

ISWD523 DISEÑO DE SOFTWARE GR2SW

Nombre: Fernando Eliceo Huilca Villagómez **Fecha:** 04/06/2025 **Curso:** GR2SW

Ejercicio:

En una habitación se encuentra un sensor de luz, el sensor es capaz de encender o apagar las luces de la habitación en función de un umbral determinado.

Razonamiento para el diseño:

Para implementar esta lógica de forma flexible y desacoplada, se utiliza el patrón Observer, el cual es ideal cuando un objeto debe notificar a otros sobre cambios en su estado.

A continuación, identificamos los elementos del patrón dentro de nuestro contexto:

Publicador/Sujeto Observado: Es el componente que representa el estado cambiante del sistema y que notifica a sus observadores ante una modificación.

- En nuestro caso: La luminosidad de la habitación, ya que varía a medida que se encienden o apagan los focos.






Suscriptores/Observadores: Son los elementos que se suscriben al sujeto y reaccionan ante cualquier actualización de su estado.

- En nuestro caso: Sensor de luz, que recibe las notificaciones de cambio de luminosidad y decide si debe encender o apagar los focos asignados.

Adaptación de la estructura del patrón Observer sobre nuestro caso:


Ahora, siguiendo la estructura clásica del patrón, el sujeto observado (Luminosidad) debe implementar los siguientes elementos:

1. Lista de observadores
2. Métodos agregar y eliminar observadores
3. Método notificar

 Luminosidad	
	listSensoresEscuchadores : Sensor
	agregarSensor(sensor : Sensor)
	eliminarSensor(sensor : Sensor)
	notificar(cantidadDeLuminosidad)

ISWD523 DISEÑO DE SOFTWARE GR2SW

Ahora podemos agregar cosas propias de nuestra clase según el contexto del ejercicio:

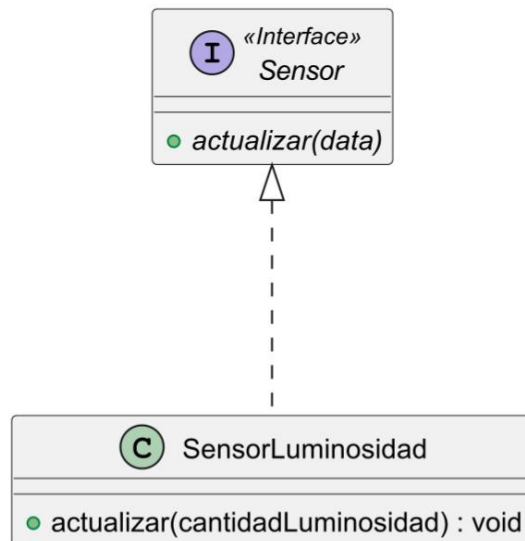
 Luminosidad
<ul style="list-style-type: none">❑ cantidadDeLuminosidad : double❑ listSensoresEscuchadores : Sensor
<ul style="list-style-type: none">● sumarLumenes(double : lumenes)● agregarSensor(sensor : Sensor)● eliminarSensor(sensor : Sensor)● notificar(cantidadDeLuminosidad)● getCantidadLuminosidad() : double

Luego, y siguiendo la estructura del patrón, cada observador debe implementar una interfaz común que le permita ser notificado de los cambios que ocurren en el sujeto observado.

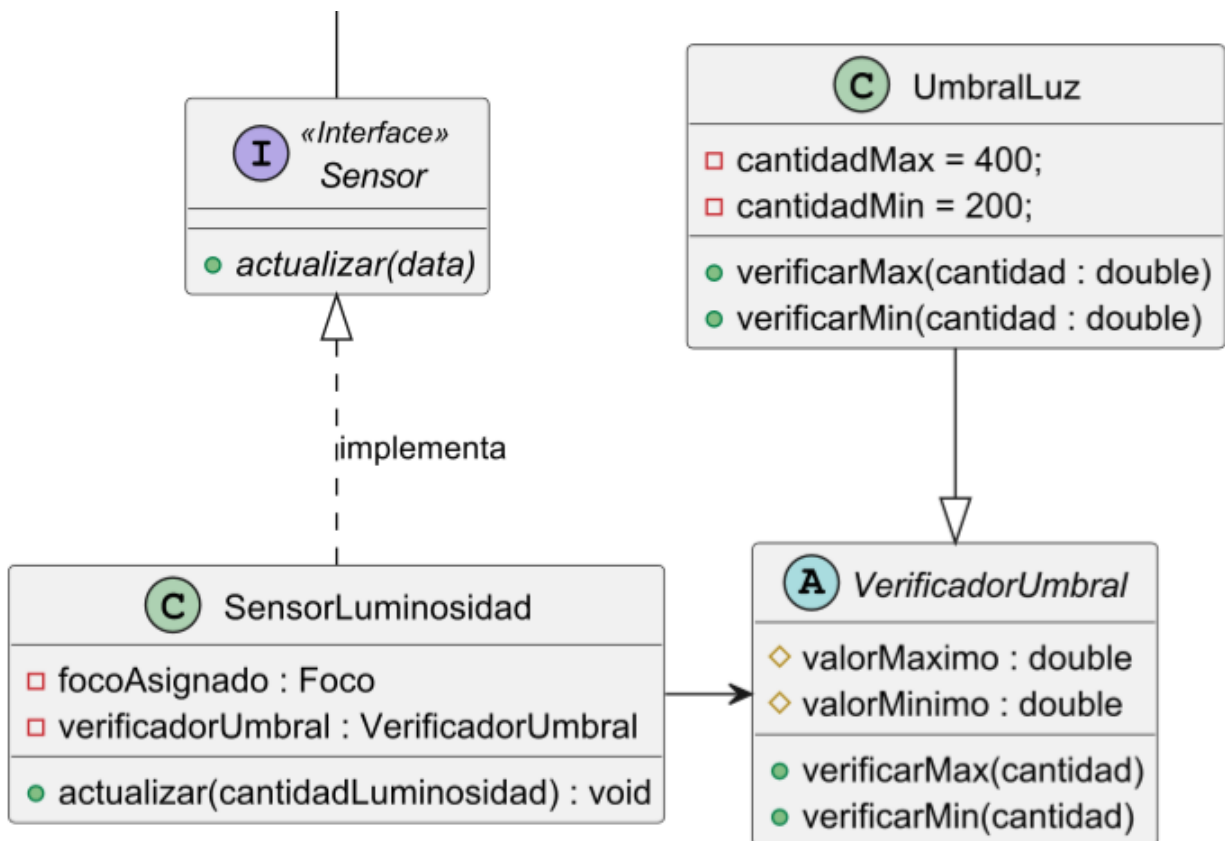
En nuestro caso, el observador es el sensor de luz, que reacciona ante los cambios de luminosidad. Por lo tanto, la interfaz que debe implementar los observadores (como SensorLuminosidad) debe contemplar lo siguiente:

1. La interfaz tiene un único método de actualizar, que será llamado por el sujeto observado.
2. Este método actualizar recibe como argumento el nuevo valor de luminosidad, de forma que el observador pueda evaluarlo y decidir su acción.
3. La interfaz no debe depender de detalles internos de la clase Luminosidad.

ISWD523 DISEÑO DE SOFTWARE GR2SW



Ahora, agregamos la lógica de nuestro ejercicio sobre esta estructura:



ISWD523 DISEÑO DE SOFTWARE GR2SW

Como conclusión tenemos el siguiente diagrama:

Patron Observer - Diagrama Clases

