CS 335 — Software Development

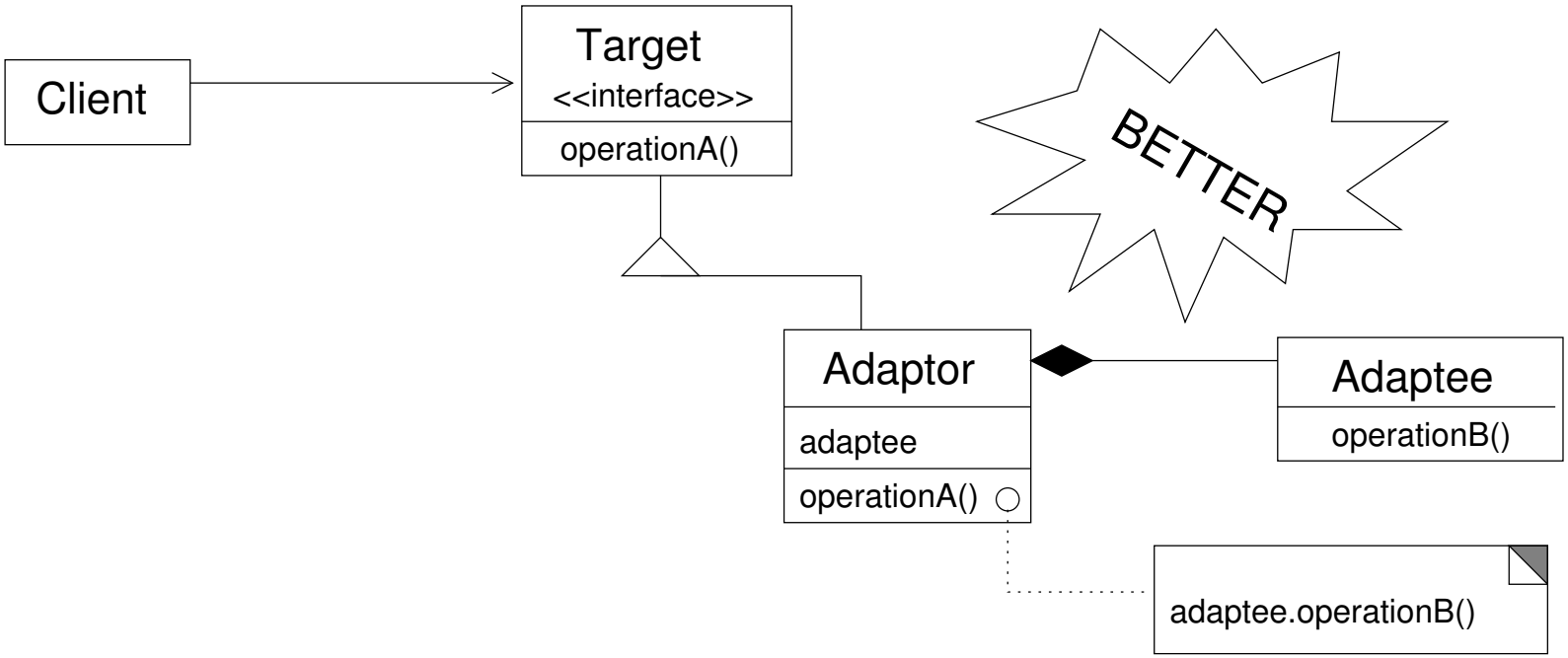
"Several Patterns": Adapter, Decorator, Composite, and Bridge

