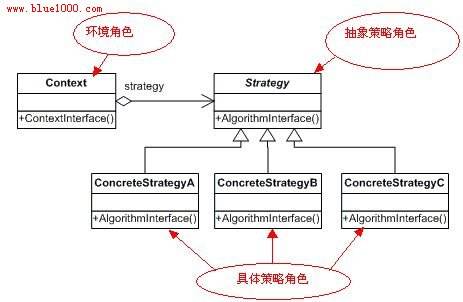
设计原则：

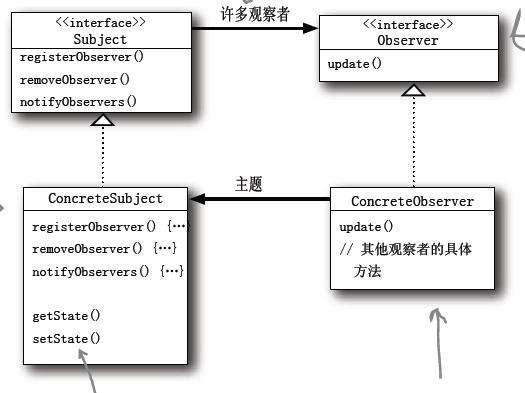
1. 找出应用中可能需要变化之处，把他们独立出来，不要和那些不需要变化的代码混在一起（开放封闭原则）把会变化的部分取出并封装起来，以便以后可以轻易地改动或扩充此部分，而不影响不需要变化的其他部分。所有的模式都提供了一套方法让“系统中的某部分改变不会影响其他部分”。
2. 针对接口编程而不是针对实现编程
3. 多用组合，少用继承
4. 为了交互对象之间的搜耦合设计而努力
5. 类应该对拓展开放，对修改关闭
6. 要依赖抽象，不要依赖具体类（依赖倒置原则）

设计模式：

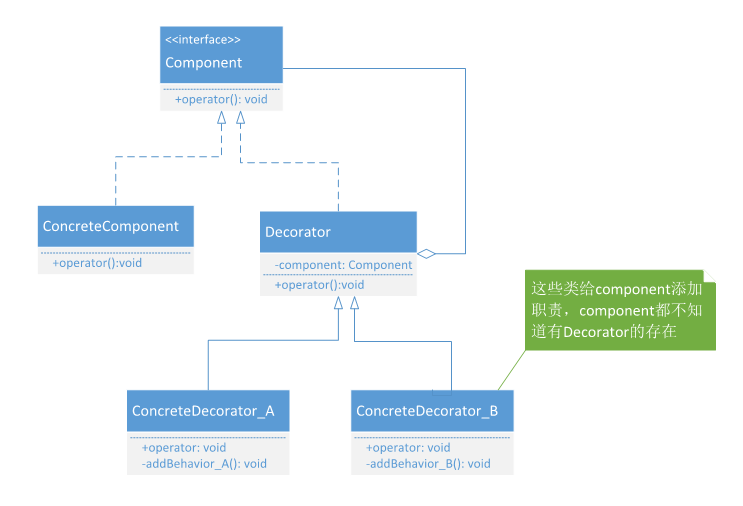
1. 策略模式定义了算法族，分别封装起来，让他们之间可以互相替换，此模式让算法的变化独立于使用算法的客户。



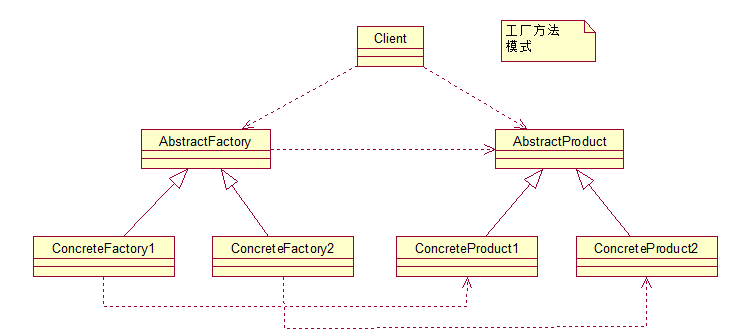
1. 观察者模式定义了对象之间的一对多依赖，这样一来，当一个对象改变状态时，它的所有依赖者都会受到通知并自动更新。



1. 装饰者模式动态地将责任附加到对象上。若要拓展功能，装饰者提供了比继承更有弹性的替代方案



1. 工厂方法模式定义了一个创建对象的接口，但子类决定要实例化的类是哪一个。工厂方法让类把实例化推迟到子类



5、抽象工厂模式提供一个借口，用于创建相关或依赖对象的家族，而不需要明确指定具体类

