1. 打好基础

第1章 欢迎进入软件构建的世界

\*软件开发过程

+定义问题（problem definition）

需求分析（requirements development）

规划构建（construction planning）

软件架构（software architecture），或 高层设计（high-level design）

详细设计（detailed design）

编码与调试（coding and debugging）

单元测试（unit test）

集成测试（integration testing）

集成（integration）

系统测试（system testing）

保障维护（corrective maintenance）

第2章 用隐喻来更充分地理解软件开发

\*

1. 三思而后行：前期准备

\*功能需求相关注意点：

+输入； 包括来源，精度，取值范围，出现频率

+输出；目的地，精度，取值范围，出现频率，格式

+通信接口；包括握手协议，纠错协议，通信协议

+计时相关；处理时间，数据传输率，系统吞吐量

+安全级别

+可靠性；软件失灵的后果，发生故障时需要保护的至关重要信息，错误检测与恢复策略

机器内存和剩余磁盘空间的最小值

1. 关键的“构建”决策

\*

第二部分 创建高质量的代码

第5章 软件构建中的设计

\*高质量设计的特征

+最小的复杂度（Minimal complexity）

+易于维护（Ease of maintenance）

+松散耦合（loose coupling）:程序的各个组成部分之间关联最少。

+可扩展性（extensibility）

+可重用性（reusability）

+高扇入（high fan-in）:让大量的类使用某个给定的类。系统很好地利用了低层次工具类

+低扇入（low fan-in）:低扇入原则是有益的；高扇入使用大量其他类，变得过于复杂。

+可移植性（portability）

+精简性（leanness）:设计出的系统没有多余的部分

+层次性（stratification）

+标准技术（Standard techniques）

\*设计的层次：

+Level 1: 软件系统: 子系统或者包。

+Level 2: 分解为子系统或包。识别出所有子系统；比如说数据库，用户界面，业务规则，命令解释器，报表引擎。

+Level 3: 分解为类

+Level 4: 分解成子程序

+Level 5: 子程序内部的设计

\*记录你的设计成果

+把设计文档插入到代码里

+用Wiki来记录设计讨论和决策

+写总结邮件

+使用数码相机：把白板上画出的图表照成相片然后嵌入到传统的文档里

+保留设计挂图

+使用CRC(类，职责，合作者)卡片

+在适当的细节层创建UML图

第6章 可以工作的类

1. 高质量的子程序
2. 防御式编程

\*保护数据免遭非法输入数据的破坏

+1.检查所有来源于外部的数据的值： 文件，UI，网络，注入的SQL命令，HTML等

+2.检查子程序所有输入参数的值

+3.决定如何处理错误的输入数据

\*建立自己的断言机制