软件工程—需求分析

类图

类、对象和它们之间的关系是面向对象技术中最基本的元素。类图技术是OOP方法的核心

类图标加上他们之间的关系就构成了**类图**

应用

类图用于对系统静态设计视图建模。与数据模型不同，它不仅显示了信息的结构，同时还描述了系统的行为。

类图中可以包含接口，包，关系等建模元素，也可以包含对象，链等实例

类图的典型应用在下面三类建模

1. 对系统的词汇建模
2. 对简单协作建模
3. 对逻辑数据库模式建模

通常包含以下内容

1. 类
2. 接口
3. 协作
4. 依赖，泛化和关联关系

类图可以包含注解和约束

类图可以包含有包或子系统，二者都用于把模型元素聚集成更大的组件

类主要包含以下几个部分

1. 名称（Name）名称是每个类所必有的构成，用于和其他类相区分
2. 属性（Attribute）类的一个组成部分描述了类所代表事物的属性
3. 操作（Operation）操作是对类的对象所能做的事务的抽象

UML约定

1.类名的首字母要大写，放在矩形的偏上部。

2.如果类名是由两个单词组成，那么将两个单词合并，第二个单词首字母大写。

3.正体字说明类是可被实例化的，斜体字说明类为抽象类

关系

依赖：它表示类之间的使用关系

例：有两个元素X、Y，如果修改元素X的定义可能会引起对另一个元素Y的定义的修改，则称元素Y依赖(Dependency)于元素X

泛化：它表示类之间的一般和特殊关系

泛化表示类与类之间的继承关系，接口与接口之间的继承关系。

关联：它表示对象之间的结构关系

对于两个相对独立的对象，当一个对象的实例与另一个对象的一些特定实例存在固定的对应关系时，这两个对象之间为关联关系

实现：它是规格说明和其实现之间的关系

大多数情况下，实现关系用来规定接口和实现接口的类或组件之间的关系

类图的抽象层次（考定义）

在软件开发的不同阶段使用的类图具有不同的抽象层次，一般地，类图可分为三个层次，即**概念层，说明层，实现层**

概念层：类图描述应用领域中的概念，一般地，这些概念与类有很自然的联系，但两者没有直接的映射关系

说明层：类图描述软件的接口部分，而不是软件的实现部分

实现层：真正考虑类的实现问题，揭示实现细节

非功能性需求的几个方面

1. 可靠性
2. 可用性：容易学习，使用效率，记忆性，错误恢复，主观满意度
3. 有效性：性能、可伸缩性、可扩展性
4. 可移植性

可靠性的衡量

1. 安全性
2. 事务性：原子性，一致性，隔离性，持久性
3. 稳定性

非功能性需求的来源

软件架构师是非功能性需求的主要来源

非功能性需求是一个迭代的过程

用例描述所包含的内容

1. 简要描述
2. 前置条件
3. 基本事件流
4. 其他事件流
5. 异常事件流
6. 后置条件