Nos tocó hacer el Ej 4 de la práctica 4.

El enunciado trata de un decodificador que la empresa les da a sus clientes para que puedan ver películas.

El decodificador muestra una grilla y también sugiere películas.

Nosotros tenemos que implementar una aplicación para que cumpla con esto.

El decodificador puede ser configurado para que sugiera 3 películas que no se hayan reproducido por distintos criterios.

Uno de los criterios es novedad, este criterio muestra las películas más recientes.

El otro es la similaridad, acá se muestran las películas más nuevas similares a alguna que ya se reprodujo.

Y el otro criterio es el puntaje: las películas de mayor puntaje, para igual puntaje considera las más recientes.

También teníamos que tener en cuenta que la configuración del decodificador no es fija, sino que el usuario la puede cambiar.

Esto último nos da pie para usar un patrón y el que elegimos fue el patrón Strategy.

El cliente no va a necesitar todas las estrategias en todo momento, de modo que no necesitamos cargar un objeto con toda esa lógica dentro cada vez que queramos usarlo, de éste modo estaríamos ahorrando memoria.

La misma estrategia puede usarse en diferentes contextos, evitando el código duplicado.

Entonces por qué no usamos el patrón [State](http://www.dofactory.com/Patterns/PatternState.aspx)? El State permite **hacer diferentes cosas** dependiendo del estado del objeto. Lo que cambia de acuerdo al estado es **que** se hace.

En cambio el patrón [Strategy](http://www.dofactory.com/Patterns/PatternStrategy.aspx) permite **hacer lo mismo de diferentes maneras**. Lo que cambia es **como** se hace. Permite que el algoritmo cambie sin que afecte a los clientes que lo usan.

Acá tenemos el UML reducido con sus participantes.

* Contexto, la clase Decodificador. es el responsable de crear y mantener una referencia a una estrategia concreta.
* Strategy, la estrategia: en este caso es la clase Criterio. Declara una interfaz común para todos los algoritmos que soporta. Esta interfaz va a ser usada por el contexto para invocar a la estrategia concreta.
* Estrategia concreta, las clases hijas de la jerarquía, Implementa el algoritmo utilizando la interfaz definida por la estrategia.