**1. Patrón Creacional – Singleton**

**Ventajas:**

* Controla el acceso a una única instancia global
* Reduce el uso de recursos al evitar múltiples instancias
* Facilita la gestión compartida de datos (ej. configuración, conexión DB)

**Desventajas:**

* Dificulta las pruebas unitarias (por su estado global)
* Puede generar acoplamiento fuerte
* Rompe el principio de responsabilidad única si la clase hace más que garantizar una instancia

**2. Patrón Estructural – Adapter**

**Ventajas:**

* Permite reutilizar clases existentes sin modificarlas
* Facilita la integración con código heredado o bibliotecas externas
* Desacopla el cliente de las implementaciones reales

**Desventajas:**

* Añade complejidad al diseño
* El uso excesivo puede generar una gran cantidad de clases adaptadoras

**3. Patrón de Comportamiento – Strategy**

**Ventajas:**

* Facilita cambiar algoritmos en tiempo de ejecución
* Promueve el principio de abierto/cerrado (Open/Closed)
* Reduce condicionales complejos y mejora la legibilidad

**Desventajas:**

* Aumenta el número de clases en el sistema
* El cliente debe conocer las diferentes estrategias para seleccionarlas correctamente
* Puede implicar una sobreingeniería para problemas simples