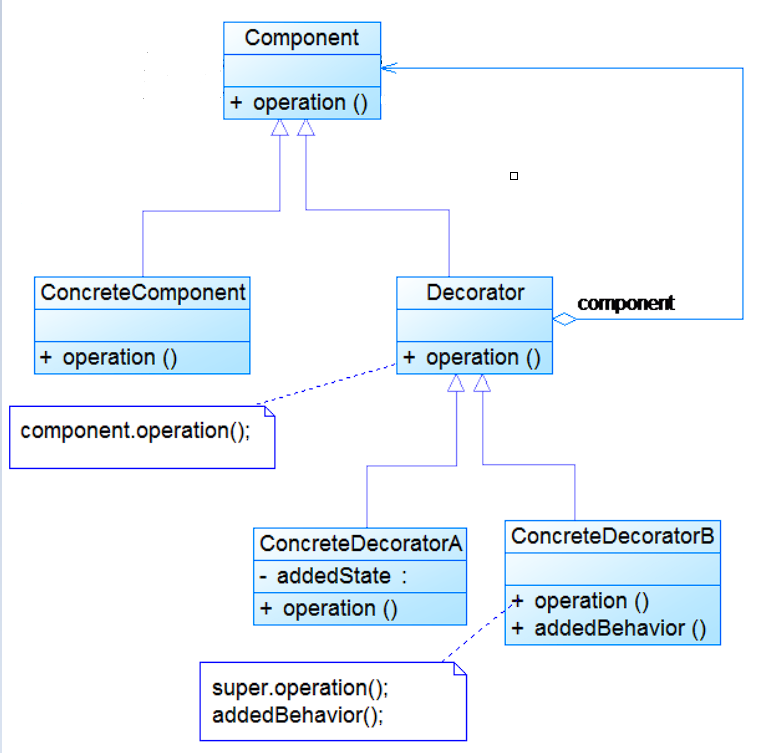
1. 模式定义

装饰模式：动态地给一个对象增加一些额外的职责（Responsibilitv)，就增加

对象功能来说，装饰模式比生成子类实现更为灵活。其别名也可以称为包装器（wropper)，与适配器模式的别名相同，但它们适用于不同的场合。根据翻译的不同，装饰模式也有人称之为“油漆工模式”，它是一种对象结构型模式。



**二、**模式动机

装饰模式以对客户透明的方式动态地给一个对象附加上更多的责任，换言之，客户端并不会觉得对象在装饰前和装饰后有什么不同。装饰模式可以在不需要创造更多子类的情况下，将对象的功能加以扩展。这就是装饰模式的模式动机。

三、例子

多重加密系统

四、优缺点

优点：

·装饰模式与继承关系的目的都是要扩展对象的功能，但是装饰模式可

以提供比继承更多的灵活性。

·可以通过一种动态的方式来扩展一个对象的功能，通过配置文件可以

在运行时选择不同的装饰器，从而实现不同的行为。

·通过使用不同的具体装饰类以及这些装饰类的排列组合，可以创造出

很多不同行为的组合。可以使用多个具体装饰类来装饰同一对象，得

到功能更为强大的对象。

·具体构件类与具体装饰类可以独立变化，用户可以根据需要增加新的

具体构件类和具体装饰类，在使用时再对其进行组合，原有代码无须

改变，符合“开闭原则”。

缺点：

·使用装饰模式进行系统设计时将产生很多小对象，这些对象的区别在于它们之间相互连接的方式有所不同，而不是它们的类或者属性值有所不同，同时还将产生很多具体装饰类。这些装饰类和小对象的产生将增加系统的复杂度，加大学习与理解的难度。

·这种比继承更加灵活机动的特性，也同时意味着装饰模式比继承更加易于出错，排错也很困难，对于多次装饰的对象，调试时寻找错误可能需要逐级排查，较为烦琐。

五、适用环境

在不影响其他对象的情况下，以动态、透明的方式给单个对象添加职责。需要动态地给一个对象增加功能，这些功能也可以动态地被撤销。

当不能采用继承的方式对系统进行扩充或者采用继承不利于系统扩展和维护时。不能采用继承的情况主要有两类：第一类是系统中存在大量独立的扩展，为支持每一种组合将产生大量的子类，使得子类数目呈爆炸性增长；第二类是因为类定义不能继承(final)。

六、示例代码