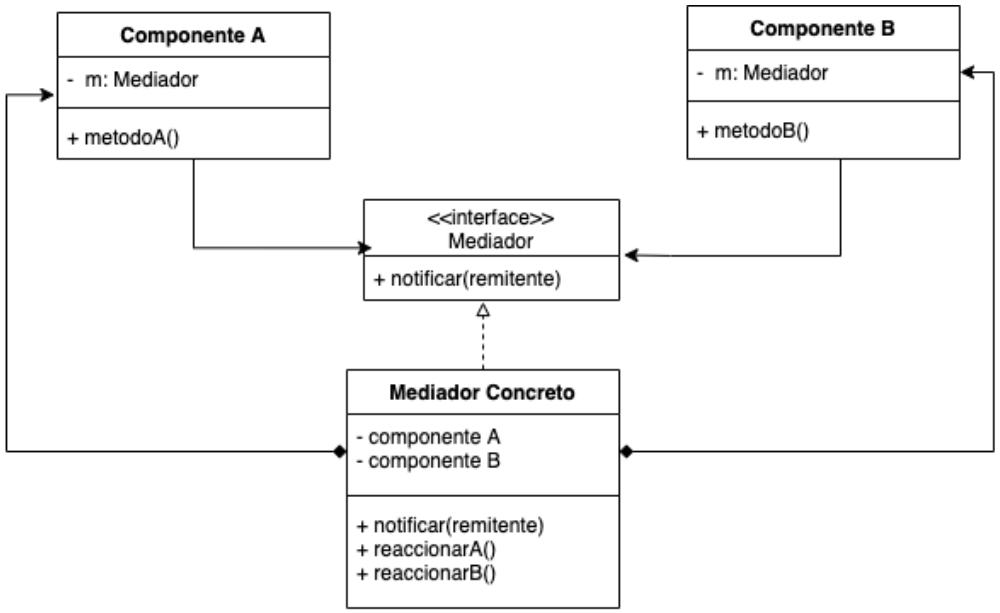
Patrón de Diseño: Mediador
Nombre Alumno: Angel Odiel Treviño Villanueva

Nombre: Mediador
Clasificación: Patrón de Comportamiento

Intención: Restringe la comunicación directa entre objetos y los obliga a comunicarse entre ellos a través de un objeto mediador, el cual les indica como deben de actuar.

Aplicabilidad: Este patrón es recomendable usar cuando cambiar las clases es difícil de cambiar debido a que esta relacionado con otras clases, no se puede reutilizar el componente debido a su dependencia con otros componentes o cuando se crean componentes de subclases solo para reutilizar una función básica. Esto es debido a que el mediador te permite quitar dependencias entre otros componentes y así poder tener una comunicación mas simple entre componentes.

Estructura:



El diagrama consiste en un conjunto de componentes, para este diagrama son 2 pero pueden ser mas, los cuales tienen métodos que realizar. Estos componentes implementan la interfaz de mediador, el cual notificara que se hizo una acción del componente correspondiente. El mediador concreto es el que junta nuestros componentes y es el que genera la interacción entre los componentes gracias a la interfaz. Los componentes no deben de saber sobre los otros componentes, ellos solo necesitan notificarle al mediador que realizaron una acción para que este realice la acción correspondiente en el Mediador concreto.

Consecuencias: Este patrón ayuda aplicar el principio de responsabilidad única debido a que extraes la comunicación entre varios componentes a un solo lado, puedes utilizar componentes individuales fácilmente. También ayuda a mantener el principio de Open/closed debido a que te permite introducir nuevos mediadores sin cambiar la estructura de los componentes y por último este te permite reducir el acoplamiento entre varios componentes Lo que si tienes que tener en cuenta es que tienes que evitar un mediador se vuelva un God Object, el cual se refiere a un objeto que realiza muchas funciones.

Implementación: Para poder Implementarlo se necesitan 5 pasos.

1. Identificar un grupo de clases que interactúan entre si y el cual puedan beneficiarse si los hacemos mas independientes.
2. Encapsular las interacciones en una nueva clase, la cual será nuestro mediador.
3. Crear una nueva instancia de esa clase y modificar los objetos componentes para interactuar únicamente a través del mediador.
4. Durante este proceso tendrás que equilibrar el principio de desacoplamiento con el principio de distribuir la responsabilidad de manera uniforme.
5. También hay que tener cuidado de crear un God Object.

Usos Conocidos: Este patrón se puede ver utilizado en la vida real gracias a las torres de aviación y como dan indicaciones para los aviones para poder aterrizar o despegar. Esto es debido a que como los aviones no saben las posiciones de los otros aviones, estos se comunican con la torre de control, nuestro mediador, y el cual les indica que acciones tomar a los diferentes aviones.

Patrones relacionados: Este patrón esta relacionado con Cadena de responsabilidad, comando, observador y fachada, ya que estos tratan de conectar de diferentes maneras los que envían y reciben peticiones.

Ejemplo: Este patrón se puede observar también en las listas enlazadas, ya que este funciona como mediador para los diferentes nodos en la lista. La lista indica si se necesita realizar la búsqueda dentro de los elementos o imprimir la lista, y hace la comunicación entre los diferentes nodos de la lista. En el ejemplo creamos una lista, la cual nos ayudara a insertar nodos o borrarlos y mostrar nuestro contenido de los nodos.

Video: https://youtu.be/m6MLOr1GkBg