课程概况

- **教材:《软件工程导论(第6版)》,张海藩编著,清华大学出版社**

  第1章 软件工程学概述
  第2章 可行性研究
  第3章 需求分析
  第5章 总体设计
  第6章 详细设计
  第7章 实现
  第8章 维护
  第9章 面向对象方法学引论
  第10章 面向对象分析
  第11章 面向对象设计
  第12章 面向对象实现

- 课件、研讨题目、实验要求
(按课程进度陆续上传)

http://disk.lehu.shu.edu.cn

提取码：2016WSEYSZ

- 考核方式
  考勤和课堂表现：**10%**
  研讨：**15%**
  实验：**15%**
  书面考试：**60%**

- 教师联系方式

袁世忠

s.z.yuan@163.com

答疑安排：周四14:10 ~ 15:40，计1023室

第1章

软件工程学概述

1.1 软件危机

软件危机的介绍
产生软件危机的原因
消除软件危机的途径

软件危机介绍

- 软件危机指在计算机软件的开发和维护过程中，所遇到的一系列严重问题。

- 开发费用和进度难以估算和控制，大大超过预期的资金和规定日期

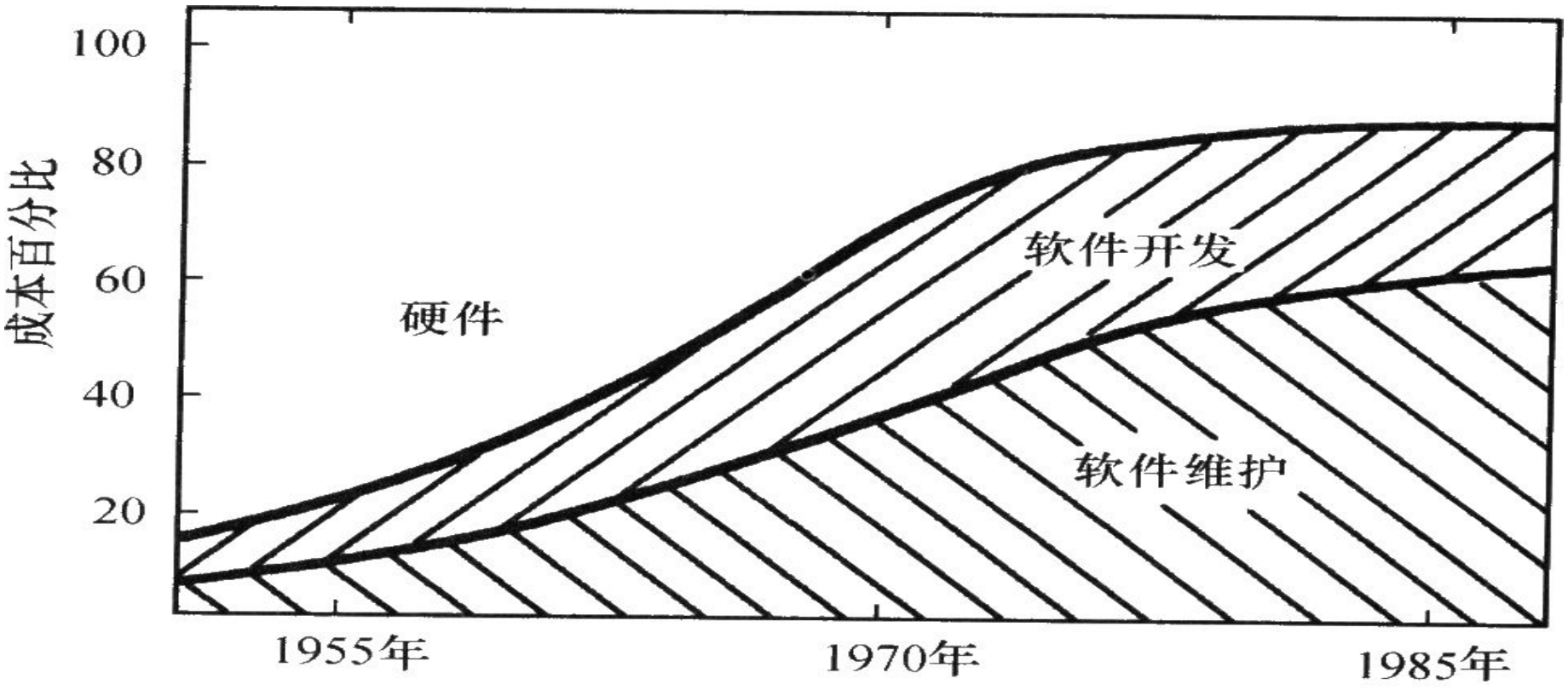
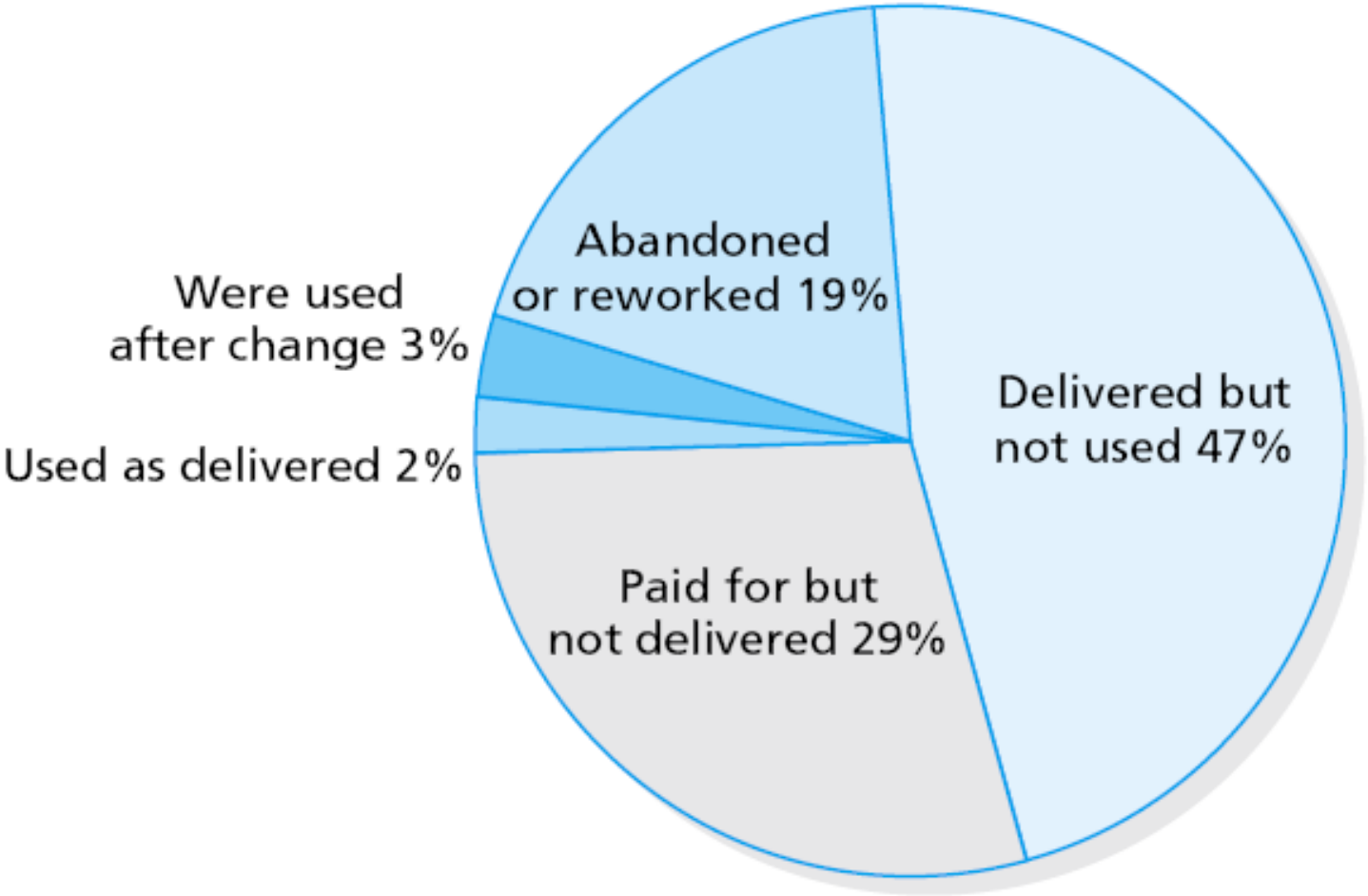


