Laura Sofia Murcia – 202123099

Mónica González – 202116872

Camilo Quimbay - 202313861

**Caso 1 – Infraestructura Computacional**

**Descripción general:**

El problema general que se resuelve dentro de este caso consiste en la creación y envío de productos partiendo de un productor y llegando a un distribuidor. Este proceso se modela haciendo uso de Threads en Java con los que se manejan 4 productos, 2 operarios internos y 4 distribuidores que interactúan con 2 depósitos, una cinta y productos de 2 tipos diferentes.

A continuación, se explicará cada parte del proceso dividiéndolo en 3 momentos:

1. Del productor al depósito de producción
2. Del depósito de producción al depósito de distribución
3. Del depósito de distribución al distribuidor

**Parte 1.** Proceso del productor al depósito de producción

En esta primera parte, el productor a partir del número de productos definidos que tiene que generar empieza a generar un producto que coincida con su mismo tipo, es decir tipo A o tipo B. Una vez ha terminado de crear la cantidad de productos necesarios, crea el producto con el tipo FIN\_A o FIN\_B. Después de esto, hace un llamado al método guardarProducto() sobre el depósito de producción. Este método permite la sincronización sobre el depósito en donde si no hay espacio para guardar el producto, la thread del productor se queda dormida. En caso de que el espacio si esté disponible, se guarda el producto en el depósito y se actualiza el tamaño disponible del depósito. Adicionalmente, para evitar posibles problemas con que los productos que indican el fin de la ejecución del programa lleguen antes que los demás, se tiene un método aparte para guardar estos dentro del depósito, en donde solo pueden ser almacenados una vez en el depósito de producción no se encuentren productos tipo A o B.

**Parte 2.** Proceso del depósito de producción al depósito de distribución

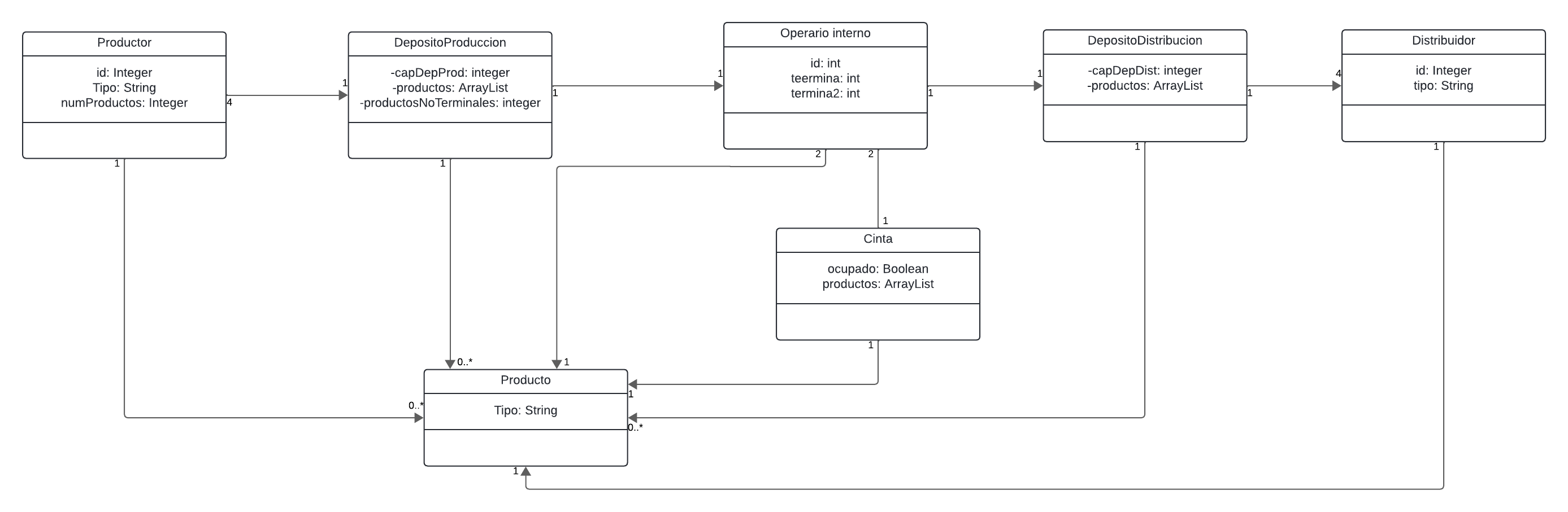
La siguiente parte del proceso consiste en llevar los productos del depósito de producción al depósito de distribución. Para esto, se tienen en cuenta 2 threads adicionales que corresponden a los operarios internos. Estos operarios funcionan con un ID el cual determina la dirección en la que trabajan. El operario con el ID 1, realiza el trabajo de llevar los productos del depósito de producción a la cinta y el operario con el ID 2 los lleva de la cinta al depósito de distribución.

