Proyecto 1: Tienda en línea del supermercado Wolmar

UNAM, Facultad De Ciencias.

Curso: Modelado y Programación
Profesor.Rosa Victoria Villa Padilla
Ayudante. Arturo Lemus Pablo
Ayudante. Fernando López Balcazar
Ayudante. Itzel Azucena Delgado Díaz

Equipo: Prietos en aprietos

SanMartin Macias Juan Daniel No. Cuenta 318181637
López Diego Gabriela No. Cuenta 318243485
Rivera Zavala Javier Alejandro No. Cuenta 311288876

28 de Octubre de 2022.
Ciudad de México.

1. Patrón de diseño Strategy

Utilizamos el patrón Strategy para el requisito de múltiples idiomas en la tienda virtual.
Haciendo uso de la mayor ventaja de este patrón, que es el hacer al código más límpio para su mantenimiento, evitandonos estar considerando (en todas las partes donde se tenga que imprimir un mensaje) múltiples casos, usando un alto número de condicionales.
Con este patrón la agregación de un nuevo idioma a futuro quedaría muy sencillo, puesto que sólo sería ir agregando una clase correspondiente a ese nuevo idioma que implemente a la clase región, sin tener que modificar nada en la parte donde se mandan a llamar estos mensajes.

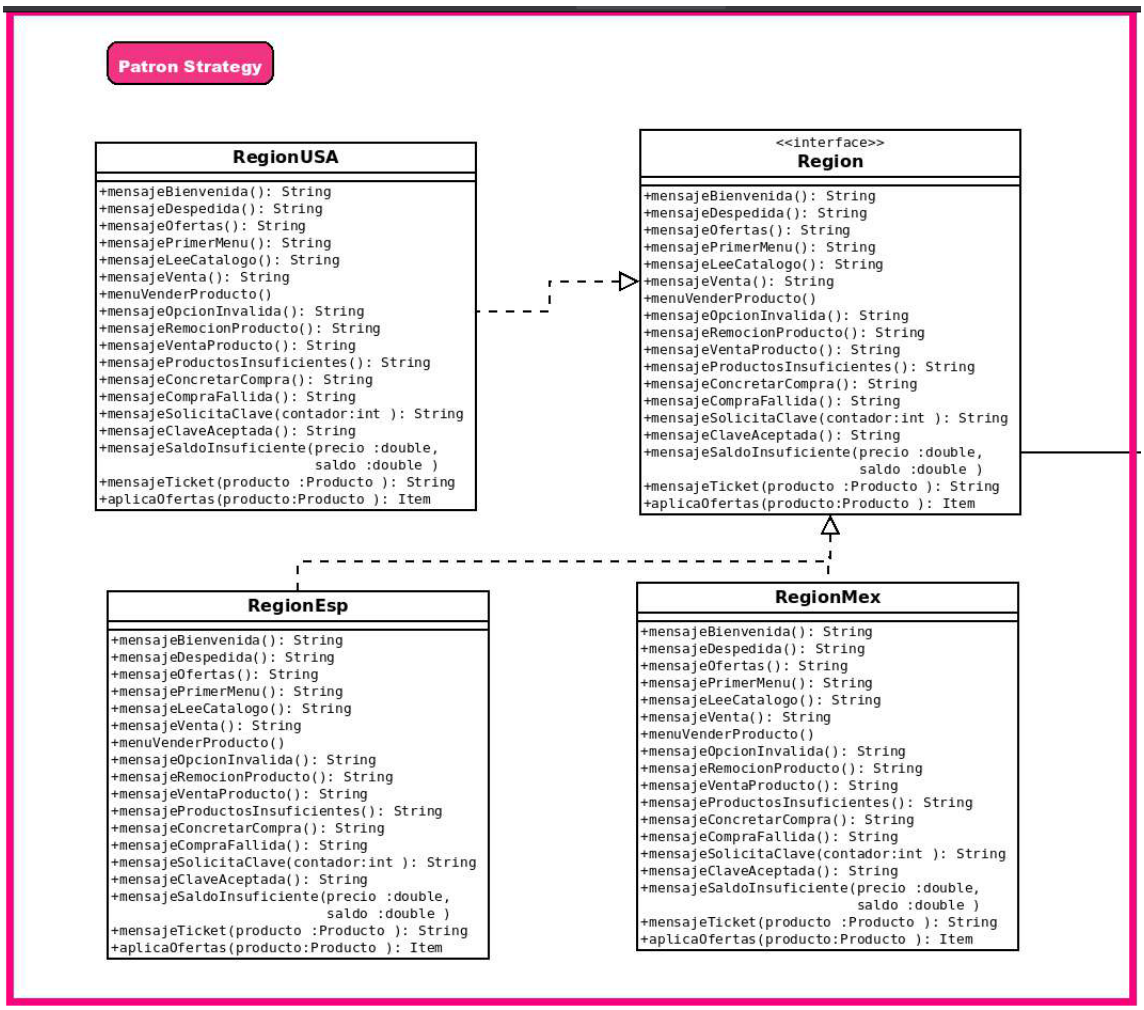

Figura 1: Diagrama UML Strategy

2. Patrón de diseño Iterator

Utilizamos este patrón en varias partes del proyecto.

