Object Oriented Analysis & Design
面向对象分析与设计

Lecture_09 GOF 设计模式（一）

1）单实例　　2）适配器　　3）外观　　4）观察者

主讲: 姜宁康 博士

4、GOF设计模式三: 外观模式 Facade

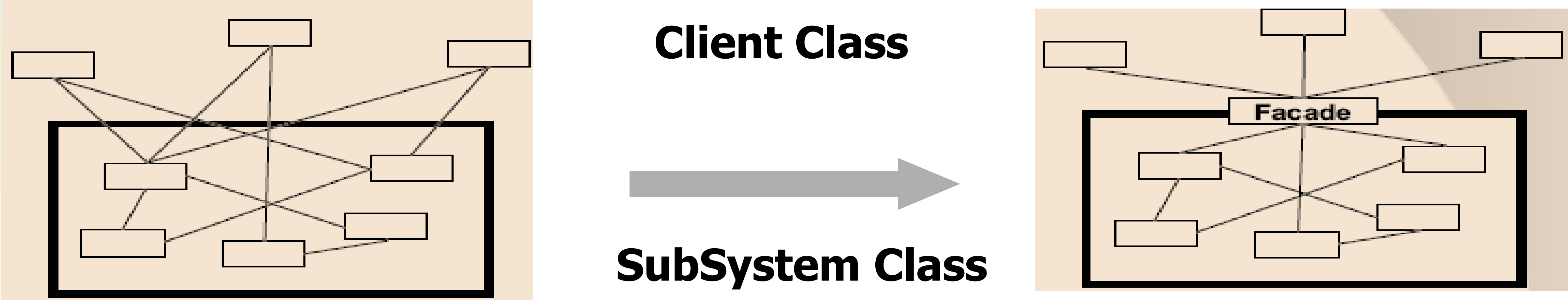
- “现有系统”功能强大、复杂，开发“新系统”需要用到其中一部分，但又要增加一部分新功能，该怎么办?

4.1 Facade Pattern: Key Features

Intent	**You want to simplify how to use an existing system. You need to define your own interface. 现有系统的接口比较复杂，你希望利用原有的功能重新定义新的接口**
Problem	**You need to use only a subset of a complex system. Or you need to interact with the system in a particular way. 如果只希望使用现有系统的部分功能、或者以一种特殊的方式与现有系统交互**
Solution	**The Facade presents a new interface for the client of the existing system to use. 外观模式提供一套新的接口供客户使用现有系统**
Participants and collaborators	**It presents a simplified interface to the client that makes it easier to use. 它呈现给客户一套简单地接口，易于使用。**
Consequences	**The Facade simplifies the use of the required subsystem. However, because the Facade is not complete, certain functionality may be unavailable to the client. 现有系统的部分功能，客户可能会不可用**
Implementation	**Define a new class (or classes) that has the required interface. Have this new class use the existing system. 新定义的类，使用了现有系统的功能，为客户提供所需要的接口**