图 1.1　硬件/软件成本变化趋势

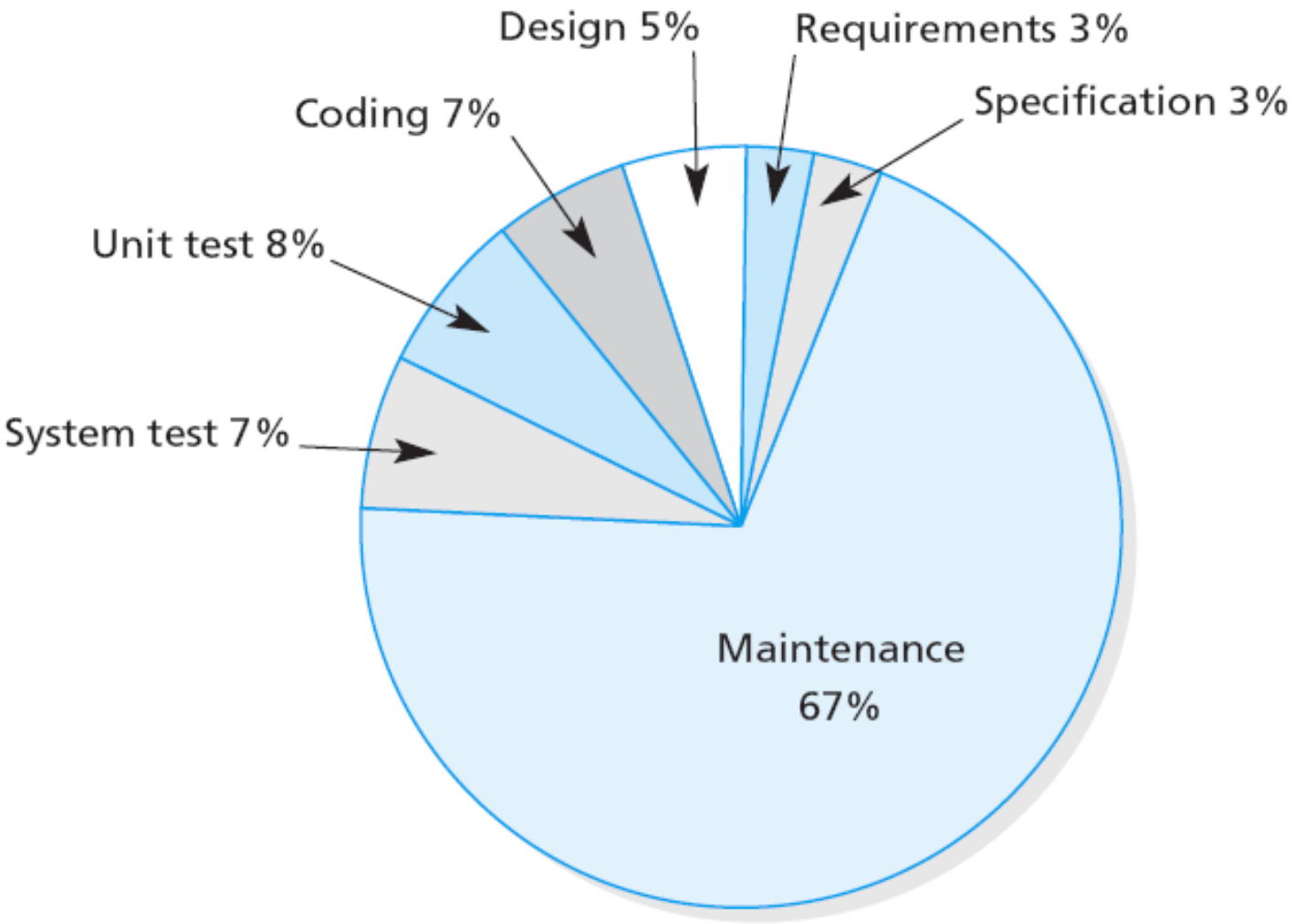
- 软件需求分析不够充分，用户不满意“已经完成”的软件系统



- 软件质量难于保证
  - 1963 年，美国用于控制火星探测器的计算机软件中的一个“，”号被误写为“.”，而致使飞往火星的探测器发生爆炸，造成高达数亿美元的损失。
  - 1966年，IBM 360 机的操作系统。花费 5000 人一年的工作量，写了近 1 万行代码。错误百出，每次的新版本就是从前一版本中找 1000个程序错误而修正的结果。

