are working 100%. If not, they are idling. Having fewer operators who move between stations focusing the effort on where it's needed most is a method for improving efficiency at the cost of overall completion time. However, those considerations will be out of scope for this current simulation.

Classes:

Production Line

Methods

- Process Order Sequence
  - Hands the first piece of the first production order to the first station and then continues to loop until all products are completed.

Attributes

- Stations
- Current time
  - Simulation will be handled by a master clock running on the production line and the orders will hand off their parts to the stations, one at a time, which will set a completion time for that process based on the

Station

Methods

- Receive Product
  - Looks at the cycle time for that product for that station and sets a completion time. Sets CurrentlyActive for the station to be true
- Process Product
  - Checks to see if product is finished (compares current time against Current Product Finish Time), if so. Writes process results to the product and tries to move it forward in position. If successful, sets Current Product to None, letting the line know that it is ready to receive another product. Either way, sets CurrentlyActive to false.
- Calculate Efficiency
  - Divides the station Utilization Counter by the total time to calculate an efficiency.

Attributes

- Station Utilization Counter
  - Sum of CurrentlyActive counts over time.
- Current Product Finish Time
  - Tracks when the current product will be finished

- Current Product
  - The serial number of the product currently being produced on the station.
- Currently Active?
  - Tracks when the station is being utilized and when it is not, for efficiency calculations

First Station *Subclass of Station*

Methods

- Process Product
  - As it begins processing an order, the first station creates the products to fill that order and generates their serial number.

Last Station *Subclass of Station*

Methods

- Process Product

  When the last station finishes processing a product, it removes it from the line and marks it off as complete. In a real process, it would be then moved to logistics for shipment to customer.

Product

Attributes

- Process Results
  - Simulated random results to show that the product was run.
- Cycle Times
  - Station by Station times required for processing of this product type.

Production Order

Attributes

- Part Number
- Quantity
- Priority Level
  - Allows for rush orders to be moved to front of que.
- Specific Parts Contained
- A list of the actual instances of the Product class which are associated with this order.

Order Sequence *This is a less useful object, perhaps its not needed...*

Attributes

- Orders