Yhojan Alejandro Franco Pardo - 202225577

Luis Hernán Bobadilla Moreno - 202223854

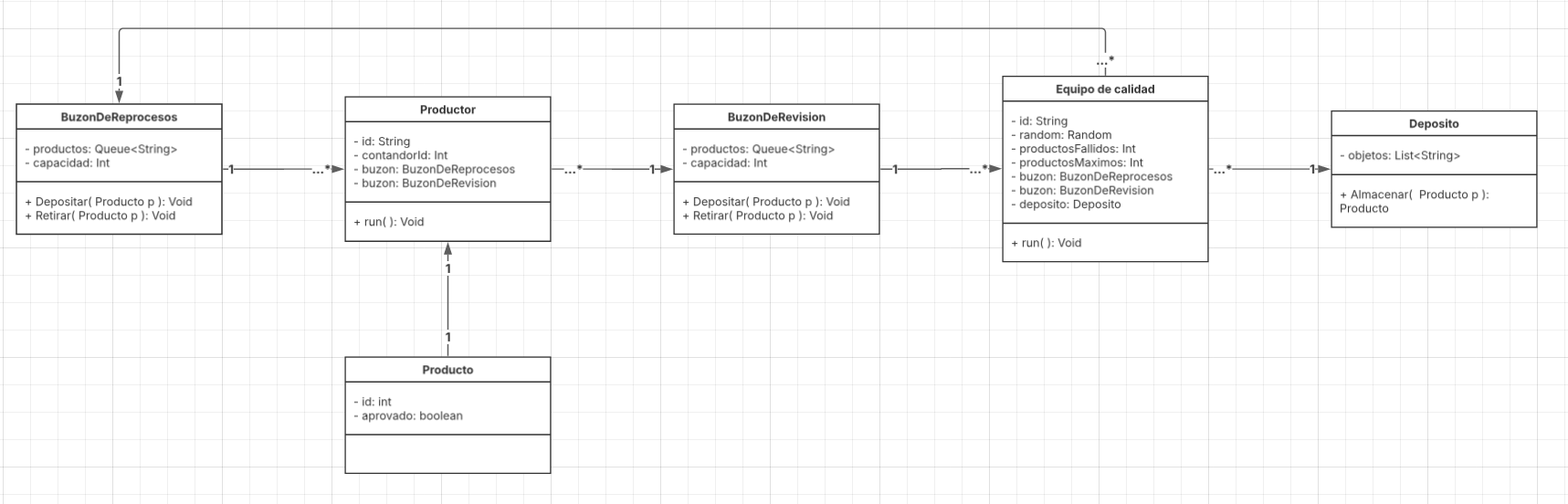
Maria Alejandra Sanabria Salazar - 202326786

**DISEÑO DEL CASO 1 GRUPO BAQ**

El programa implementa un sistema de producción donde los productos son generados por productores, revisados por un equipo de calidad y finalmente almacenados en un depósito si cumplen con los estándares requeridos. El diseño del sistema se basa en el uso de hilos para la concurrencia y sincronización a través de buzones compartidos.

### **Diagrama de Clases:**

El siguiente diagrama de clases representa la estructura y relaciones entre los objetos principales del sistema:



#### **Clase Producto**

Representa un producto en la línea de producción con un identificador (‘id’) y un estado de aprobación.

* **Atributos:**
  + id: int - Identificador único del producto.
  + aprobado: boolean - Indica si el producto pasó la revisión de calidad.

#### **Clase Productor**

Se encarga de generar nuevos productos o tomar productos rechazados para reprocesarlos.

* **Atributos:**
  + id: int- Identificador del productor.
  + contadorId: int - Contador global de productos generados.
  + buzonReproceso: BuzonDeReprocesos - Buzón compartido donde se depositan productos rechazados.
  + buzonRevision: BuzonDeRevision - Buzón donde se envían productos para ser revisados.
* **Métodos:**
  + run(): void - Ciclo de generación y envío de productos a la revisión.

#### **Clase BuzonDeRevision y BuzonDeReprocesos**

Nota: Tanto el Buzón de Revisión como el Buzón de Reprocesos son instancias de la misma clase Buzon, pero cumplen roles distintos: el primero almacena productos para ser evaluados por el Equipo de Calidad, mientras que el segundo guarda productos rechazados para ser reprocesados por los Productores.

Son estructuras de almacenamiento intermedias donde los productos esperan para ser procesados.

