Infraestructura Computacional Caso 1 informe

Alberto Mario Pertuz Noriega 202025856

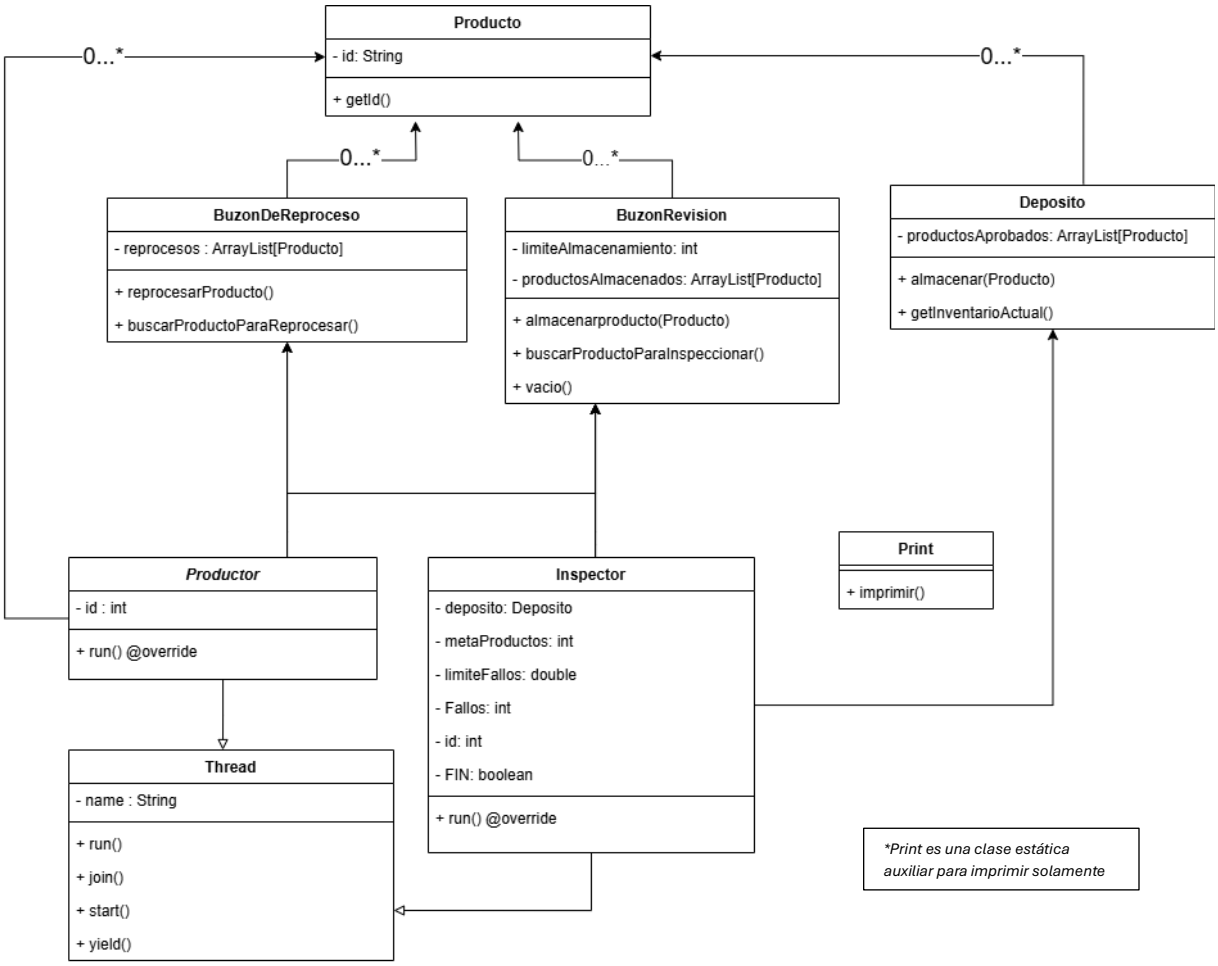
Andres Botero Ruiz 202223503

Andres Felipe Charry Camacho 202214507

Introducción

Este documento presenta el diseño, funcionamiento y validación del sistema desarrollado para la simulación de una línea de producción en una empresa de ensamblaje. La implementación está basada en Java y utiliza threads para representar los operarios involucrados en la producción y control de calidad de los productos.

UML



Descripción de clases

La línea de producción se compone de los siguientes elementos:

Productores: Encargados de generar nuevos productos o reprocesar productos defectuosos.

Inspectores: Inspecciona los productos y determina si son aprobados o deben ser reprocesados.

Depósito: Almacena los productos aprobados.

Buzón de revisión: Espacio limitado donde los productores depositan los productos antes de ser revisados.

Buzón de reproceso: Almacena productos defectuosos para su reproceso por los productores.