Feb 24, 2014

Adapter pattern

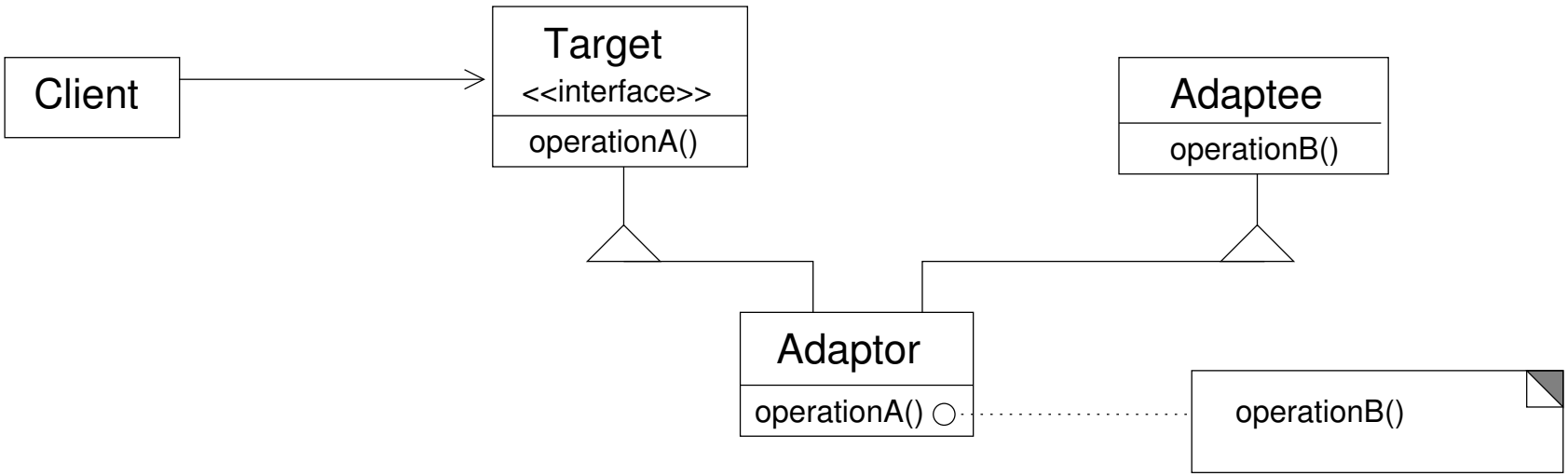
Convert the interface of a class into another interface that clients expect. Adapter lets classes work together that couldn't otherwise because of incompatible interfaces. [DP, pg 139]

(Object) Adapter pattern



Compare DP pg 140.

(Class) Adapter pattern

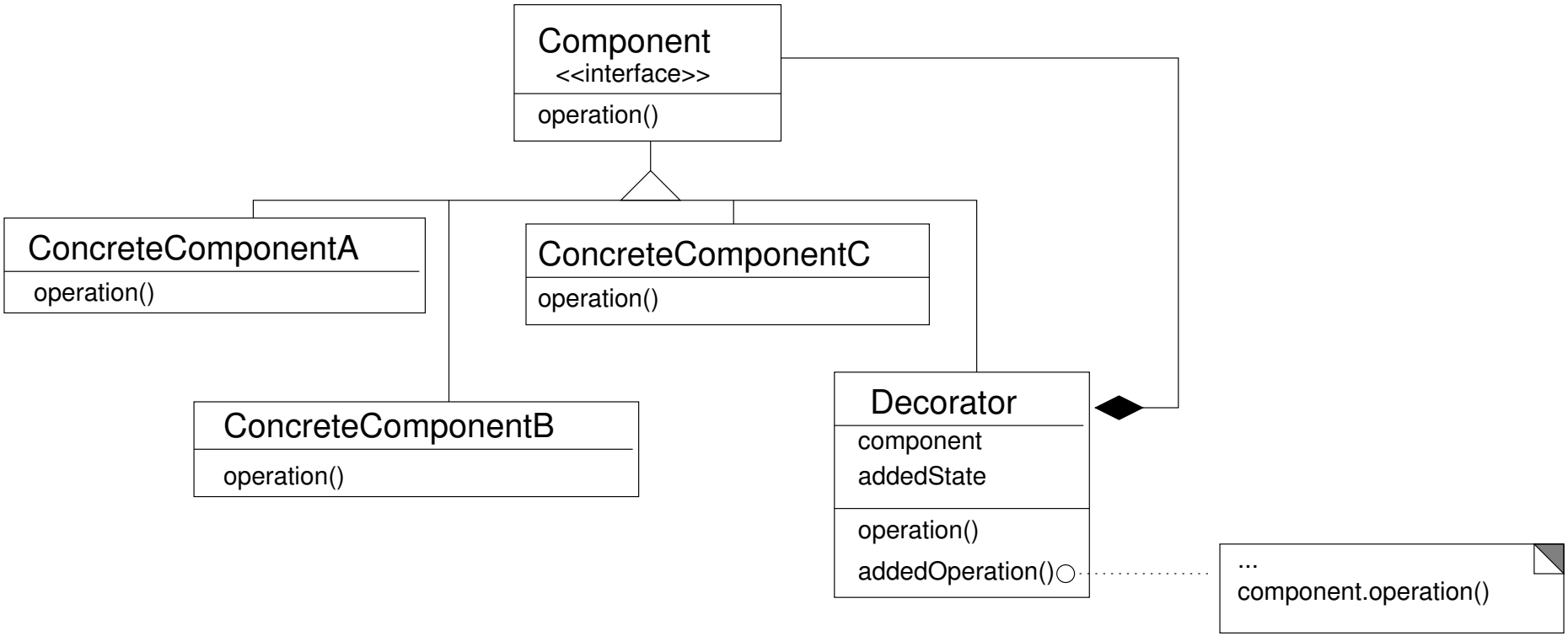


Compare DP pg 140.

