Taller Integrador: Patrones de diseño

Sistema de Requerimientos Académicos.

Objetivos

· Evaluar que patrones de diseño se deben utilizar en el desarrollo de un sistema.

· Aplicar varios patrones de diseño dentro de un mismo sistema.

Requerimientos del sistema:

Se desea desarrollar un sistema de cajero automático que permita manejar consultas de saldo, retiros y depósitos de cuentas bancarias, para esto se le provee una clase llamada Account, la cual es la única que no debe ser modificada, pero también se proveen 3 clases más, que pueden ser modificadas a conveniencia, y sobre las que se debe aplicar los patrones de diseño elegidos.

Desarrollar

1. Indique para cada uno de los patrones estudiados si pudiera o no servir dentro del desarrollo de este sistema. (explique)

    a. Creacionales.

    - Singleton, si sirve dentro del desarrollo del sistema ya que al Crear un único cajero Automático de dólares, es decir una única instancia se debería usar este patrón, ya que de esta forma se garantizará que se tenga una única instancia y un punto de acceso global a ella.

      Porque se necesita una única instanciación al momento de realizar un depósito y/o retiro así como consulta de saldos de la cuenta bancaria

    - Factory Method: No se utiliza porque estamos definiendo el proceso a realizar, es decir, retiro de dinero, depósito y consulta de saldo en su cuenta

- Abstract Factory: no nos sirve porque no necesitamos crear o tener familias de diferentes clases para resolver el problema, en este caso para la moneda.

b. Estructurales.

- Adapter: No sirve, porque no necesitamos conectar dos interfaces incompatibles por lo tanto un “wrapper” no es mucha utilidad.

- Composite:No sirve por que el usuario no podrá tratar objetos individuales y composiciones de objetos de forma uniforme.

- Decorator: No debido a que no vamos a añadir más funcionalidad de la principal a la que se necesita que es depositar, retirar y consultar el saldo

c. De Comportamiento.

- Memento: No sirve porque por motivos de seguridad, el comportamiento tipo “last checkpoint” no es recomendable para una transacción bancaria.

- Strategy: No sirve, porque Strategy se enfoca en proveer varias alternativas “para moverse de un punto A a un punto B” (resolver un requerimiento). Las transacciones especificadas sólo tienen un modo de efectuarse.

- Chain of Responsibility: Sí sirve, para el uso de los manejadores. Más específicamente, para que un depósito o retiro sólo involucra una denominación de moneda a la vez(Existe un manejador por denominación)

- Iterator: No sirve, porque para el requerimiento en cuestión no se requiere recorrer una colección completa (en este caso, los manejadores. Es menester que el recorrido termina cuando encuentre el manejador apropiado)

2. Diseñe un diagrama de clases del sistema, aplicando los patrones elegidos.