软件维护困难

- 通常没有保留适当的文档资料



产生软件危机的原因

- 与软件本身的特点有关
- 与软件开发与维护的方法不正确有关

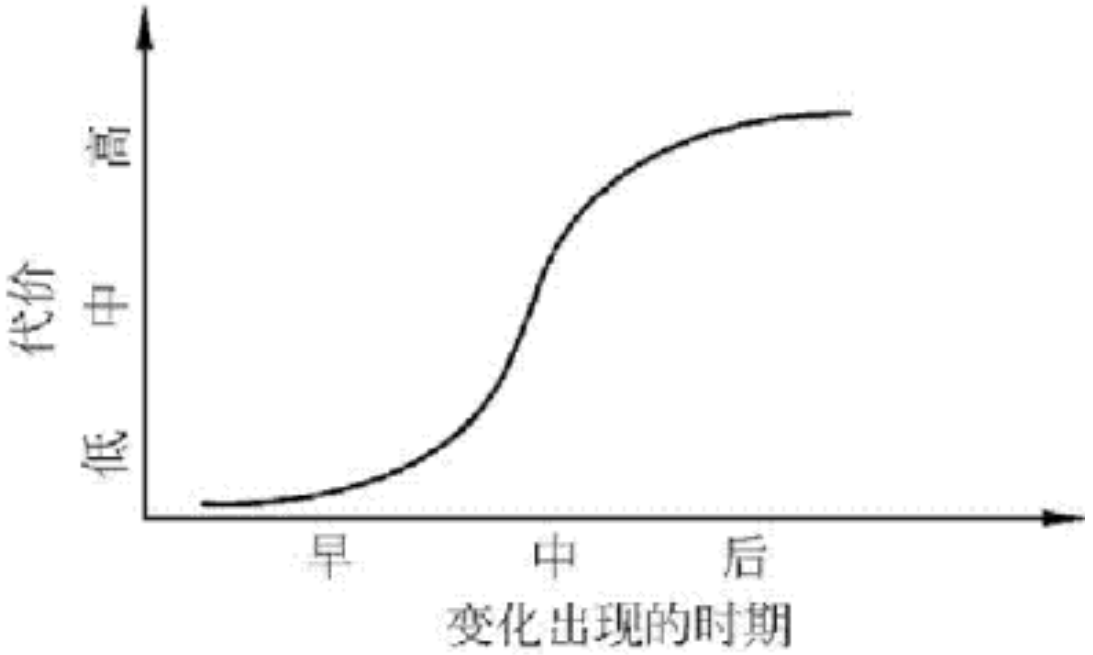
软件与一般程序不同

- 软件远比一般程序规模庞大，复杂性高。
- 大型软件开发既有技术问题，还有社会问题。
  - 开发团队成员分工合作
  - 技术与管理的矛盾
  - 软件开发人员对软件应用的领域知识的了解

类别	参加人员数	研制期限	产品规模（源程序行数）
微型	1	1～4 周	0.5k
小型	1	1～6 月	1k～2k
中型	2～5	1～2 年	5k～50k
大型	5～20	2～3 年	50k～100k
甚大型	100～1000	4～5 年	1M(=1000k)
极大型	2000～5000	5～10 年	1M～10M

软件开发和维护方法中存在的问题

- 缺乏有效的、系统的技术手段和管理方法
- 用户和软件开发人员的理解鸿沟
- 错误的认识和做法
  - 忽视软件需求分析的重要性
  - 认为软件开发就是写程序并设法使之运行
  - 轻视软件维护等



