第一章　软件工程介绍

王美红

主要内容

- 软件的本质
- 软件危机与软件工程
- 软件工程
- 软件过程
- 软件工程实践
- 软件神话

1.1 软件的本质

- 软件扮演着信息转换的角色：生产、管理、获取、修改、显示或者传输各种不同的信息
- 软件提供了计算机控制(操作系统)、信息通信（网络）以及应用程序开发和控制（软件工具和环境）的基础平台。

1.1 软件的本质

- 软件的定义：
  - （1）指令的集合（计算机程序），通过执行这些指令可以满足预期的特征、功能和性能需求
  - （2） 数据结构，使得程序可以合理利用信息
  - （3）软件描述信息，它以硬拷贝和虚拟形式存在，用来描述程序操作和使用。

软件角色的演化

- 计算机软件的地位在50多年的时间中发生了很大的变化：
  1. 20世纪70，80年代“新的工业革命”“信息和知识（由计算机控制）将成为21世纪能源的焦点”……
  2. 20世纪90年代，“反虚拟生活”“未来不是计算”被刻意妖魔化
  3. 20世纪90年代末，随着网络的意义逐步体现，千年虫等问题，计算机软件职业“复兴”
  4. 现代庞大的软件产业已经成为工业经济中的主导因素

软件与硬件的差异特征

- 软件是开发的，而不是传统意义上生产制造的（成本）
- 软件的非通用性
- 软件不会磨损

图1-1 硬件失效曲线图

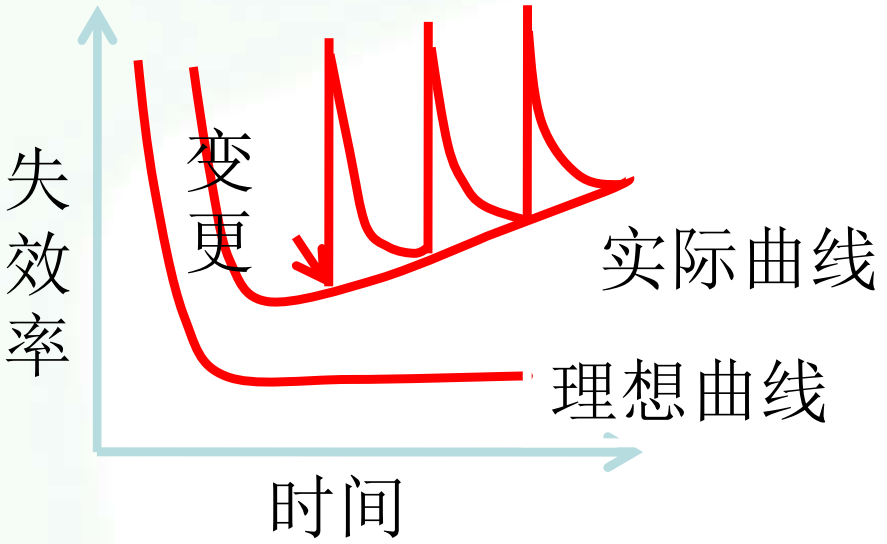


图1-2 软件失效曲线图

软件的应用领域

- 系统软件
- 应用软件
- 工程/科学软件
- 嵌入式软件
- 产品线软件
- Web/移动应用软件
- 人工智能软件

遗留软件

- 有些软件年代比较久，甚至过于久远了。
- 具有生命周期长以及业务关键性的特点，大多具有质量差的缺点
- 随着时间的推移，遗留软件基于下述原因演化：
  - 软件需求修改其适应性；软件必须根据新的业务需求进行升级；软件必须扩展以具有与更多现代化系统和数据库的协作能力；软件架构必须进行改进以适应多样化的网络环境

