# Registro de Decisiones

Se considera importante registrar las decisiones para hacer un seguimiento de la evolución de la solución, facilitar su compresión y justificar las decisiones.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fecha | Decisión | Contexto | Ventajas | Desventajas | Alternativas |
| 02/09/2015 | Se implementa el patrón Singleton sobre la clase Usuario. | Diagrama de Clases  Código Fuente | Permite que la aplicación utilice una única instancia global del usuario que inicia sesión en un browser. Además desde el punto de vista del diseño e la propia clase responsable de crear la única instancia y permite acceso global mediante un método de clase. |  |  |
| 02/09/2015 | Se implementa el patrón Strategy para las Estadísticas y Reportes. | Diagrama de Clases  Código Fuente | Permite definir familias de algoritmos relacionados, lo que posibilita agrupar funciones comunes y facilitar la reutilización del código. Elimina el uso de sentencias condicionales (if, switch/case). | Incrementa el número de objetos que pueden ejecutarse en una aplicación. | Podriamos utilizar un patron |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fecha | Decisión | Contexto | Ventajas | Desventajas | Alternativas |
| 02/09/2015 | Se implementa el patrón Visitor para Proponer Recetas al Usuario y realizar Recomendaciones en la preparación de las mismas. | Diagrama de Clases  Código Fuente | Permite definir una nueva operación sin cambiar las clases de los elementos que opera. Facilita el añadir nuevas Clases de Elementos Concretos. | Rotura de la encapsulación. |  |