* **Atributos:**
  + productos: Queue<String> - Cola de productos pendientes.
  + capacidad: int - Tamaño máximo del buzón.
* **Métodos:**
  + depositar(Producto p): void - Agrega un producto a la cola.
  + retirar(): Producto - Retira un producto de la cola.

#### **Clase EquipoDeCalidad**

Se encarga de evaluar los productos en el buzón de revisión y decidir si se aprueban o se reenvían al buzón de reproceso.

* **Atributos:**
  + id: int - Identificador del equipo de calidad.
  + productosFallidos: int - Contador de productos defectuosos.
  + productosMaximos: int - Límite de productos antes de finalizar la produ cción.
  + buzonReproceso: BuzonDeReprocesos - Buzón donde se depositan productos fallidos.
  + buzonRevision: BuzonDeRevision - Buzón desde donde se retiran los productos para evaluar.
  + deposito: Deposito - Depósito donde se almacenan los productos aprobados.
* **Métodos:**
  + run(): void - Ciclo de evaluación de productos y almacenamiento de los aprobados.

#### **Clase Deposito**

Es el almacenamiento final de los productos aprobados.

* **Atributos:**
  + objetos: List<String> - Lista de productos aprobados.
* **Métodos:**
  + almacenar(Producto p): Producto - Agrega un producto aprobado al depósito.

**Funcionamiento del Programa**

El programa implementa una **línea de producción concurrente** con múltiples hilos que interactúan mediante **buzones compartidos**, donde los objetos Productor, EquipoCalidad y Deposito trabajan en conjunto. La comunicación y sincronización entre hilos se gestiona con **monitores (synchronized), comunicación mediante wait/notifyAll y visibilidad de memoria con variables volátiles**.

La ejecución sigue un ciclo de vida bien definido:

1. **Productores** generan productos o recuperan productos fallidos para reprocesarlos.
2. **Equipos de calidad** validan los productos, almacenando los aprobados y enviando los rechazados a reproceso.
3. **El sistema se detiene** cuando se alcanzan los criterios de finalización.

## **Interacciones Entre Objetos y Sincronización**

Cada par de objetos en el sistema se sincroniza a través de estructuras de datos compartidas que implementan **bloqueos y señales** para gestionar el acceso concurrente.

### **1. Productor ⬌ Buzón de Revisión**

* El Productor genera productos nuevos y los coloca en BuzonDeRevision.
* BuzonDeRevision tiene una **cola con capacidad limitada**, por lo que si está lleno, el Productor **bloquea su ejecución (wait)** hasta que haya espacio.
* Cuando el EquipoCalidad retira un producto del buzón, se notifica al Productor con **notifyAll()** para que continúe su ejecución.

**Evita condiciones de carrera** en la inserción de productos en la cola.

**Evita el desperdicio de CPU**, ya que el Productor espera de manera eficiente en caso de que el buzón esté lleno.

### **2. Equipo de Calidad ⬌ Buzón de Revisión**

* EquipoCalidad toma productos de BuzonDeRevision para su inspección.
* Si el buzón está vacío, el EquipoCalidad **bloquea su ejecución (wait)** hasta que haya productos disponibles.
* Luego, verifica aleatoriamente si un producto es aprobado o rechazado.

**Sincronización efectiva:** EquipoCalidad espera sin consumir CPU cuando el buzón está vacío.

**Evita desbordamientos** asegurando que el Productor solo agregue cuando haya espacio.

### **3. Equipo de Calidad ⬌ Buzón de Reproceso**

* Si un producto es rechazado, se envía al BuzonDeReprocesos.
* Productor revisa continuamente si hay productos en BuzonDeReprocesos y los retira para reprocesarlos.
* Se utilizan **métodos sincronizados para garantizar la atomicidad** al acceder al buzón.

**Evita inconsistencias**, asegurando que los productos sean correctamente reprocesados.

**Eliminación de bloqueos mutuos**, permitiendo la reutilización eficiente de productos.

### **4. Equipo de Calidad ⬌ Depósito**

* Si un producto es aprobado, se envía al Deposito, donde se almacena de manera definitiva.
* Cuando el Deposito alcanza el número requerido de productos aprobados, la ejecución se detiene.
* Para detener los hilos de manera segura, se envía una **señal de terminación (Producto -1)** y se modifica volatile boolean fin = true, asegurando **visibilidad de memoria**.

