Project II: Production Dashboard Integration

Authors: Eduardo Martinez, Fernanda Mena, Brandon Avalos.

This project extends a discrete-event simulation system by integrating it with a dynamic web-based dashboard. The goal is to visualize key production metrics generated from simulation runs, facilitating a better understanding of factory performance in real-time-like environments.

Project Overview

The system consists of two separate Git repositories:

1. `SG2_Team3`

This repository contains the simulation and data generation code.

- `plotting.py`: Main script that runs 365 simulations representing a whole year of production,, aggregates results, and exports them as a structured JSON file.

2. `SG2_Tean3_MidTermII`

This repository contains the interactive dashboard for visualizing simulation outputs.

- `index.html`: Central dashboard file to visualize production statistics.
- `data/`: Folder that receives the JSON file generated by `plotting.py`.

Key Features

Simulation Features

- **Six Workstations**: Simulate sequential operations with failure events, bottlenecks, and maintenance.
- **Accident & Rejection Modeling**: Simulates accidents (1%) and product rejections (5%) for realistic production insights.
- **Statistical Metrics**:
  - Production output
  - Rejection rate
  - Supplier occupancy
  - Workstation delays, fixing time, bottlenecks, and downtime

Dashboard Visualization

- Multiple views: Daily, Weekly, Monthly, etc.
- Interactive metrics:

    - Rechazo (% rejection)
    - Ocupación (occupancy)
    - Retrasos (delays)
    - Accidentes (accidents)
- Pie and bar charts generated using D3.js or similar libraries.

File Structure

```
project/
├── SG2_Team3/
│   └── plotting.py              # Simulation runner and JSON generator
└── SG2_Tean3_MidTermII/
    ├── data/                    # Output JSON goes here
    ├── index.html               # Dashboard UI
```

How to Run

1. **Clone Both Repositories**:

```
git clone https://github.com/EduardoMV/SG2_Team3.git
git clone https://github.com/Fernandamenass/SG2_Tean3_MidTermII.git
```

2. **Run Simulation Script**:

```
cd SG2_Team3
python plotting.py
```

This generates a JSON file inside `SG2_Tean3_MidTermII/data`.

3. **Launch Dashboard**:
   Open `index.html` from `SG2_Tean3_MidTermII` in any web browser (Live Server recommended).

Technical Notes

- Built with **Python** (SimPy for simulation logic).
- Frontend: **HTML**, **CSS**, **JavaScript**, and **D3.js**.
- Modular: JSON decouples the simulation from the UI.
- Data granularity: Daily, Weekly, Monthly, Quarterly, Yearly.

Simulation Configuration (Customizable Parameters)

You can adjust how the simulation behaves by modifying the `__init__` method of the `ManufacturingFacility` class in `EUMV_FMS.py`. Below are the available parameters and what each controls:

```
self.stations = [simpy.Resource(env, capacity=1) for _ in range(6)]
```

- Defines the number of processing slots per station (default: one product at a time).

```
self.bins = [25, 25, 25, 25, 25, 25]
```

- Represents the material capacity at each station before needing resupply. Lower values increase supplier demand.

```
self.suppliers = simpy.Resource(env, capacity=3)
```

- Number of suppliers available to refill bins. More suppliers reduce supply delays.

```
self.failure_probs = [0.02, 0.01, 0.05, 0.15, 0.07, 0.06]
```

- Probability of failure at each station. Increasing values simulate less reliable equipment or harsher conditions.

```
simulation_results = run_all_runs(num_runs=365, simulation_time=5000)
```

"num_runs=365" means the simulation will run for 365 days.

You can increase or reduce this number to simulate a different time period (e.g., 30 for a month, 90 for a quarter).

simulation_time=5000 controls how long each individual day lasts in simulation time units. This can also be adjusted to simulate different production intensities.

Other internal counters are automatically calculated, such as:

- `total_production`: number of accepted products.
- `rejected_products`: total defective products at the end of the line.
- `supplier_busy_time`: time suppliers are occupied across the run.
- `last_product_time`: used for delay tracking between units.

These parameters can be tuned to test different manufacturing scenarios or simulate production under stress conditions.

Acknowledgments

This is part of an academic project in Discrete Event Simulation. Developed with the collaboration of SG2 Team 3. Visual tools created specifically for improving simulation accessibility and communication.