El primer operario implementa espera semi-activa para revisar si hay algún producto en el depósito y si la cinta está ocupada. Cuando se tengan productos y espacio en la cinta, los recoge del depósito y los mueve. De forma similar, el segundo operario espera a que se tenga espacio en el depósito de distribución y se tenga algún producto dentro de la cinta. Cuando estas dos cosas se cumplan, saca el producto de la cinta y lo guarda en el depósito de distribución. Debido a que los operarios internos no conocen cuantos productos se generan en total, para parar se tienen 2 variables que contabilizan cuantos productos de terminación han pasado. Si se han pasado 4 productos de terminación los operarios acaban su operación.

**Parte 3.** Proceso del depósito de distribución al distribuidor

Finalmente, los distribuidores consumen los productos que se encuentran en el depósito de distribución. Para esto, se tiene que mientras no encuentren un producto tipo FIN deben esperar pasivamente a sacar los productos del depósito. Al momento de sacar un producto del depósito se verifica que el depósito no esté vacío y que exista un producto que corresponda con el tipo del distribuidor. Así, los distribuidores duermen a menos que puedan consumir el producto adecuado. Adicionalmente, se tiene una condición extra y es que se consuman todos los productos que no sean de FIN primero. Esto se hace para evitar que termine la ejecución de los distribuidores y se puedan quedar productos sin procesar.

**Diagrama de clases:**

****

Tenemos las Clases

**Productor:**

Que tiene como atributos id (identificador único del productor) que es un entero, tipo que indica el tipo de productor, numProductos: que es un entero e indica la cantidad de productos que se generaron. El productor está asociado con la clase DepositoProduccion, es una relación 4 a uno ya que tenemos cuatro tipos de productores que producen un producto a la vez para almacenarlo en el depósito; También tenemos la relación a Producto de 0 a muchos ya que el productor puede o no producir varios productos y cada producto es hecho por un productor.

**DepósitoProducción:**

Que tiene como atributos capDepProd entero que indica la capacidad del depósito de producción, productos que es un arraylist con los productos almacenados en el depósito, productosNoTerminales que es un entero que indica la cantidad de productos que no han sido enviados. DepositoProduccion está relacionado con OperarioInterno que es una relación de uno a uno ya que un operario mueve un producto a la vez; También tenemos la relación con Producto que es de 0 a muchos ya que en un depósito se pueden almacenar o no varios productos.

**Producto:**

Tiene el atributo de tipo string que indica el tipo de productos (A y B) y tiene relaciones con todas las demás clases (se especifican el tipo de relación en las clases restantes)

**OperarioInterno**

Tiene como atributos id (identificador único del operario), termina que es un entero y representa el fin de llevar el producto del depósito de producción a la cinta y termina2 que representa la finalización de llevarlo al depósito de distribución. OperarioInterno está relacionado con Producto y su relación es de 2 a uno porque tenemos 2 operarios internos que llevan un producto, su otra relación es con Cinta de 2 a uno ya que existen dos operaciones que utilizan una cinta y por último tiene una relación con DepositoDistribución que es de uno a uno.