Funcionamiento general del Programa

1. Los productores generan productos nuevos o reprocesan productos del buzón de reproceso (prioridad sobre generación).
2. Si el buzón de revisión está lleno, los productores esperan antes de generar o reprocesar.
3. El equipo de calidad revisa los productos en el buzón de revisión:

    -Si el producto es aprobado, se envía al depósito.

    -Si es rechazado, se envía al buzón de reproceso.
4. El equipo de calidad usa un número aleatorio para determinar si un producto es defectuoso (múltiplo de 7 indica fallo).
5. Cuando se alcanza el número máximo de fallos permitidos, todos los productos son aprobados.
6. Una vez alcanzada la cantidad total de productos a procesar, se genera un producto "FIN" para indicar el fin del programa.
7. Los productores, al recibir el mensaje "FIN" en el buzón de reproceso, terminan su ejecución.

Sincronización y Control de Concurrencia

1. Sincronización entre **BuzonDeReproceso** e **Inspector**

El *Inspector* utiliza el método *reprocesarProducto* de *BuzonDeReproceso* para añadir productos rechazados al buzón de reprocesos. Este método está sincronizado para asegurar que solo un hilo pueda añadir un producto al mismo tiempo.

2. Sincronización entre **BuzonDeRevision** e **Inspector**

El *Inspector* utiliza el método *buscarProductoParaInspeccionar* de *BuzonDeRevision* para obtener productos para inspeccionar. Este método está sincronizado para asegurar que solo un hilo pueda retirar un producto al mismo tiempo. Además, el *Inspector* utiliza el método *vacio*() para verificar si el buzón de revisión está vacío. Este método también está sincronizado para asegurar que la verificación sea precisa.

3. Sincronización entre **BuzonDeRevision** y **Productor**

El *Productor* utiliza el método *almacenarProducto* de *BuzonDeRevision* para añadir productos al buzón de revisión. Este método está sincronizado y utiliza *wait* para manejar la espera pasiva cuando el buzón está lleno.

4. Sincronización entre **BuzonDeReproceso** y **Productor**

El *Productor* utiliza el método *buscarProductoParaReprocesar* de *BuzonDeReproceso* para consultar si hay productos rechazados para revisar. Este método está sincronizado para asegurar que solo un hilo pueda retirar un producto al mismo tiempo.

5. Sincronización entre **Deposito** e **Inspector**

El *Inspector* utiliza el método *almacenanar* de *Deposito* para añadir productos aprobados al depósito. Este método está sincronizado para asegurar que solo un hilo pueda añadir un producto al mismo tiempo. Además, el *Inspector* utiliza el método *getInventarioActual* para obtener el número actual de productos en el depósito. Este

método también está sincronizado para asegurar que la lectura del inventario sea precisa.

Funcionamiento Global del Sistema

1. **Productor:** El *Productor* genera nuevos productos y los almacena en el *BuzonDeRevision*. Si hay productos en el *BuzonDeReproceso*, los reprocesa y los vuelve a enviar al *BuzonDeRevision*.

2. **Inspector**: El *Inspector* obtiene productos del *BuzonDeRevision* para inspeccionarlos. Si un producto es rechazado, se envía al *BuzonDeReproceso*. Si es aprobado, se almacena en el *Deposito*. El *Inspector* sigue inspeccionando productos hasta que se alcanza la meta de productos aprobados.

3. **BuzonDeReproceso**: Almacena productos rechazados por el *Inspector* y los proporciona al *Productor* para su reproceso.

4. **BuzonDeRevision**: Almacena productos generados por el *Productor* y proporciona productos al *Inspector* para su inspección.

5. **Deposito**: Almacena productos aprobados por el *Inspector*.

Validación del Programa

El programa se valida verificando que el numero de productos en el inventario sean mayores o iguales a la meta de productos. El método *testDynamicParameteres()* usa valores aleatorios para verificar que el programa funciona bien con diferentes cuellos de botella (muchos trabajadores y un tamaño de buzón pequeño, o al revés).

```
Test passed -> Inventario actual:    27, Meta de productos:    12
Test passed -> Inventario actual:    28, Meta de productos:    12
Test passed -> Inventario actual:    26, Meta de productos:    12
Test passed -> Inventario actual:    52, Meta de productos:    28
Test passed -> Inventario actual:    42, Meta de productos:    28
Test passed -> Inventario actual:    42, Meta de productos:    28
Test passed -> Inventario actual:    45, Meta de productos:    28
Test passed -> Inventario actual:    45, Meta de productos:    28
Test passed -> Inventario actual:    45, Meta de productos:    28
Test passed -> Inventario actual:    39, Meta de productos:    12
Test passed -> Inventario actual:    42, Meta de productos:    12
Test passed -> Inventario actual:    68, Meta de productos:    12
```