Decorator pattern

Attach additional responsibilities to an object dynamically. Decorators provide a *flexible alternative* to subclassing for extending functionality. [DP, pg 175; emphasis added]

Decorator pattern

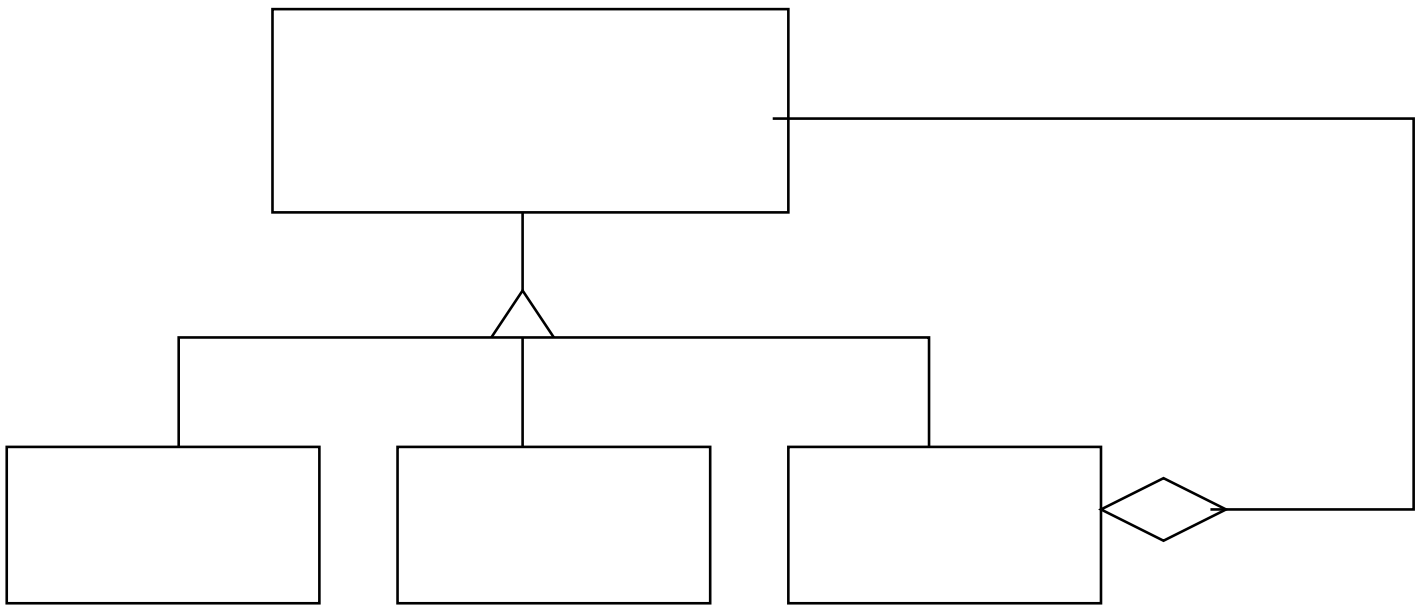


Compare DP pg 177.

Composite pattern

Compose objects into tree structures to represent part-whole hierarchies. Composite lets clients treat individual objects and compositions of objects uniformly. [DP, pg 163]

