增加了抽象通知者，可以是接口，也可以是抽象类。

```
//通知者接口
abstract class Subject{
    protected String name;
    public Subject(String name){
        this.name = name;
    }
    //同事列表
    private ArrayList<Observer> list = new ArrayList<Observer>();//针对抽象的Observer编程
    private String action;
    //增加同事（有几个同事需要秘书通知，就增加几个对象）
    public void attach(Observer observer){
        list.add(observer);
    }
    //减少同事
    public void detach(Observer observer){
        list.remove(observer);
    }
    //通知
    public void notifyEmployee(){
        //给所有登记过的同事发通知
        for(Observer item : list){
            item.update();
        }
    }
    //得到状态
    public String getAction(){
        return this.action;
    }
    //设置状态（就是设置具体通知的话）
    public void setAction(String value){
        this.action = value;
    }
}
```

具体的通知者类可能是前台秘书，也可能是老板，它们也许有各自的一些方法，但对于通知者来说，它们是一样的，所以它们都去继承这个抽象类Subject。

```
//老板
class Boss extends Subject{
    public Boss(String name){
        super(name);
    }

    //拥有自己的方法和属性
}

//前台类
class Secretary extends Subject{
    public Secretary(String name){
        super(name);
    }

    //拥有自己的方法和属性
}
```

对于具体的观察者，需更改的地方就是把与“前台秘书”耦合的地方都改成针对抽象通知者。