|  |  |  |
| --- | --- | --- |
|  | Decorator | Strategy |
| Facilidad para agregar nuevos criterios de filtro | Aporta mayor facilidad | Es menos flexible |
| Posibilidad de eliminar filtros dinámicamente | Muy difícil implementarlo | Más fácil hacerlo |
| Posibilidad de intercambiar el orden en que se  aplican los filtros o el posterior manejo  de resultados | Es posible | No es posible |
| Simplicidad | Menos simple | Más simple |
| Cohesión o cuántos objetivos resuelve el componente desarrollado | Es muy cohesivo | Menos cohesivo |
| Mantenibilidad | Menos mantenible | Más mantenible |

El decorator me permite agregar nuevos criterios muy fácilmente pero eliminarlos o modificarlos se torna muy complejo ya que desdecorar el componente requiere conocer al decorator específico (la clase), y a su vez en el strategy es menos flexible para agregar, pero modificar o eliminar dinámicamente es muy sencillo.

El Strategy es más sencillo que el decorator, ya que en este último para manejar los constructores hay que conocer bien el decorator mientras que el strategy no.

En el decorator es posible por la lógica que tiene ya que si es postProceso siempre va a hacer el super() para que se ejecute el procesamiento para luego ejecutarse si mismo. Entonces por más que cambiemos el orden de los filtros, el orden de ejecución ya esta contemplado. A diferencia del Strategy que tienen que seguir un orden específico.

Cohesion en el decorator muchos objetos chicos en lugar de uno grande, son objetos más cohesivos pero más difíciles de entender / debuggear. En el Strategy el componente debe conocer que Strategy esta usando a diferencia del decorator. Entonces hay cierto grado de acoplamiento que disminuye la cohesión.

Mantenibilidad: El decorator es menos mantenible del strategy, no solo por lo mencionado en el primer ítem. Sino que también en el decorator el objeto pierde de identidad al cambiar la capa del decorator y también al ser más difícil de debuggear es menos mantenible. En cambio en el strategy el componente siempre sigue siendo el mismo (mantiene identidad) al utilizar diferentes strategies.