Venta masiva - depósito, pedidos y envíos

Construcción de Interfaces de Usuario.
TPI – General Belgrano – versión del 06/10/2017

Se pide desarrollar el modelo de parte de la operación de Saleizon, una empresa de venta masiva online. Saleizon vendió solamente libros durante mucho tiempo, ahora también vende vinos.

Los clientes registran pedidos desde la Web, para cada pedido se genera un envío que incluye la mercadería indicada en el pedido.

1. Depósito

Se trata de construir un modelo del depósito de Saleizon. En este depósito hay ejemplares de libros, y botellas de vino. Vamos a llamar **elementos** del depósito a ejemplares y botellas. Cada elemento corresponde a un **producto**, el libro para el ejemplar, el vino para la botella. Cada elemento tiene un número de inventario que es único dentro del depósito.

De cada libro nos interesa: título, autores, alto, ancho, espesor (todos en cm, ya veremos para qué se usan), peso, costo, precio de venta. De cada vino nos interesa: bodega, cepas (Malbec, Cabernet, Merlot, etc., pueden ser una o más), peso, costo, precio de venta.

Un ejemplo: supongamos que en el depósito hay cuatro ejemplares de Çien años de soledad", y tres botellas de Trapiche Malbec. En este caso, tenemos

- dos *productos*, el libro "Cien años de soledad", y el vino "Trapiche Malbec"
- siete *elementos*, los cuatro ejemplares y las tres botellas.

El depósito de Saleizon tiene varios **sectores**. Hay **sectores simples**, que es donde están los elementos, y **sectores compuestos**, que tienen otros sectores.

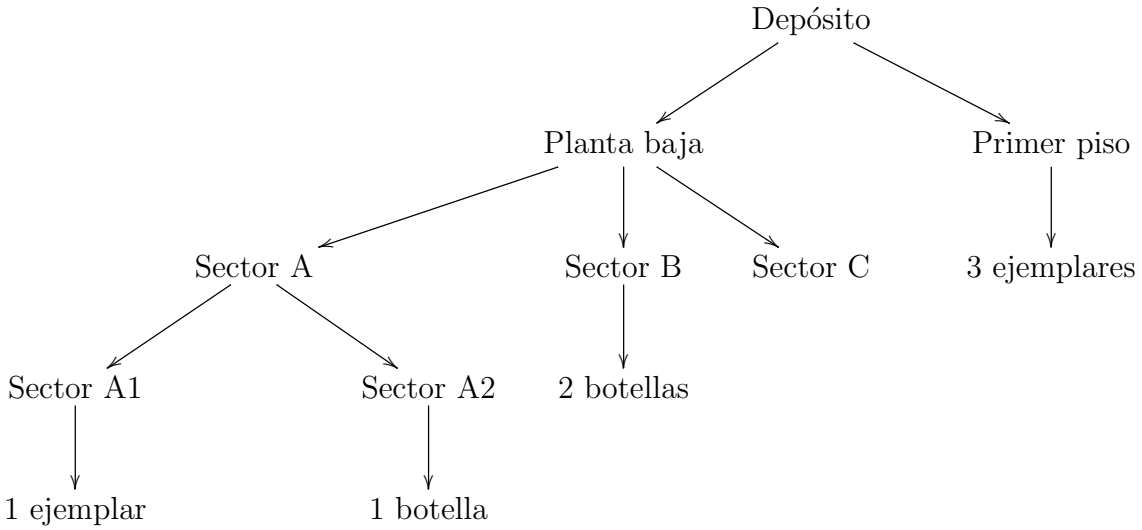
Para dar un ejemplo, describamos una posible configuración del depósito, en el que vamos a ubicar los siete elementos que describimos recién. Imaginemos un depósito con planta baja y primer piso.

- El *primer piso* es un sector simple, ahí están tres de los ejemplares de "Cien años de soledad".
- La *planta baja* es un sector compuesto, tiene los sectores A, B y C.

  El sector A es un sector compuesto, tiene los sectores A1 y A2, los dos simples. En el A1 está el otro ejemplar de "Cien años de soledad", en el A2 está una de las botellas de Trapiche Malbec.

  El sector B es simple, ahí están las otras dos botellas. El sector C, que también es simple, está vacío.