消除软件危机的途径

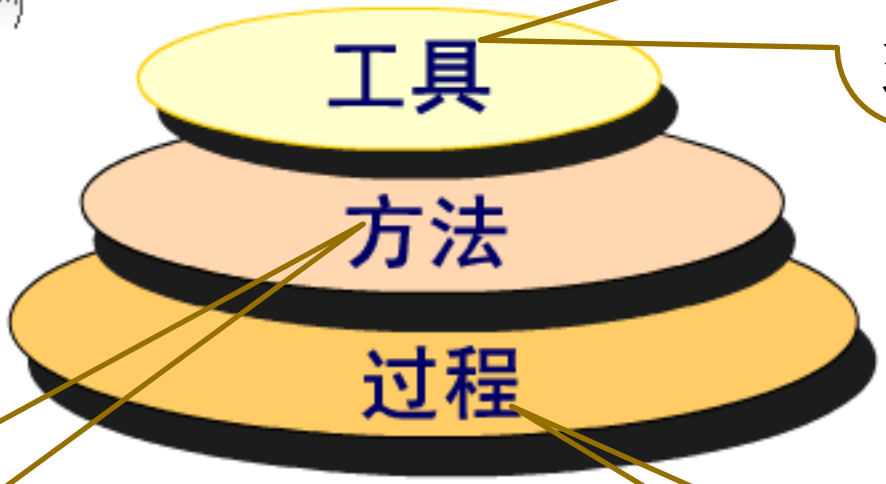
- **彻底消除“软件就是程序”的错误观念。**
  - 一个软件必须由一个完整的配置组成，事实上，软件是程序、数据及相关文档的完整集合。文档是开发、使用和维护程序所需要的图文资料。
- **充分认识到软件开发是一种组织良好、管理严密、各类人员协同配合、共同完成的工程项目，不是个人独立的劳动。**
- **推广和使用在实践中总结出来的软件开发的成功技术和方法。**
- **开发和使用更好的软件工具**

1.2 软件工程

软件工程的介绍
软件工程方法学

- 采用工程的概念、原理、技术和方法来开发与维护软件，把经过时间考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来，以经济地开发出高质量的软件并有效地维护它，这就是软件工程。

软件工程学的三个基本要素



为软件工程方法提供自动或半自动的软件支撑环境

完成软件开发的各项任务的技术方法

规定了完成各项任务的工作步骤

两种使用最广泛的软件工程方法学

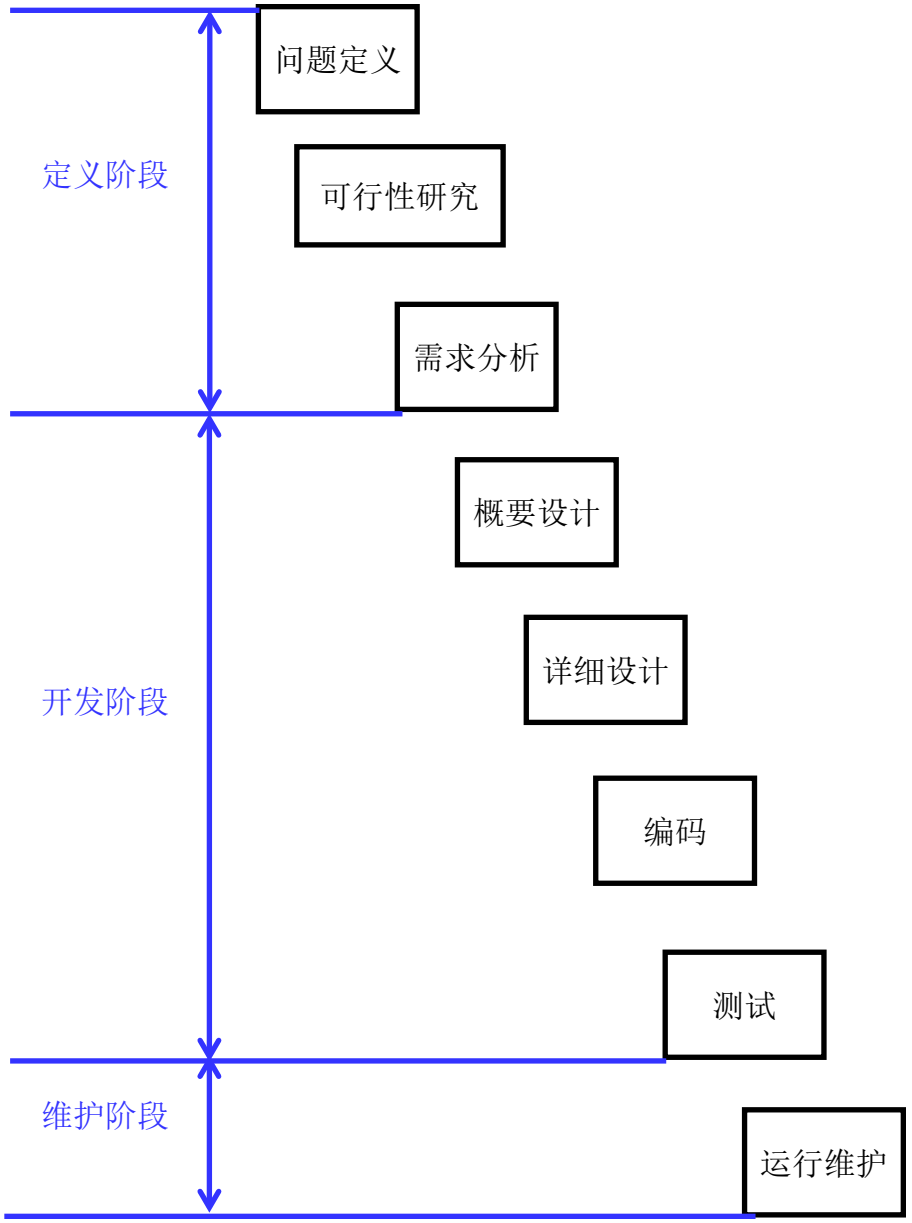
- **传统方法学**
  - 也称为生命周期方法学或结构化范型
  - 采用结构化技术(结构化分析、设计和实现)
  - 结构化范型要么面向行为，要么面向数据
- **面向对象方法学**
  - 把数据和行为看成同等重要，以数据为主线，把数据和对数据的操作紧密地结合
  - 面向对象方法=对象+类+继承+用消息通信