**Evita la pérdida de productos aprobados** mediante un almacenamiento ordenado.

**Finalización segura de los hilos**, asegurando que no haya ejecuciones pendientes después de alcanzar el objetivo.

# **Funcionamiento Global del Sistema**

1. **Inicio**
   1. Se crean múltiples hilos Productor y EquipoCalidad.
   2. Los Productor comienzan a generar productos, llenando BuzonDeRevision.
2. **Producción y Validación Concurrente**
   1. EquipoCalidad consume productos y los evalúa.
   2. Los productos aprobados se almacenan en Deposito, los fallidos se reenvían a BuzonDeReprocesos.
   3. Productor revisa BuzonDeReprocesos, retirando productos fallidos y reenviándolos a BuzonDeRevision.
3. **Finalización**
   1. Cuando Deposito alcanza el número de productos aprobados requerido, EquipoCalidad envía la señal de terminación.
   2. Se notifica a todos los hilos con notifyAll(), asegurando que no queden hilos bloqueados.
   3. Todos los hilos finalizan de forma segura.

# **Validación del Sistema**

Para garantizar que el sistema funciona correctamente bajo concurrencia, se validan los siguientes aspectos:

**Consistencia de Datos**

* Se asegura que ningún producto sea procesado dos veces o se pierda durante la ejecución.

**Sincronización Correcta**

* Se validan las esperas activas en wait() y las notificaciones en notifyAll() para evitar bloqueos innecesarios.

**Manejo de la Concurrencia**

* Se verifica que no haya **condiciones de carrera** en el acceso a los buzones.

**Correcta Finalización**

* Todos los hilos terminan su ejecución cuando se alcanzan las condiciones de parada, sin dejar procesos en estado inactivo.

**Eficiencia y Paralelismo**

* Se verifica que la producción, validación y almacenamiento se ejecuten en paralelo sin esperas innecesarias.

### **Pruebas:**

1. **Con pocos productores y productos:**
   1. Verificar que FIN se envía una sola vez.
   2. Que los productores se detienen tras FIN.
2. **Con varios productores:**
   1. Que todos trabajan en paralelo.
   2. Que se respeta la capacidad del buzón.
3. **Límite de fallos (10%):**
   1. Al llegar al máximo de fallos, todos los siguientes son aprobados.

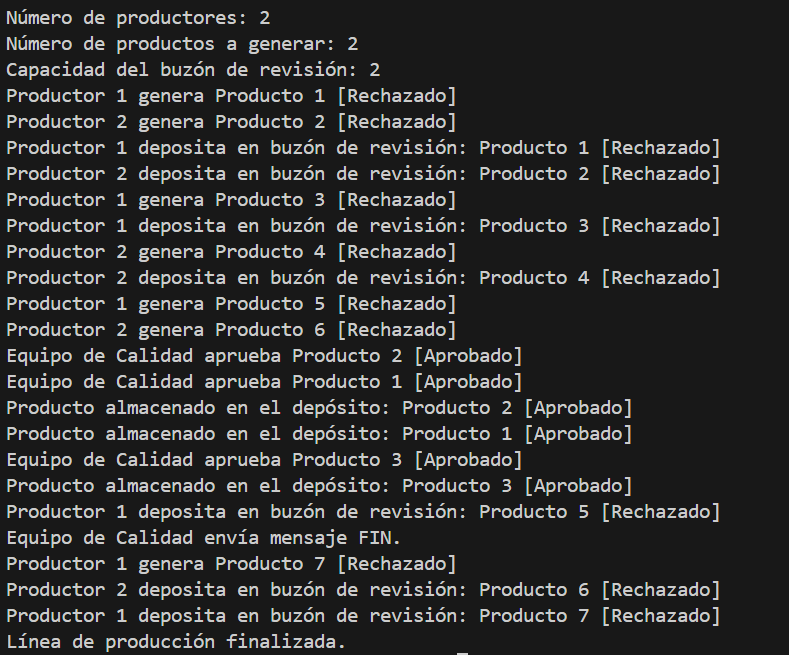
### Resultado de Prueba Clave:

Con:

* **Productores:** 2
* **Productos para generar:** 2
* **Capacidad:** 2

Se obtuvo:

* Un solo FIN.
* Productores detenidos tras FIN.
* Número exacto de aprobados.



Con:

* **Productores:** 3
* **Productos para generar:** 5
* **Capacidad:** 2

Se obtuvo:

* Un solo FIN.
* Productores detenidos tras FIN.
* Número exacto de aprobados.