4.1 Facade Pattern: Key Features

- 动机 Motivation
  - 最小化通信 Minimize communication
  - 最小化依赖 Minimize dependencies

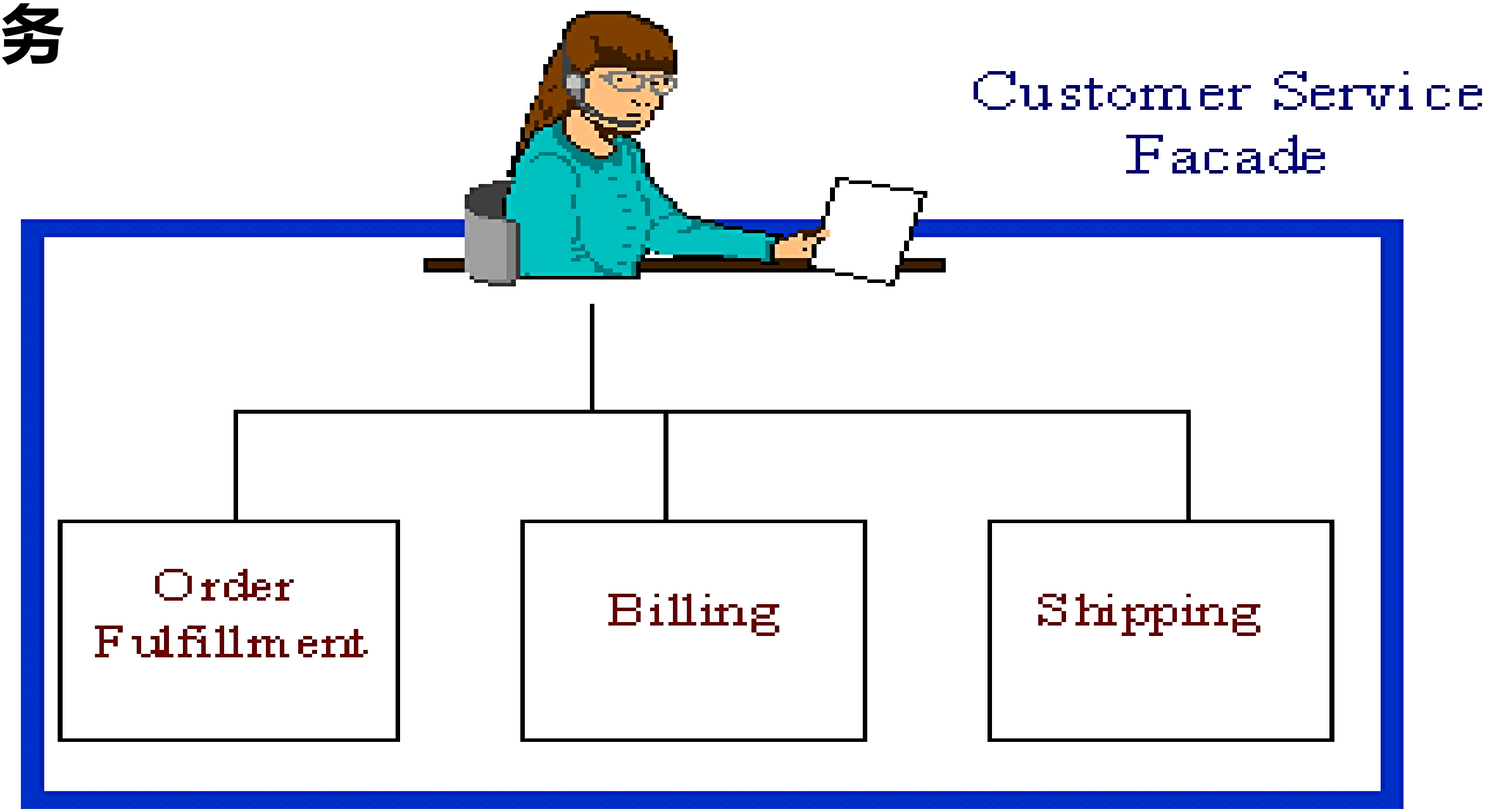


Client Class

SubSystem Class

4.2 外观模式的比喻 Non-software examples

- 例如，贸易公司的客户通过目录下订单，就类似一个“外观模式
  - 客户联系公司的客户联系代表
  - 客户联系代表 就作为“外观”，为客户提供接口，完成下单、付费、运输等公司内部复杂的功能
  - 客户很轻松完成任务



4.3 外观模式与适配器模式的比较

- 外观定义了新的接口，适配器使用旧的接口 Facade defines a new interface, whereas Adapter uses an old interface
- 适配器使得两个不一致的接口协同工作，而不是定义一个新的 Adapter makes two existing interfaces work together as opposed to defining an entirely new one
- 外观模式的本意是产生一个轻便的接口，适配器的本意是把现有的接口转换一下 The intent of Facade is to produce a simpler interface, and the intent of Adapter is to design to an existing interface
- 一个外观接口可能包装了多个现有系统的对象、也可能增加了一些新功能，而适配器只是包装一个对象 While Facade routinely wraps multiple objects and Adapter wraps a single object
- 多数情况下，外观是单实例的 Facade objects are often Singletons because only one Facade object is required
  - GRASP原则的外观控制器，就是外观模式

4.4 适用性 Applicability

- 简化复杂系统的使用 Provide a simple interface to a complex subsystem
  - You do not need to use all the functionality of a complex system and can create a new class that contains all the rules for accessing that system
  - If this is a subset of the original system, as it usually is, the API that you create for the new class should be much simpler than the original system's API
- 在使用复杂系统的某个功能时，可以加入新功能 You want to use the functionality of the original system and want to add some new functionality as well
- 为客户软件层提供一个接口，隐藏了现有系统的细节 Provide an interface to a software layer
  - You want to encapsulate or hide the original system
- 使客户与现有系统之间减少耦合、增加独立性、可移植性
- 节约成本
  - 开发新的外观类的成本，要比每个人都去熟悉现有系统、了解如何使用现有系统的功能要节省许多

- 本讲结束