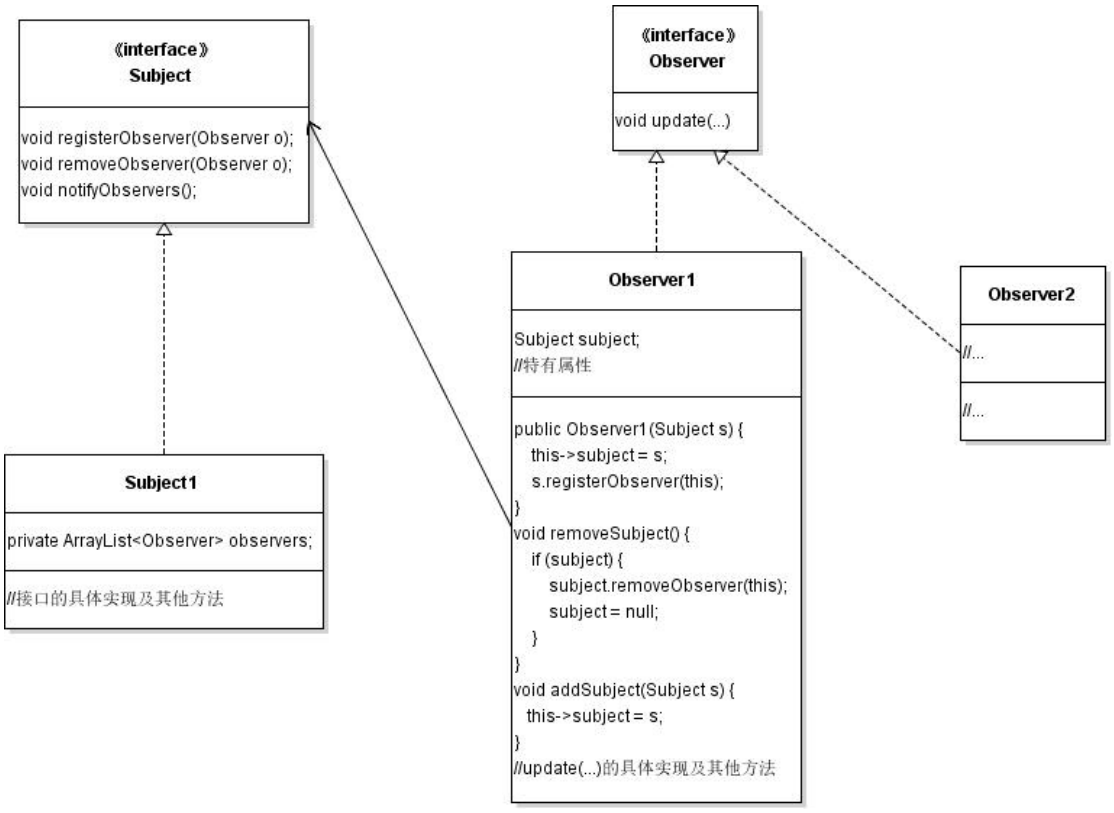
观察者模式（observer pattern）
首先看看报纸的订阅过程：
①报社的业务是出版报纸。
②客户向报社订阅报纸，只要报社有新报纸出版就会向所有客户发送。
③当客户不想再看报纸的时候，可以取消订阅。
观察者模式就是报纸的订阅过程，在观察者模式中，报社被称为“主题”，客户被称为“观察者”。主体对象用来管理某些数据，一旦数据发生改变，就会通知所有的观察着对象（已经订阅或注册该主题）。
定义：**观察者模式定义了对象之间的一对多依赖，当一个对象改变状态时，它的所有依赖着都会收到通知并自动更新。**
由面对接口编程，所有具体的观察者都应该实现一个统一的接口，这个接口中实现一个update()方法，而在主体对象中维护这样一个接口列表，每当数据改变时通过接口的 update()方法通知每个观察者，并且主题对象应该包含注册和取消订阅方法，使新的客户能够订阅该主题成为一个观察者，而旧的观察者能够取消订阅。这样一来，主题和观察者之间实现了松耦合，主题只知道观察者实现了某个接口，而不需要知道观察者是谁以及它的任何细节，并且在任何时候，我们都可以增加新的观察者，只要它实现了特定的接口，而无需改动主题代码。<设计原则：**为了交互对象之间的松耦合设计而努力，降低对象之间的互相依赖**>
观察者模式设计图如下：



主题可以采用“推”的方式将数据交给观察者，也就是在 update()方法中以参数的形式将所有的数据推给所有的观察者，但是当观察者所需要的数据各不相同而且数据很多时这种方式显得很繁琐，这时可以采用“拉”的方式，也就是在主题中提供一些公开的 getter()方法，在观察者的 update()方法中调用主题的 getter()方法获取所需数据。