1.3 软件生命周期

软件产品或系统一系列相关活动的全周期

软件生命周期的各个阶段

- **软件定义**
  - 总目标、可行性；导出实现策略及系统功能；估计资源和成本，并且制定工程进度表。
- **软件开发**
  - 具体设计和实现
- **软件维护**
  - 使软件持久地满足用户的需要



1. **问题定义**
   - 要解决的问题是什么
   - 确定用户要求解决的问题、工程的目标和规模
2. **可行性研究**
   - 问题有行得通的解决办法吗
   - 经济可行性、技术可行性、法律可行性，不同的方案
3. **需求分析**
   - 为了解决问题，目标系统必须做什么
   - 确定系统必须具有的功能和性能，运行环境，预测发展前景
   - 规格说明书(specification)
4. **总体设计（概要设计）**
   - 概括地说，应该怎样实现目标系统
   - 设计实现目标系统的几种可能的方案，推荐一个最佳方案
5. **详细设计**
   - 怎样具体地实现目标系统
   - 设计程序的详细规格说明
6. **编码和单元测试**
   - 写出正确的易理解、易维护的程序模块并仔细测试每个模块
7. **综合测试**
   - 集成测试和验收测试，现场测试或平行运行
8. **软件维护**
   - 使系统持久地满足用户的需要
   - 改正性维护，适应性维护，完善性维护，预防性维护