1. En la parte de imprimir el menú, para recorrer la estructura donde están almacenados e irlos imprimiendo uno por uno.
2. En la parte encargada de verificar si el cliente es un usuario dado de alta, es decir, donde se inicia sesión y se verifica si los datos son correctos
3. En el carrito de compras, para ir iterando sobre todos los productos en éste y así saber cuánto es el precio a pagar.

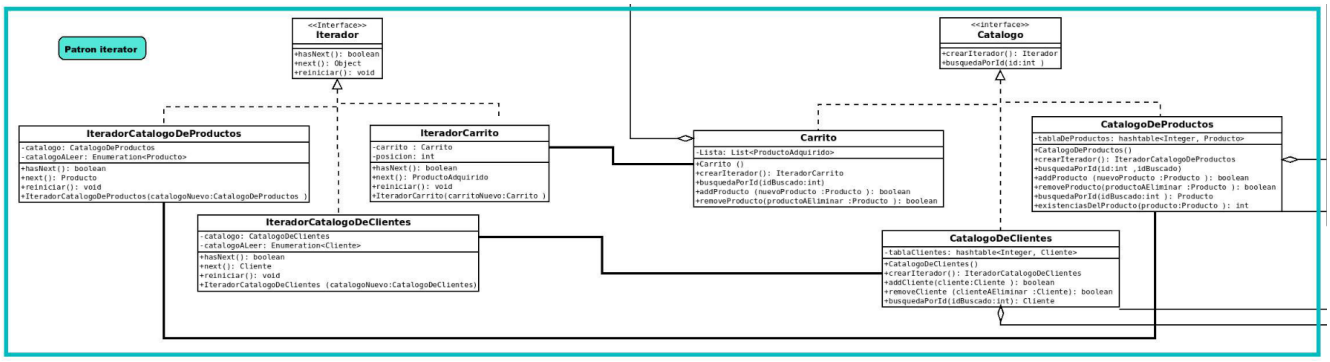

Figura 2: Diagrama UML patron Iterator

Razón de uso

Decidimos utilizar este patrón para recorrer los catálogos de nuestro proyecto debido a su sencillez de implementación, además de su fácil moldeabilidad para este tipo de casos donde se tienen que recorrer distintas estructuras, sin importar realmente qué estructura está iterando.

Usamos una lista de productos simples para el catálogo de productos pensando que a futuro, una implementación donde varios usuarios accedan al catálogo permitiría que cada usuario pueda acceder a los artículos que necesita sin tener que esperar a que se desocupen

3. Patrón de diseño Decorator

Como se observa en el siguietne *UML* este patrón está muy relacionado con el patron Prototype, ya que utilizamos ambos para el mejor manejo de los productos y las funciones necesarias pedidas para el proyecto

