|  |  |  |
| --- | --- | --- |
| CARACTERISTICAS | SOLUCION  CON  OBSERVER | SOLUCION  SIN OBSERVER |
| Facilidad para agregar nuevos tipos de monitoreo | A nivel diagrama es fácil ya que hay que crear una clase que implemente la interfaz monitor, luego en la implementación se agrega uno por uno en la colección monitores en repositorioDeRecetas | A nivel diagrama es igual que observer, esta solución no cambia esta parte, ya que es necesario que sean polimórficas entre sí para comprender el mensaje notificar, dentro de notificarATodos, en cuanto al código, es mas fácil instanciar una colección auxiliar y setearle todos los monitores que queramos en un solo paso, y luego pasar por parámetro esta colección |
| Simplicidad | Es más simple que la alternativa sin oberver porque tenemos una variable como colección que está dentro de la clase observada, y esto reduce la cantidad de parámetros en la firma de los métodos | Es menos simple esta solución ya que se estaría llegando a violar el code small de long parameter list, ya que consideramos que 4 parámetros es mucho para un método que solo se encarga de listar y notificar |
| Cohesión o cuantos objetivos a la vez resuelve el componente desarrollado | Es cohesiva ya que delega luego de listar el método notificar, | Permanece la cohesión pese a que se recibe un parámetro de mas, pero no se efectúa ninguna acción sobre el mismo, ya que se lo pasa por parámetro al método notificar |
| Mantenibilidad | Es más fácil de mantener ya que hay menos código en la firma de los métodos, y es más fácil testear porque está definido el método agregar monitor | es más difícil de mantener ya que hay mas código en la firma de los métodos, y en los tests hace falta crear una lista en la cual agregar todos los monitores |