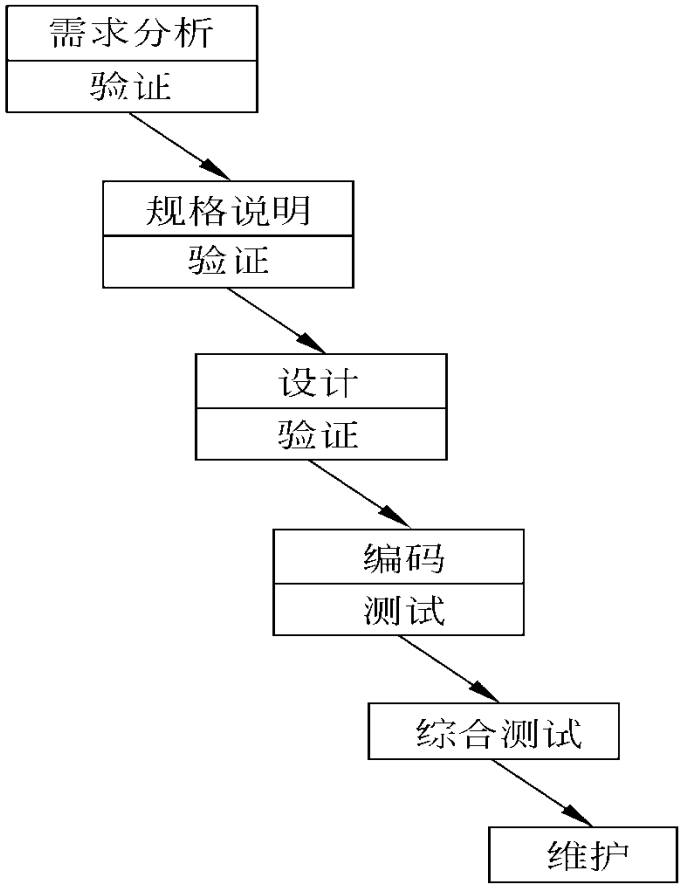
1.4 软件过程

什么是软件生命周期模型
典型的软件生命周期模型

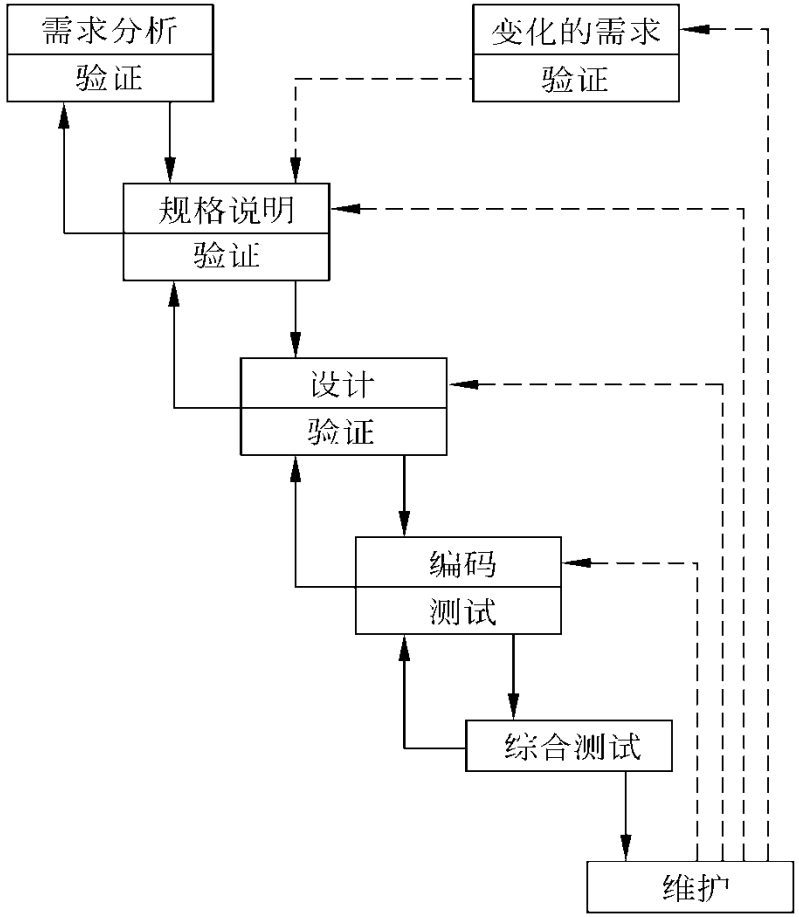
- 生命周期模型规定了把生命周期划分成哪些阶段及各个阶段的执行顺序，因此，也称为过程模型。
  - 任务框架，各项任务的工作步骤
  - “什么人”、“什么时候”、“做什么”以及“怎样做”
- 典型的过程模型
  - 瀑布模型(Waterfall model)
  - 快速原型开发模型(Rapid Prototyping model)
  - 增量模型(Incremental model)
  - 螺旋模型(Spiral model)
  - 喷泉模型(Fountain model)
  - 极限编程（XP，eXtreme Programming)
  - RUP（Rational Unified Process）

瀑布模型

- 理想的瀑布模型



- 实际的瀑布模型



- **瀑布模型的特点**
  - 阶段间具有顺序性和依赖性
  - 推迟实现的观点
  - 质量保证的观点(文档驱动)
    - 每个阶段都必须完成规定的文档
    - 每个阶段结束前都要对所完成的文档进行评审
- **瀑布模型的缺点**
  - 开发过程一般不能逆转，否则代价太大。软件的实际情况必须到项目开发后期才能看到。
  - 无法解决软件需求不明确或不准确的问题；可能导致最终开发的产品不能真正满足用户需要。

第1周研讨题

- **软件生命周期模型**
  - 瀑布模型的主要缺点
  - 快速原型模型的特点
  - 增量模型的特点
  - 螺旋模型的特点