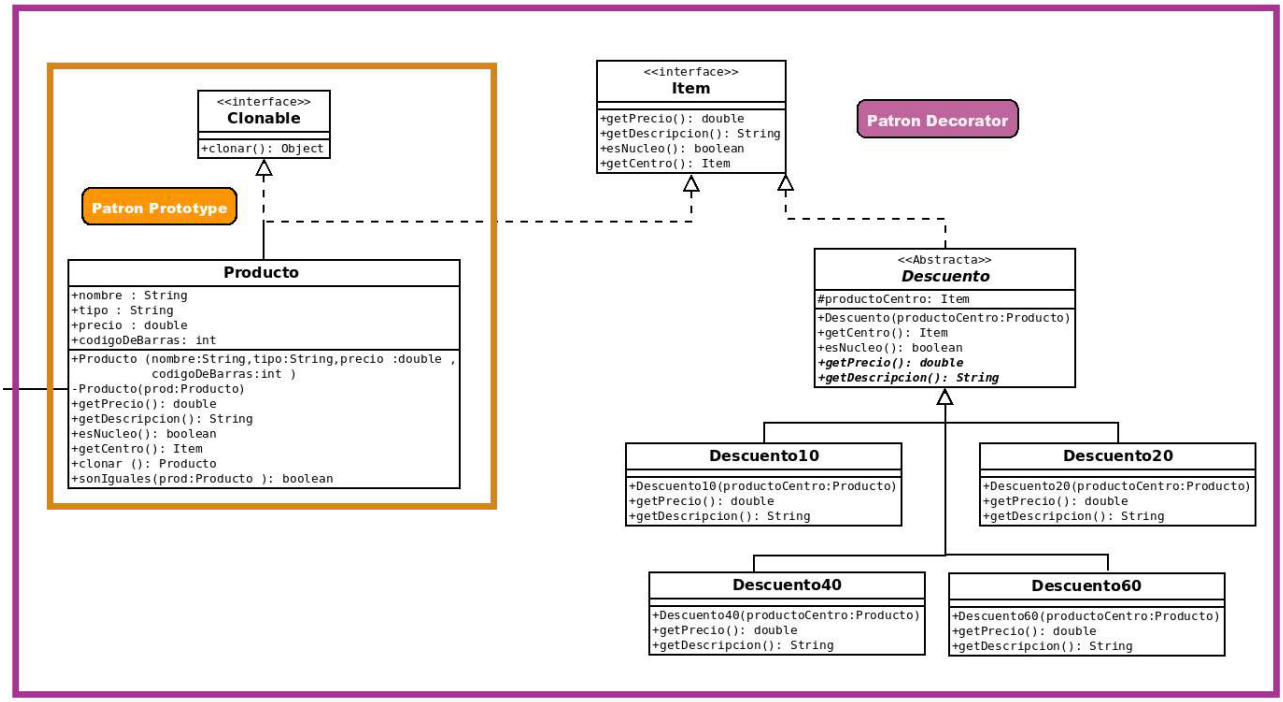

Figura 3: Diagrama UML patron Decorator

Razón de uso
Decidimos utilizar este Patrón en la parte de aplicar los descuentos en ciertos productos, ya que así nos evitamos problemas de descuentos permanentes, es decir, no se modifica el precio del producto *Raíz*, sólo lo cubrimos con un nuevo precio y regresamos ese nuevo producto con el precio en descuento, sin embargo, para futuras ejecuciones el precio seguirá con su precio normal (probablemente, debido a que las promociones se ejecutan al azar).

4. Patrón de diseño Proxy

Utilizamos este patrón para hacer un ayudante virtual, el método concretaCompra de esa clase se encarga de gestionar todo lo necesario para que la tienda pueda despachar los productos.