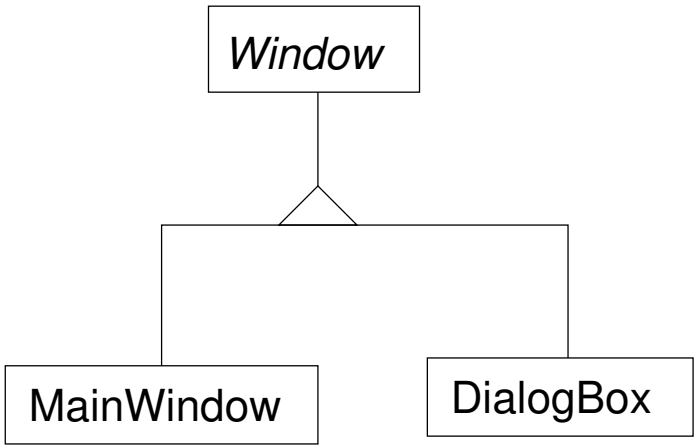
Composite pattern

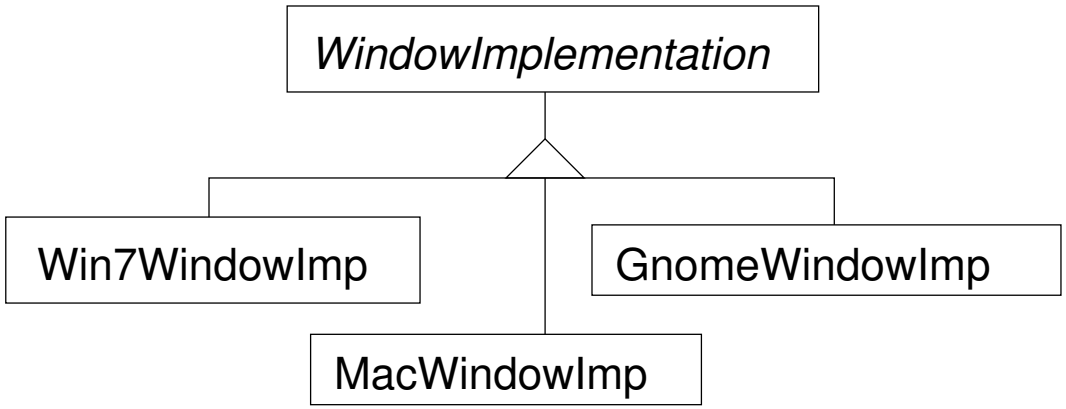


Bridge pattern

Decouple an abstraction from its implementation so that the two can vary independently. [DP pg 151]

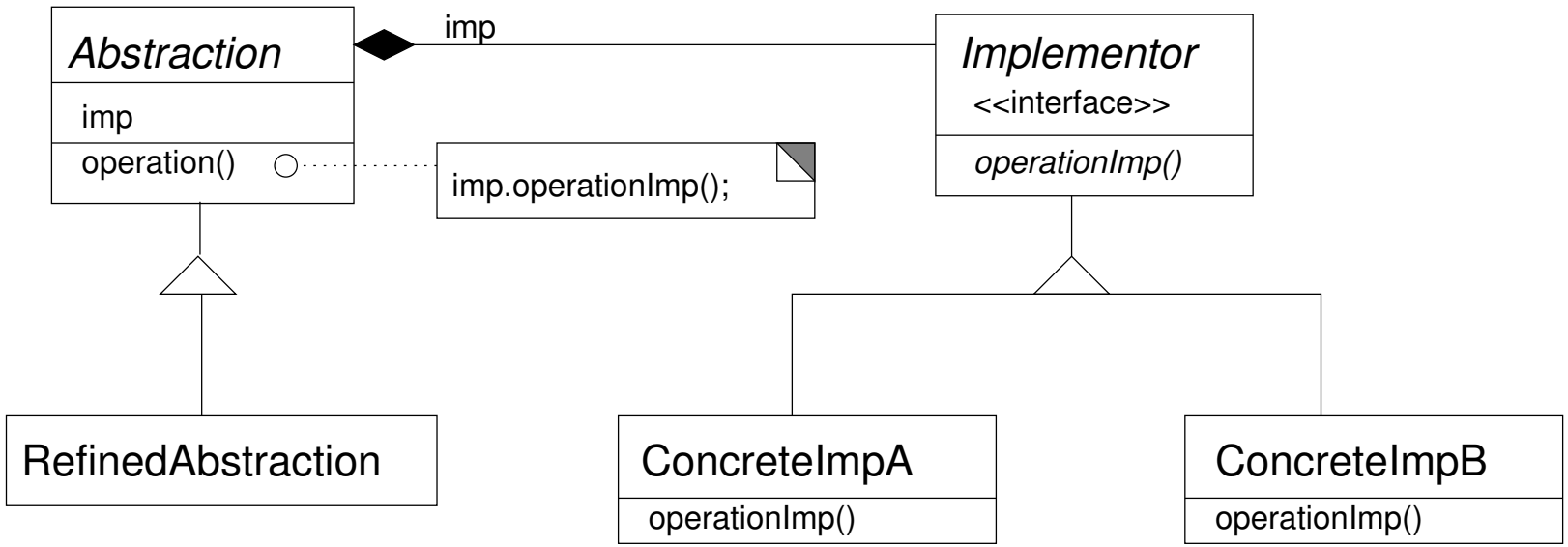
Bridge pattern





Compare DP pg 151.

Bridge pattern



Compare DP pg 152.

Bridge pattern