Informe con los principios y patrones de diseño utilizados en el ejercicio de “Cotización de acciones en el mercado bursátil”

Principios de Diseño Utilizados

1. **Responsabilidad Única (SRP):** Cada clase tiene una única responsabilidad:

- MercadoAcciones se encarga de actualizar las acciones y notifica a los observadores.
- Cliente procesa la información utilizando una estrategia definida (simple, detallada y concreta).
- Estrategia encapsula la lógica para procesar los datos según la necesidad del cliente.

2. **Abierto/Cerrado (OCP):** El sistema está abierto a ampliaciones, pero cerrado a modificaciones. Así podríamos, por ejemplo, agregar nuevas estrategias derivadas de Estrategia sin modificar las clases existentes.

3. **Inversión de Dependencia (DIP):** Las clases dependen de abstracciones en lugar de implementaciones concretas. Por ejemplo, los clientes dependen de la interfaz Estrategia, no de sus implementaciones.

4. **Sustitución de Liskov (LSP):** Las subclases de Estrategia pueden sustituir a la clase base sin alterar el comportamiento esperado del sistema.

5. **Segregación de Interfaces (ISP):** Los observadores implementan la interfaz Observer, que define únicamente el método actualizar.

Patrones de Diseño Utilizados

Patrón Observer:

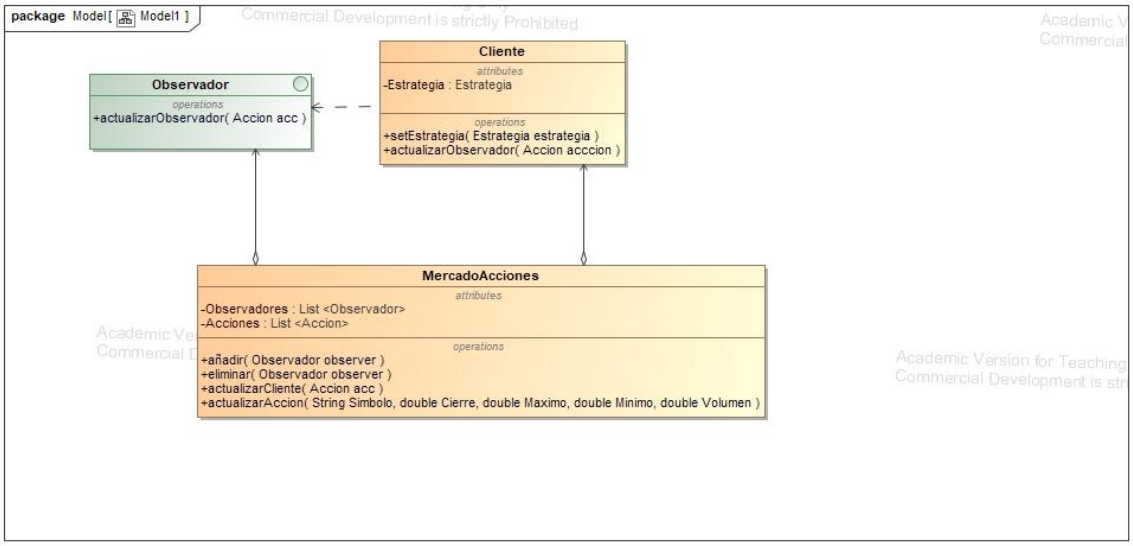
El patrón Observer permite establecer una relación de dependencia uno-a-muchos entre el mercado bursátil y los clientes interesados. Cuando el mercado actualiza una acción, notifica a todos los observadores registrados. Este patrón desacopla el sujeto (“MercadoAcciones”) de sus observadores (“Cliente”), permitiendo agregar o eliminar observadores sin modificar el sujeto.

Clases Involucradas y Roles

- **Sujeto (Subject):** MercadoAcciones.
- **Observador (Observer):** Interfaz Observador.
- **Sujeto Concreto (ConcreteSubject):** Implementación de MercadoAcciones.
- **Observadores Concretos (ConcreteObserver):** Implementaciones de Cliente.

Diagrama de Clases

Se incluye un diagrama UML que muestra la relación entre MercadoAcciones, Observador y Cliente.



Patrón Estrategia

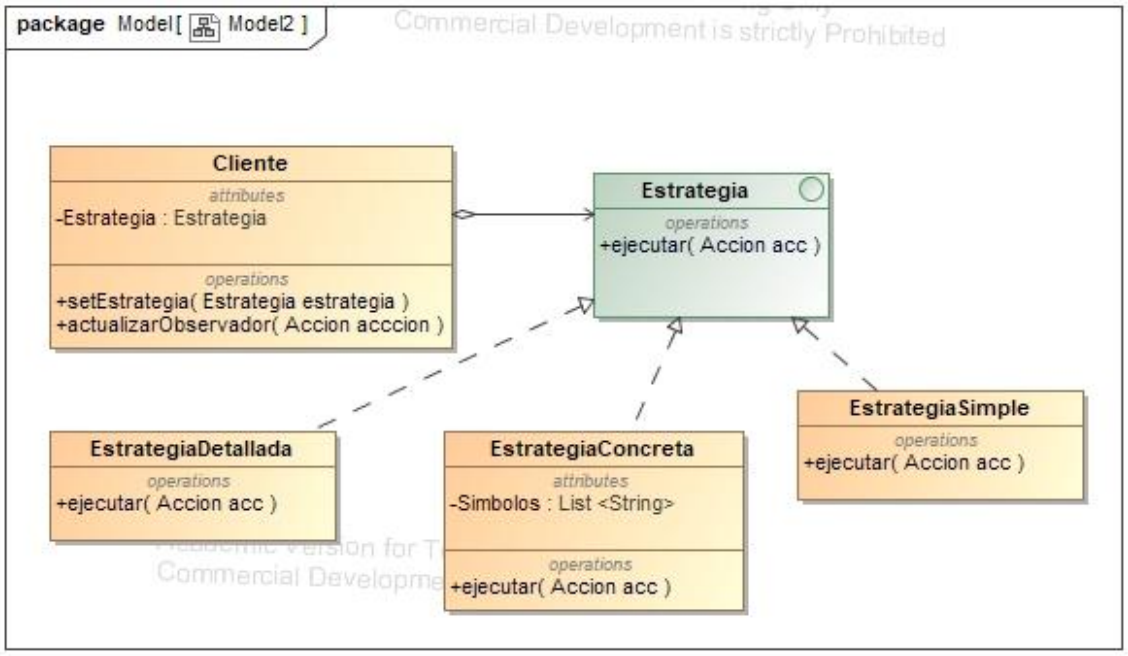
El patrón Estrategia encapsula algoritmos de procesamiento en clases separadas, permitiendo que las estrategias sean intercambiables. Esto es útil en este sistema para manejar diferentes tipos de procesamiento de datos bursátiles (precio de cierre, datos detallados, información específica de acciones).

Clases Involucradas y Roles

- **Contexto:** Cliente.
- **Estrategia :** Interfaz Estrategia.
- **Estrategias Concretas:** EstrategiaSimple, EstrategiaDetallada, EstrategiaConcreta.

Diagrama de Clases

Se muestra un diagrama UML con las relaciones entre Cliente y las estrategias concretas, destacando la implementación de la interfaz Estrategia.



Este diseño modular basado en patrones asegura:

- Desacoplamiento entre el mercado y los clientes (Observer).
- Flexibilidad para modificar o agregar nuevos comportamientos (Strategy).

Los principios SOLID garantizan un código robusto y adaptable, preparado para futuros cambios o extensiones.