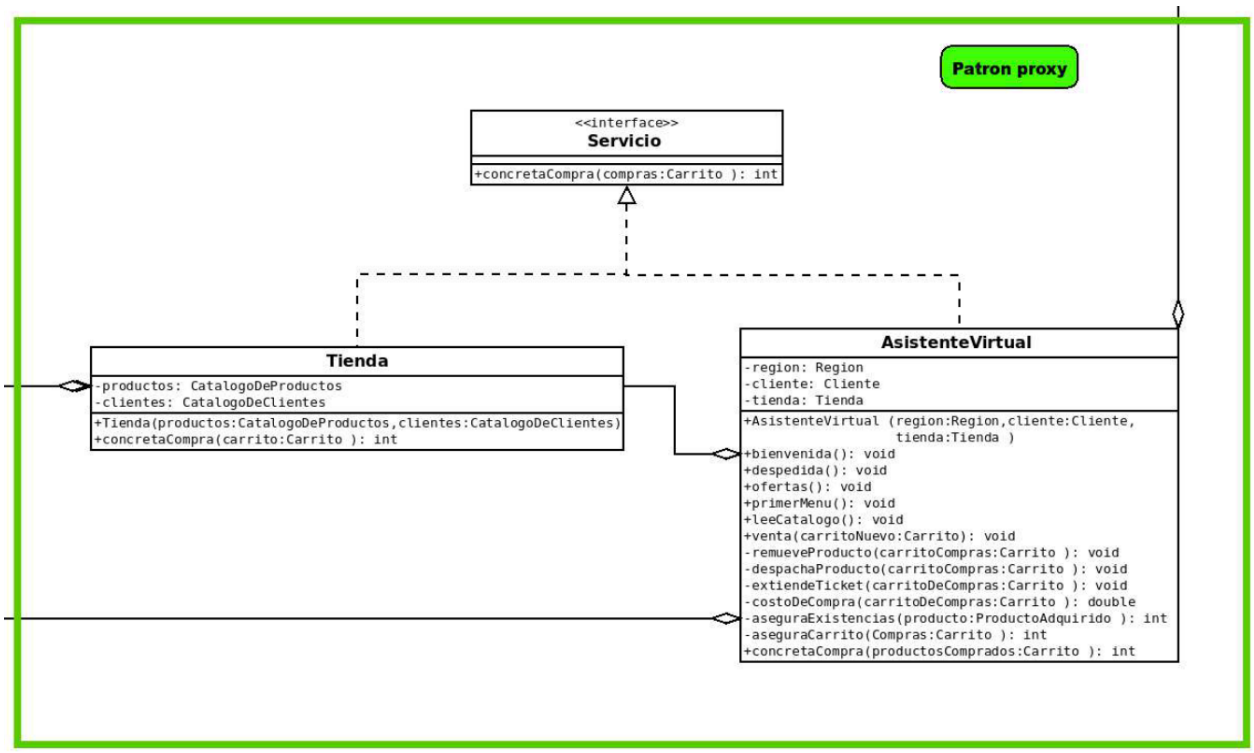

Figura 4: Diagrama UML patron Proxy

Razón de uso
Decidimos utlizar este patrón para este se encargue de casi todo al momento de estar ejecutando, como el nombre de la clase que lo implementa, lo utilizamos como un asistente virtual, ya que en el main practicamente se manda a llamar todo lo de esta clase.

5. Patrón de diseño Prototype

Utilizamos este patrón sólo para optimizar la creación de los productos de la tienda, práctimente, sólo se crearon unos pocos y los demás son copias de estos mismos.
Esto ayuda a tener más limpio el código y no tener que estar mandando a llamar el constructor de Producto cada que se vaya a hacer otro Producto que ya esté integrado en el menu.

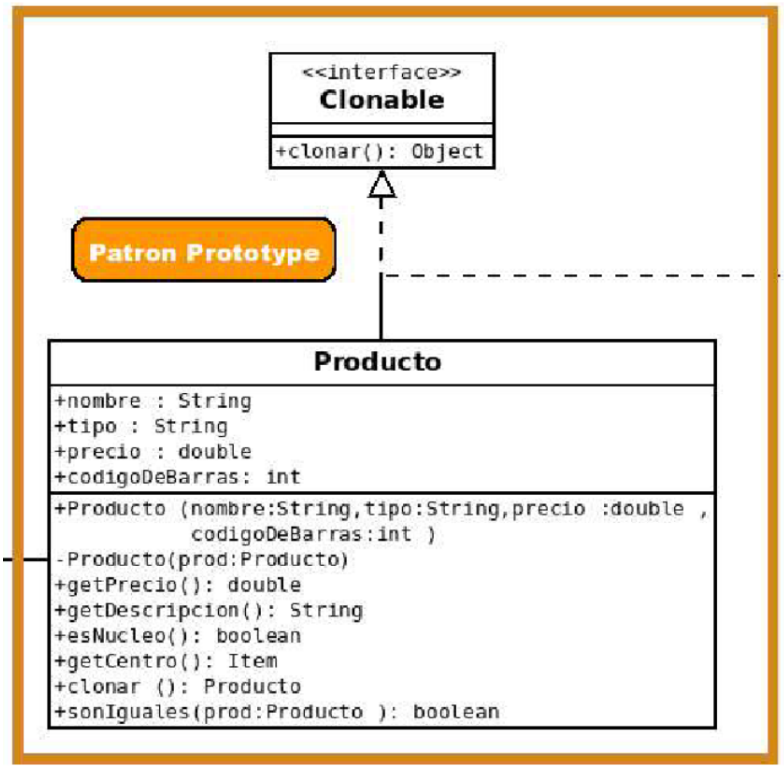

Figura 5: Diagrama UML patron Prototype

Por último.
Algunos puntos a aclarar para este proyecto son los siguientes:

1. En la carpeta entregada sólo incluimos el diagrama del caso cuando se compra un producto de forma exitosa, además se incluyó dos veces el UML realizado para el proyecto, uno en formato *png* y otro *pdf* (Por si disminuía la cálidad en el png). Con los siguientes nombres:

   DiagramaDeSecuencia-Proyecto1.png
   UMLProyecto1.pdf
   UMLProyecto1.png

2. Para compilar el proyecto escribimos lo siguiente en terminal

   javac *.java

   Y para ejecutarlo

   java Main

3. Los únicos clientes registrados en el sistema del proyecto son los siguientes:

   a) Arturo:
   ID: 315161718
   Password: AlfredoAdameTKM
   Pin de su tarjeta: 1234

   b) Rosa:
   ID: 308091011
   Password: HolaJapon789
   Pin de su tarjeta: 2468

   c) Itzel:
   ID: 326272829
   Password: HolaMundo123
   Pin de su tarjeta: 1379

   d) Fernando:
   ID: 218192021
   Password: MessiGOAT
   Pin de su tarjeta: 6789