Patrones de diseño

Derian Herrera, Julio Mejia, Randi Paredes , Marco Garcia y Alisson Chino

October 9, 2020

I. Resumen

En el siguiente articulo observaremos algunos de los patrones de diseño usados. Los diseñadores expertos no solucionan los problemas desde sus principios sino que reutilizan soluciones que anteriormente funcionaron. Aqui se encuentran los patrones de diseño que resuelven problemas especificos y hacen el diseño flexible y reusable.

II. Abstract

In the following article we will observe some of the design patterns used. Expert designers do not solve problems from the beginning instead they reuse solutions that previously worked. Here are the design patterns that solve specific problems and they make the design flexible and reusable.

III. Introduccion

Los patrones de diseño es un tema importante en el desarrollo de software actual, lo que busca es ayudar a los desarrolladores de software a resolver problemas comunes creando un lenguaje comun para comunicar ideas y experiencias acerca de problemas y soluciones. El usar patrones de diseño ayuda a tener un software de calidad. Según su proposito los patrones se pueden clasificar en tres: De creacion, proceso de creacion de objetos. De estructura, tratan composicion de clases y/o objetos. De comportamiento, se caracterizan en la forma que interactuan y reparten responsabilidades a sus clases y objetos.

IV. Desarrollo

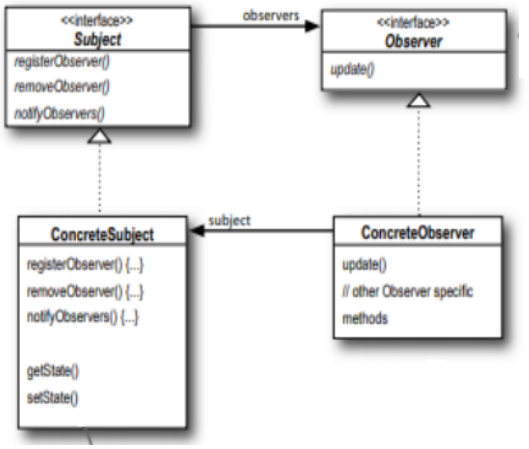
i. Patron de diseño observer

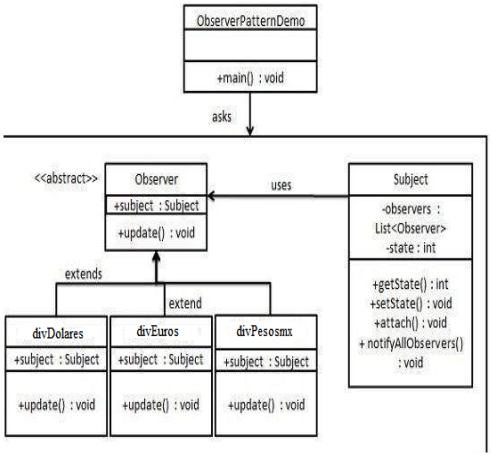
El patron de diseño observer es un patron comportamental,este patron define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y se actualicen automáticamente todos los objetos que depende de él.

- **Descripcion**
  - Los objetos principales son "Subject" y "Observer"
  - La motivacion de este patron es la reutilizacion
  - Puede no haber relacion directa entre objetos
  - El tipo de interaccion es conocida como publicar-suscribir
  - El subject es publicador de notificaciones
  - Cualquier numero de observers puede suscribirse para recibir notificaiones

- **Componentes**
  Subject
  Cualquier numero de observers puede observar a subject
  Observer
  Define una interfaz de actualizacion para los objetos Observer que deben ser notificados de los cambios en el Subject
  Concrete Subject
  Almacena estados de interes para los objetos Concrete Observer.
  Envia notificaciones a sus observers cuando el estado cambia.
  Concrete Observer
  Mantiene referencia de objetos Concrete Subject.
  Almacena los estados que deben ser

consistentes con los subjects.
Implementa las observaciones del observer.





- **Ventajas**
  -Permite variar los sujetos y observadores indepedientemente.Se puede rehusar sujetos sin el rehuso de observadores y viceversa.
  -Permite agregar observadores sin modificar el sujeto o los observadores.
- **Desventajas**
  -No se especifica el receptor de una actualizacion.Se envia a todos los objetos interesados.
  -Actualizaciones inesperadas.Se podrian dar actualizaciones en casacada muy ineficientes.
- **Ejemplo**
  Crearemos un ejemplo en el cual al ingresar un monto en soles el programa nos dara la alerta segun el observador de cuando sera el cambio a la moneda de ese observador.

ii. Segundo patron

- subtitulo
  Contenido.....
  sub sub tiutl consalto de linea
  LLenar completar.....
  - Para enumerar y resaltar

iii. Tercer patron

agregar

iv. Cuarto patron

v. Quinto patron

vi. Sexto patron

vii. Setimo patron

viii. Octavo patron

V. Conclusiones

VI. Recomendaciones

References

[1] lisabeth Freeman, Kathy Sierra (2004). Head First design patterns, Sebastopol, CA: O'Reilly.