La organización que describimos recién se puede graficar en este árbol:



Remarcamos lo de *posible*: no sabemos cómo es la configuración del depósito, y puede cambiar con el tiempo (p.ej. agregando nuevos sectores). Hay que hacer un modelo que soporte sectores compuestos y sectores simples.

Requerimientos

1. agregar un ejemplar a un sector simple.
2. agregar una botella a un sector simple.
3. dado un autor, obtener el conjunto de libros de ese autor de los que hay al menos un ejemplar en el depósito.
4. dado un producto (libro o vino), obtener un conjunto con los elementos (ejemplares o botellas) de ese producto en el depósito.
5. dado un identificador, conocer en qué sector simple está el elemento con ese identificador. Si no está en ninguno, que lance excepción.
6. retirar un elemento de un sector simple. Si no está, lanzar excepción.
7. vaciar un sector simple, mandando todos sus elementos a otro sector simple que se pasa como parámetro.

Sugerencias

Para el modelo de sectores, conviene usar el pattern Composite, el que usamos para los productos fabricados en el ejercicio de mercadería.

Para el requerimiento 3, hace falta manejar por separado los ejemplares. Para los requerimientos 4 y 5, conviene manejar todos los elementos juntos. Para resolver esta dualidad, recomendamos que:

- cada sector maneje por separado sus ejemplares de libro y sus botellas. Pero que también tenga un mensaje que devuelve el conjunto con todos sus elementos, que es la unión de los dos conjuntos.
- los ejemplares tengan dos métodos, getProducto() y getLibro(), que miran al mismo atributo pero con distinto tipo (Producto y Libro respectivamente).

  Alternativa para esto (para gente osada): usar *generics* como vimos en el ejercicio de exámenes

2. Pedidos y envíos

En esta parte hay que agregar al modelo de la operatoria de Saleizon, los **pedidos** que hacen los clientes. Cada pedido incluye un conjunto de productos, donde puede haber libros y vinos mezclados. Para simplificar, vale establecer que en cada pedido se considera un elemento de cada producto, no más. De cada pedido nos va a interesar el cliente y la fecha de pedido.

Cuando la gente de expedición procesa un pedido, se crea un **envío** para el mismo. Al crearse un envío, se le asigna un identificador que sale de un numerador correlativo (al primer envío se le asigna el 1, al segundo el 2, etc.).

Cada envío incluye:

- un ejemplar de cada libro incluido en el pedido.
- una botella de cada vino incluido en el pedido.
- folios de papel de envolver, para los ejemplares.
- para cada botella, un envoltorio de film alveolar o "plástico con burbujas" (`https://es.wikipedia.org/wiki/Film_alveolar`).
- los materiales necesarios para cerrar los envoltorios, etiquetas para los ejemplares de libro, cintas para las botellas.

Al crearse un envío, queda en un conjunto de *envíos en tránsito*. Cuando se informa en el sistema que el envío se entregó, pasa a otro conjunto, de *envíos entregados*.

Requerimientos

1. crear un envío a partir de un pedido. Para esto, enviarle al objeto que represente al pedido el mensaje `crearEnvio()`, que devuelve el envío recién creado. Si no se puede crear el envío porque falta stock de algún producto (o sea, no hay ejemplares de un libro o botellas de un vino), lanzar excepción.
2. conocer el peso total de un envío. Cada folio de papel para envolver pesa 60 gramos, cada etiqueta 20 gramos, cada envoltorio de film alveolar 80 gramos, cada cinta 10 gramos.
3. saber, dado un cliente, si tiene o no envíos en tránsito.

Sugerencias

Crear un envío implica varias cosas.

Para empezar: crear (hacer `new`) el objeto que representa al envío. A ese objeto hay que asociarlo con el pedido (se recomienda que cada envío tenga una referencia al pedido que lo creó), y asignarle el número correlativo.

Para cada producto incluido en el pedido, hay que hacer tres cosas.

1. La primera es agregar al envío un **elemento** que corresponda a ese producto. Si el producto incluido en el pedido es un libro, entonces al envío hay que agregarle un ejemplar de ese libro. Si es un vino, entonces lo que hay que agregar es una botella. Una forma fácil para esto es: buscar un elemento del producto en el depósito, sacarlo del sector en donde está, y agregarlo al envío. Para cada una de estas cosas se puede usar uno de los requerimientos de la primera parte.

