Lecture3 信息隐藏

1. 软件设计介绍

设计概述

- 设计是一个反复试验的过程
- 这个过程和这个过程的结果是不一样的
- 需求工程和设计之间存在交互作用

软件设计说明

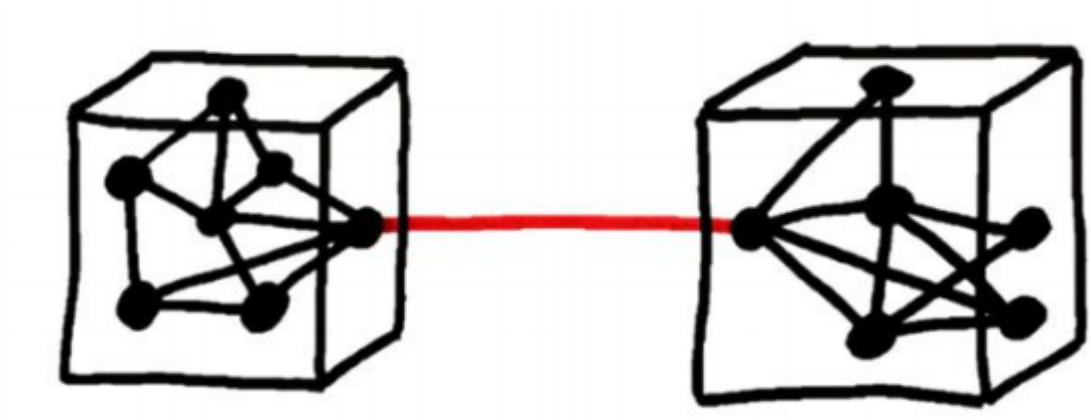
- 没有明确的公式
- 没有停止的规则
- 解决方案不是简单的对或错

2. 软件设计原则

抽象 Abstraction

- **过程抽象 Procedural abstraction**
  - 逐步求精的自然结果
  - 过程的名称表示操作的顺序
- **数据抽象 Data abstraction**
  - 目标是在数据中找到层次结构 例如，从通用数据结构到面向应用程序的数据结构

模块化、耦合和内聚 Modularity, coupling, and cohesion



模块化确定了结构标准，这些标准告诉了关于单个模块及其相互连接的一些事情

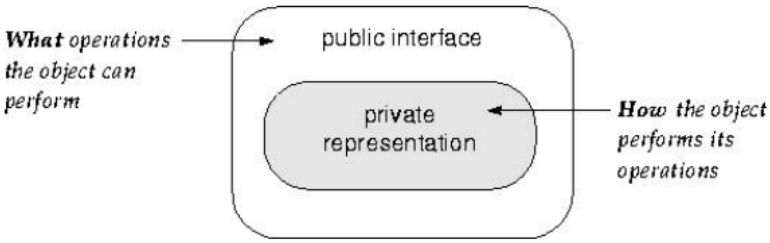
关键概念：内聚和耦合

- 内聚性 Cohesion：使模块保持在一起的粘合剂
- 耦合 Coupling：模块之间连接的强度

3. 信息隐藏 Information hiding

- 信息隐藏是一种将程序分解成模块的原理
- 在模块之间的接口中应该出现的唯一假设是那些接口**不太容易改变**
    - 易变的东西最好封装在模块里，改变的时候只用关心自己模块的变化，不需要考虑其它 的模块
    - 高内聚、低耦合
- 设计涉及一系列决策
    - 对于每个决定，考虑其他模块需要知道什么，其他模块可以不知道什么
- 信息隐藏与下面的三点高度相关
    - 抽象：如果你隐藏了一些东西，用户可能会从那个事实中抽象出来
    - 耦合：这个秘密减少了模块和环境之间的耦合
    - 内聚：秘密是什么将模块的各个部分绑定在一起

信息隐藏的动机



- 软件工程的一个基本成本是**适应变化**
    - 需要修改更多模块的更改比单独修改单个模块的更改成本更高
- 目标
    - 预测可能的变化
    - 定义捕获稳定方面的接口和捕获可变方面的实现

信息隐藏的简单示例

```
1   double sqrt(int)
```

- 可以用二分法、牛顿法、因式分解等方法实现
- 客户端不关心（或不需要知道）它是如何实现的
- 实现应该能够完全改变而不影响客户端代码（并且只需要重新链接）

信息隐藏的类型

- 算法（过程的抽象）
- 数据的表达（抽象数据类型）
- 硬件设备的特征（虚拟机、硬件抽象层等）
- 信息从哪里获得（使用哪些搜索引擎）
- 用户界面

分解的准则

- 按功能分解
  - 按主要加工工序分解
- 信息隐藏分解
  - 每个模块都有一个对其他模块隐藏的设计决策
  - 被选择和设计的接口尽可能少地揭示隐藏的信息

信息隐藏总结

- 决定哪些设计决策可能会改变，哪些可能会改变
- 将每个可能更改的设计决策放在自己的模块中
- 为每个模块分配一个接口，该接口隐藏可能更改的决策，只公开稳定的设计决策
- 确保模块的客户端只依赖稳定的接口，而不是实现
- 好处：是否能够正确地预测什么可能改变并且正确的隐藏信息，每一次的改变只影响一个模块