Patrón Observador

El patrón **observador** define una dependencia del tipo uno a muchos entre objetos, de manera que cuando uno de los objetos cambia su estado, el observado se encarga de notificar este cambio a todos los otros dependientes.

Por ejemplo:

La hoja de cálculo y gráfico de barras dependen de los datos del objeto y por lo tanto deben ser informados de cualquier cambio en su estado. Y no hay razón para limitar el número de los objetos dependientes , puede haber cualquier número de interfaces de usuario diferente para los mismos datos.

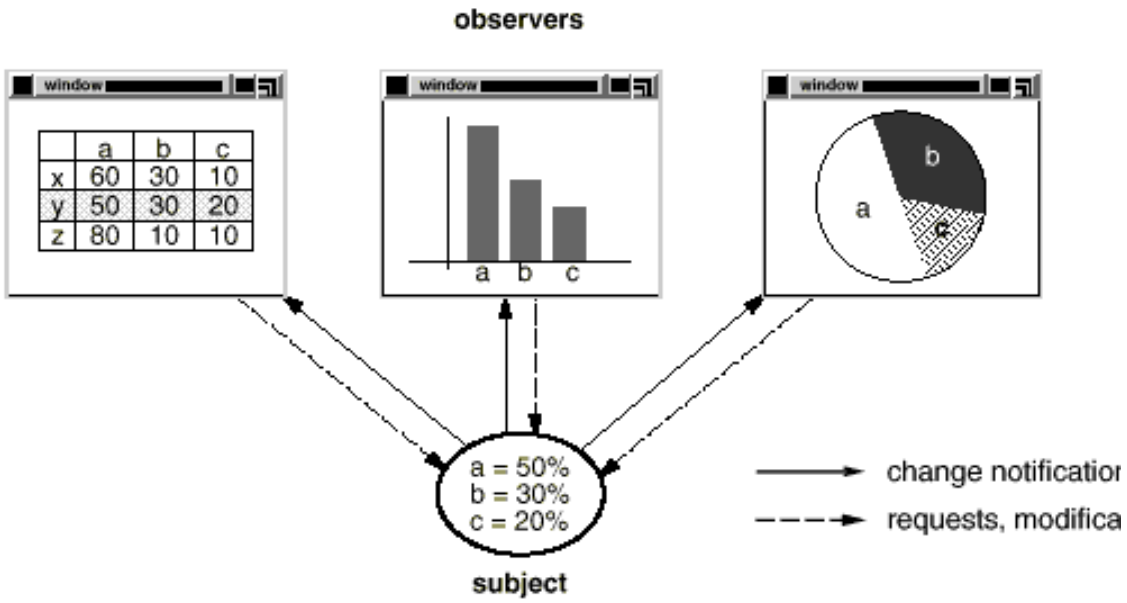
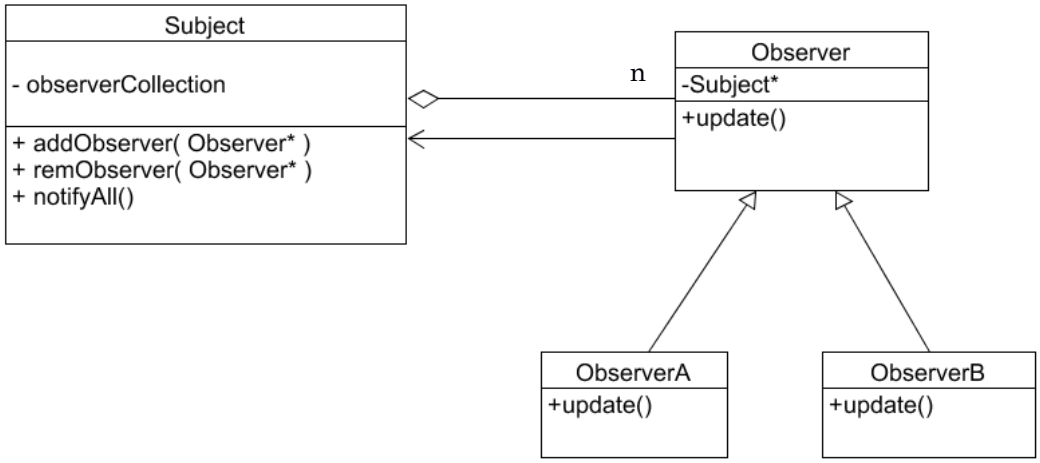


Diagrama básico del patrón observador:



El objeto observado, contiene atributos mediante los cuales cualquier objeto observador o vista se puede suscribir a él pasándole una referencia a sí mismo. El sujeto mantiene así una colección de las referencias a sus observadores.

Los observadores a su vez están obligados a implementar unos métodos determinados mediante los cuales el sujeto es capaz de notificar a sus observadores "suscritos" los cambios que sufre para que todos ellos tengan la oportunidad de refrescar el contenido representado.

```
class Subject {
          …
   public:
       void addObserver( Observer* );
       void remObserver( Observer* );
   private:
       void notifyAll();
};

 void Subject::notifyAll()  {
      for (all registered observers)
       { observer->update(); }
 }

class Observer {
 public:
 virtual void update() = 0;
 };
```

Utilice el patrón de observador en cualquiera de las siguientes situaciones:

- Cuando un cambio de un objeto requiere cambiar a los demás, y no se sabe cuántos objetos hay que cambiar.
- Cuando una abstracción tiene dos elementos, uno dependiente del otro. Encapsular estos aspectos en objetos separados le permite variar y reutilizar de manera independiente.