2. También hay que agregar el **envoltorio** para el elemento (ejemplar o botella) incorporado al envío.

   Si el elemento es un *ejemplar de libro*, se considera el perímetro a envolver, que se calcula así: (alto + espesor * 4) * (ancho * 2 + espesor * 3). Esto da un número en cm2 (centímetros cuadrados). P.ej. un libro de 20 x 12 y 2 cm de grosor, tiene un perímetro a envolver de 28 * 30 = 840 cm2.

   Cada folio de papel de envolver tiene 1500 cm2. Hay que poner la cantidad de folios necesaria para cubrir todo el perímetro. En el libro de ejemplo alcanza un folio. Si fuera un libro de 44 x 28 x 4, el perímetro sería 60 x 40 = 2400 cm2, se necesitan dos folios.

   Si el elemento es una *botella de vino*, se agrega un envoltorio de film alveolar.

3. Finalmente, hay que agregar el **material para cerrar** el envoltorio.

   Si el elemento es un *ejemplar de libro*, entonces corresponden tres etiquetas por folio que se puso.

   Si es una *botella de vino*, entonces van dos cintas si pesa 1 kg o menos, tres en caso contrario.

Para manejar la creación de un envío conviene tomar la idea del patrón Builder, como lo vimos para la creación de resoluciones. En ese caso, hay un paso que depende del tipo de pregunta: crear la respuesta. En la creación de un envío, hay dos pasos que dependen del tipo de producto: agregar el envoltorio, y agregar el material para cerrar.

Agregamos otras dos sugerencias

1. Para asignar al envío un número correlativo, crear un objeto Numerador que entienda el mensaje `darNumero()`. Usar Singleton para que haya un único Numerador.

2. En la creación de un envío hay que saber en qué depósito buscar los elementos. Para eso vale manejar el depósito como un Singleton ... por ahora.

3. Bonus

Hay varias extensiones que pueden ser interesantes.

3.1. Cuenta corriente

Agregar la capacidad de registrar los pagos de los clientes, y saber qué saldo tiene cada cliente. De lujo: también agregar el detalle de cuenta corriente, que es una lista de movimientos ordenada por fecha. Los movimientos son los pedidos y los pagos.

3.2. Impresión de etiquetas

Agregar en el envío un método `imprimirEtiqueta()`. Suponer que hay una impresora de etiquetas, accesible mediante un Singleton, que entiende los mensajes `saltarLinea()` `ponerTexto(String texto)`. Va un ejemplo de etiqueta

```
Juan Carlos Perillo

Saavedra 443
(7223) General Belgrano

8 artículos - peso total 23 kg
```

Para esto, agregar una clase Cliente con: nombre, dirección, código postal, y localidad. Esto nos da una primera idea sobre cómo interactuar con dispositivos.

Sugerencia: definir una interface de la impresora, y tener una implementación para test, que en lugar de imprimir, recolecte el texto en una lista de Strings. La idea es que después se incorpore el conector con la impresora real como otra clase que implementa la misma interface.

3.3. Faltantes en depósito

Obtener una lista de lo que hace falta incorporar para satisfacer los pedidos que no tienen envío. De cada producto, comparar la cantidad total en pedidos que no tienen preparado el envío, con el stock total en el depósito. Si el primero es mayor, entonces hay que incorporar la diferencia.

P.ej. si hay 8 ejemplares de “El señor de los anillos” en pedidos sin el envío armado, y 5 ejemplares en el depósito, entonces hay que incorporar 3.

Sugerencias

1. conviene vincular pedido y envío bidireccionalmente, o sea, que cada uno conozca al otro.

2. se puede implementar un Store que conozca el conjunto de productos.

3.4. Envíos multi-pedido

Hacer que un envío pueda concentrar más de un pedido. Para esto, agregarle a los pedidos la capacidad de entender el mensaje `agregarseAEnvio(Envio envio)`.

Todos los pedidos incluidos en un mismo envío deben ser del mismo cliente. Lanzar excepción si se trata de violar esta regla.

Agregar la capacidad de conocer el conjunto de pedidos para los que no se armó el envío correspondiente.

3.5. Ranking de clientes

Dadas dos fechas, obtener el ranking de clientes de acuerdo al importe total de los pedidos que hicieron en esas fechas, junto con ese importe.

Vale tener el conjunto de los clientes en un Store.

Nota para la entrega

Acompañar el código con dos diagramas de clases, uno para cada parte del enunciado.