EJERCICIO 1 PRÁCTICA DE DISEÑO

Principios de Diseño:

En este ejercicio utilizaremos el principio de responsabilidad única, al crear pequeñas clases que contienen un algoritmo muy concreto; y el de abierto/cerrado, abriendo nuestra solución a la extensión pero no a la modificación.. En nuestro código las clases que cumplen este principio son las clases CambioDeposito, CambioSimple. Tambien utilizamos el principio de sustitución de Liskov, en el cual al redefinir un método en una clase derivada, solo podemos sustituir una precondición por otra precondición mas débil y una postcondición mas severa. En el código cuando se pide que se pase una estrategia podemos pasar dos tipos distintos (CambioDeposito, CambioSimple).

Patrón de Diseño Usado:

Para realizar este ejercicio utilizamos el patrón ***Estrategia*** , que nos permite establecer en tiempo de ejecución el rol de comportamiento de una clase. El patrón estrategia se basa en el polimorfismo para implementar una serie de comportamientos que podrán ser intercambiados durante la ejecución del programa, logrando con esto que un objeto se pueda comportar de forma distinta según la estrategia establecida y por consiguiente en nuestro ejercicio, nos permite realizar diferentes tipos de “cambio” en tiempo de ejecución. A mayores hacemos uso del patrón ***Singleton*** diseñado para limitar la creación de objetos pertenecientes a una clase. El objetivo de este patrón es el de garantizar que una clase sólo tenga una instancia (o ejemplar) y proporcionar un punto de acceso global a ella, en nuestro caso es justo lo que necesitamos ya que queremos una instancia única por cada tipo de cambio.

DIAGRAMA DE CLASES

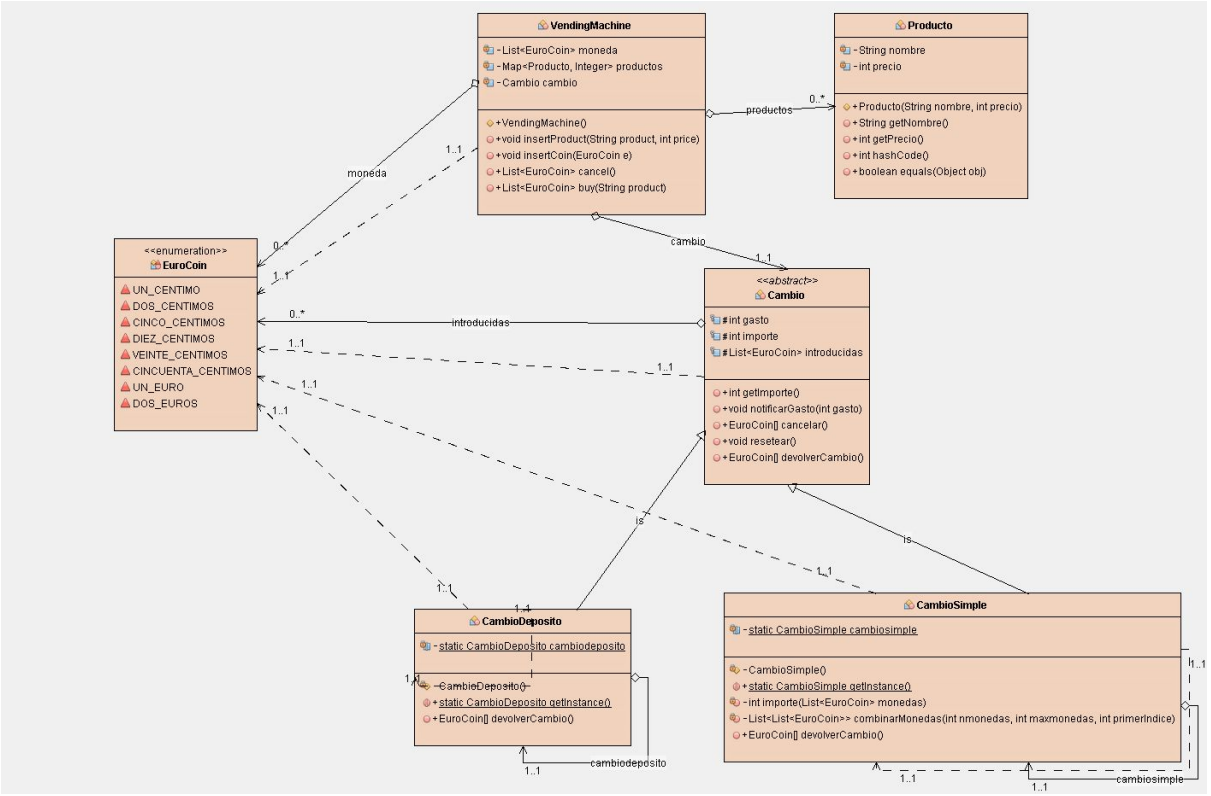

DIAGRAMA DINÁMICO