**Cinta**

Tiene como atributos ocupados que es un booleano que indica si la cinta está ocupada o no y el producto que es un ArrayList que contiene los productos recibidos en la cinta, Cinta está relacionada con Producto y es una relación uno a uno.

**DepositoDistribución**

Tiene como atributos capDeptDist es un entero que indica la capacidad del depósito de distribución y productos que cuenta con un ArrayList de los productos recibidos; DepositoDistribucion tiene relación con Distribuidor y es una relación de uno a 4.

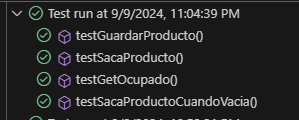
**Distribuidor**

Tiene como atributos un identificador único y un tipo; Distribuidor tiene relaciones con producto y es de uno a uno ya que cada distribuidor recibe un producto de terminación.

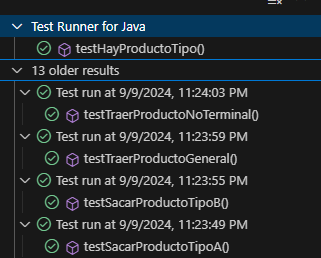
**Pruebas de funcionamiento:**

**Caso normal pruebas para DepositoProduccionTest**

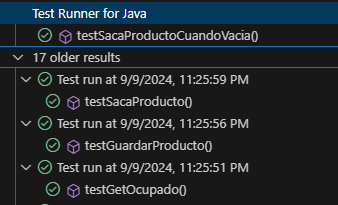
Se proponen casos base para los metodos de DepositoProducción:



También para casos base de DepositoDistribución:



Y para los casos base de la Cinta:

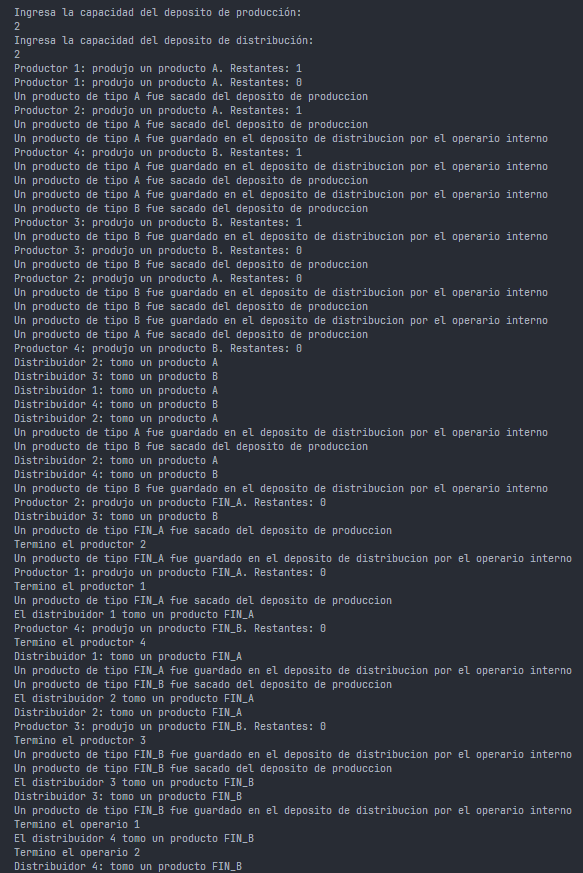


Se observa que en general las pruebas salen exitosas, aunque algunas toman un poco tiempo.

**Caso para producir 2 productos, con capacidad de depósitos de 2:**

En este caso vemos como nunca se añaden más de 2 productos a los depósitos y nunca se sacan más de 2 productos. Además, vemos como los productores producen de a 2 productos y después cuando generan el producto FIN de su tipo terminan su operación.

Por otro lado, los operarios terminan su ejecución hasta que pasan todos los productos de la cinta. Finalmente, los proveedores obtienen el producto FIN y su ejecución acaba.



**Caso donde los valores son 0 o menores:**

El programa no se ejecuta si los datos no tienen sentido.

