1. **Mô tả ví dụ sử dụng pattern Observer**

* Trong 1 công ty, Developers và Boss là việc với nhau nhưng những thay đổi của Developer trên Database lại không được cập nhật, thông báo cho Boss biết. Vấn đề đặt ra lúc này là phải dùng biện pháp nào đó mà khi Developer thực hiện những thay đổi trên Database thì những cập nhật đó phải được thông báo cho Boss.
* Chúng ta sử dụng pattern Observer và register(đăng ký) để Boss là người quan sát (Observer) Database. Boss sẽ nhận thông báo tự động cho mỗi lần có update Database. Chúng ta không cần gửi demo cho Boss theo cách thủ công, tất cả được làm tự động.
* Observer cần phải đăng ký với ISubject(đây là đối tượng mà Observer cần nhận thông tin về nó)
* Trong Observer pattern, cần phải định nghĩa ra interface cho subject và observer, chúng sẽ làm việc với nhau thông quan interface(ISubject, IObserver)
* ISubject phải cho phép các Observers có thể đăng ký và nhận thông báo khi có sự kiện xảy ra. Theo subject interface thì 3 method mà subject class cần thực thi đó là *registerObserver()*: Đăng ký observer*, removeObserver():* Hủy đăng ký observer*,*và *notifyObservers():* Thông báo cập nhật*.*

+ Khi muốn đăng ký một observer với subject thì cần gọi phương thức registerObserver của subject và truyền vào pointer đến observer. Sau đó, subject – đối tượng của class Database sẽ call hàm push\_back của vector để thêm pointer đó vào vector mObservers(biến lưu thông tin các Observers) để quản lý.

+ Khi muốn xóa một observer đã đăng ký thì tìm pointer của observer đó trong mObservers và xóa nó đi

+ Khi Developers làm gì đó với cơ sở dữ liệu thì Developers sẽ gọi phương thức *editRecord* của class Database

+ Những thay đổi sẽ được update cho Boss thông qua phương thức *notifyObservers()*

1. **Mô tả ví dụ sử dụng pattern Mediator**

* Ta xây dựng bài toán theo mô hình “*chatbox*”, tin nhắn sẽ không được gửi trực tiếp giữa 2 người mà cần phải thông qua trung gian là Mediator. Hay nói cách khác cuộc giao tiếp giữa 2 người sẽ được điều phối bởi Mediator.
* Lớp ConcreteMediator() sẽ nhận thông tin từ 2 người dùng gửi đến thông qua phương thức sendMessage(), sau đó thông tin sẽ được chuyển đến Mediator() và Colleage() để xử lý, thông tin tiếp tục được chuyển về cho người dùng
* Tin nhắn của người được gửi thông qua phương thức sendMessage() sẽ được lưu vào biến \_mediator của lớp Colleage, sau đó tin nhắn được lưu ở biến \_mediator sẽ được trả ngược về cho người dùng đích thông qua phương thức getMessage().