面向对象的设计原则

授课教师：蓝天　　电子邮箱：lantian1029@uestc.edu.cn

面向对象设计的特点

面向对象设计强调定义软件对象，并且使这些软件对象相互协作来满足用户需求。

面向对象分析和设计的界限是模糊的，从面向对象分析到面向对象设计是一个逐渐扩充模型的过程。分析的结果通过细化直接生成设计结果，在设计过程中逐步加深对需求的理解，从而进一步完善需求分析的结果。

分析和设计活动是一个反复迭代的过程。

面向对象方法学在概念和表示方法上的一致性，保证了各个开发阶段之间的平滑性。

面向对象设计的四个层次

确定系统的总体结构和风格，构造系统的物理模型，将系统划分成不同的子系统。

中层设计：对每个用例进行设计，规划实现用例功能的关键类，确定类之间的关系。

进行底层设计：对每个类进行详细设计，设计类的属性和操作，优化类之间的关系。

补充实现非功能性需求所需要的类。

注意点

面向对象设计与结构化设计的过程和方法完全不同，要设计出高质量的软件系统，记住：

- 对接口进行设计
- 发现变化并且封装它
- 先考虑聚合然后考虑继承

强内聚

类内聚——设计类的原则是一个类的属性和操作全部都是完成某个任务所必须的，其中不包括无用的属性和操作。

- 例如设计一个平衡二叉树类，该类的目的就是要解决平衡二叉树的访问，其中所有的属性和操作都与解决这个问题相关，其他无关的属性和操作在这里都是垃圾，应该清除。

弱耦合

在面向对象设计中，耦合主要指不同对象之间相互关联的程度。如果一个对象过多地依赖于其它对象来完成自己的工作，则不仅使该对象的可理解性下降，而且还会增加测试、修改的难度，同时降低了类的可重用性和可移植性。

对象不可能是完全孤立的，当两个对象必须相互联系时，应该通过类的公共接口实现耦合，不应该依赖于类的具体实现细节。

耦合方式

交互耦合

- 如果对象之间的耦合是通过消息连接来实现的，则这种耦合就是交互耦合。在设计时应该尽量减少对象之间发送的消息数和消息中的参数个数，降低消息连接的复杂程度。

继承耦合

- 继承耦合是一般化类与特殊化类之间的一种关联形式，设计时应该适当使用这种耦合。在设计时要特别认真分析一般化类与特殊化类之间继承关系，如果抽象层次不合理，可能会造成对特殊化类的修改影响到一般化类，使得系统的稳定性降低。另外，在设计时特殊化类应该尽可能多地继承和使用一般化类的属性和服务，充分利用继承的优势。

可重用性

软件重用是从设计阶段开始的，所有的设计工作都是为了使系统完成预期的任务，为了提高工作效率、减少错误、降低成本，就要充分考虑软件元素的重用性。重用性有两个方面的含义：

- 尽量使用已有的类，包括开发环境提供的类库和已有的相似的类；
- 如果确实需要创建新类，则在设计这些新类时考虑将来的可重用性。

设计一个可重用的软件比设计一个普通软件的代价要高，但是随着这些软件被重用次数的增加，分摊到它的设计和实现成本就会降低。

框架

框架是一组可用于不同应用的类的集合。框架中的类通常是一些抽象类并且相互有联系，可以通过继承的方式使用这些类。

- 例如，Java应用程序接口（API）就是一个成功的框架包，为众多的应用提供服务，但一个应用程序通常只需要其中的部分服务，可以采用继承或聚合的方式将应用包与框架包关联在一起来获得需要的服务。

一般不会直接去修改框架的类，而是通过继承或聚合为应用创建合适的GUI类。

感谢观看！

授课教师：蓝天　　电子邮箱：lantian1029@uestc.edu.cn