Modelado y programación 2022-1

Práctica 1: Implementación de los patrones Observer y Strategy

Bernal Márquez Erick. 317042522
Deloya Andrade Ana Valeria. 317277582

- ***Parte teórica:***

1. *Menciona los principios de diseño esenciales del patrón Observer y Strategy.*

   Observer: Define una relación de dependencia de uno a muchos entre un conjunto de objetos entre un "sujeto" y sus "observadores"; de forma que cuando un objeto cambia su estado, todos sus "observadores" dependientes son notificados automáticamente de este cambio.

   Los principios que sigue este patrón son:
   - ★ Diseño débilmente acoplado entre los objetos que interactúan.
   - ★ Menos interdependencia entre objetos.

   Strategy: Define una familia de algoritmos encapsulándolos y haciéndolos intercambiables dependiendo el comportamiento que se desee. El algoritmo va a cambiar su comportamiento independientemente del cliente que lo use.

   Los principios que sigue este patrón son:
   - ★ Identificar los aspectos que varían encapsulándolos y separándolos de los que no cambian.
   - ★ Favorece la composición sobre la herencia

2. *Menciona una desventaja de cada patrón.*

   Observer:
   - ★ Las actualizaciones transmitidas hacia los observadores en ocasiones pueden ser irrelevantes para algunos de ellos, se podrían producir en cascada actualizaciones ineficientes. Y más inconveniente sería si es que tenemos muchos observadores a los que les llegan notificaciones que no necesitan, malgastamos tiempo de ejecución.

   Strategy:
   - ★ No hay razón de utilizar Strategy si realmente no tenemos mas que un par de algoritmos que rara vez van a cambiar. Sólo complicaríamos el programa con exceso de las clases e interfaces que vengan con la implementación de Strategy. Por lo que el patrón solo debe ser utilizado si es que los cambios de estrategia y comportamiento son importantes para el funcionamiento del programa.

- ***Sobre la práctica:***
  El patrón Observer se implementa al momento de definir los servicios de Streaming y los clientes, siendo los servicios sujetos a observar y los clientes los observadores.

  Strategy lo implementan los servicios, pues dependiendo el tipo de servicio que tenga cada cliente va a elegir una manera diferente de cobrar.

  No se necesita de algún software adicional para el funcionamiento de la práctica. Puede ser fácilmente compilada con:

  ```
  $javac *.java
  ```

  Y ejecutada con

  ```
  $java Main.java
  ```

  En la terminal aparecerán impresiones del cobro de los servicios por cada mes, mostrando el cliente a cobrar y el tipo de servicio que tiene. El cobro de los servicios por orden es:

  - Memeflix
  - Momazon
  - Twitsh
  - Thisney
  - HVO

  Al final de cada mes se muestra el cliente y su estado de cuenta.