遗留软件

当这些变更发生时，遗留系统需要经过再工程以适应未来的多样性。

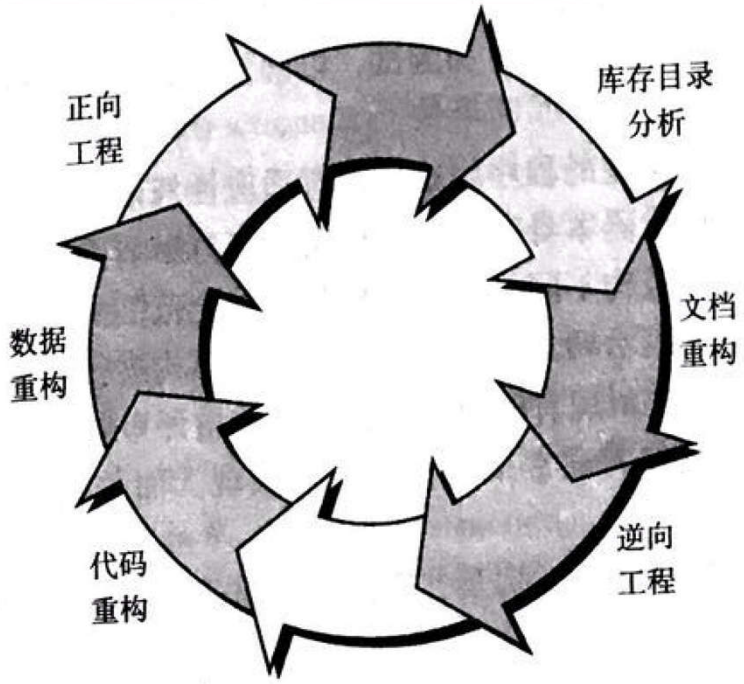


图 36-2　软件再工程过程模型

- 软件系统不断经历更新，新的软件系统从旧的软件系统中建立起来，并且……新旧所有系统都必须具有互操作性和协作性

1.2 软件危机与软件工程

- 软件工程的提出：
  - 软件工程主要是针对20世纪60年代的软件危机而提出的
- 软件危机定义：
  - 软件危机是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

软件危机案例

- IBM公司的 OS/360，共约100万条指令，花费了5000个人年；经费达数亿美元，而结果却令人沮丧，错误多达2000个以上，系统未能完全实现当初的设想。 OS/360系统的负责人Brooks这样描述开发过程的困难和混乱：“…像巨兽在泥潭中作垂死挣扎，挣扎得越猛，泥浆就沾得越多，最后没有一个野兽能够逃脱淹没在泥潭中的命运。…”

软件危机案例

- 1963年美国飞往火星的火箭爆炸，造成1000万美元的损失。

  原因是FORTRAN程序

  DO 5 I=1，3　　误写为：

  DO 5 I=1 . 3

- 1967年苏联“联盟一号”载人宇宙飞船在返航时，由于软件忽略一个小数点，在进入大气层时因打不开降落伞而烧毁。

软件危机——表现

- 软件危机的典型表现：
  1. 成本和进度估计常不准确
  2. 用户的满意度常不高
  3. 质量往往靠不住
  4. 软件通常很难维护
  5. 文档资料不完整、不合格
  6. 软件的成本高，所占比例逐年上升
  7. 软件开发生产率提高的速度慢

软件危机——原因

- 产生软件危机的原因：
  - 客观原因
    - 软件缺乏“可见性”，管理和控制其开发过程相对困难
    - 软件大多规模庞大，而复杂性随规模以指数速度上升

软件危机——原因

- 主观原因
  - 错误的认识和做法
    - 忽视软件需求分析的重要性—急于求成，仓促上阵
    - 认为软件开发就是写程序—编程只占全部工作量的10%--20%，软件配置主要包括程序、文档和数据
    - 轻视软件维护—维护费用占总费用的55%--70%

软件危机——消除

- 消除软件危机的途径：
  1. 应该树立对计算机软件的正确认识
     - 软件是程序、数据及文档的完整集合。
  2. 认识到软件开发不是某种个体劳动的神秘技巧
     - 是一种组织良好、管理严密、各类人员协调配合、共同完成的工程项目。要充分吸收和借鉴经验
  3. 推广使用在实践中总结开发出来的开发软件的成功的技术和方法。
  4. 应用和开发更好的软件工具。

1.3 软件工程

- IEEE对软件工程的定义:

（1）将系统化的、严格约束的、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件。

（2）在（1）中所述方法的研究。

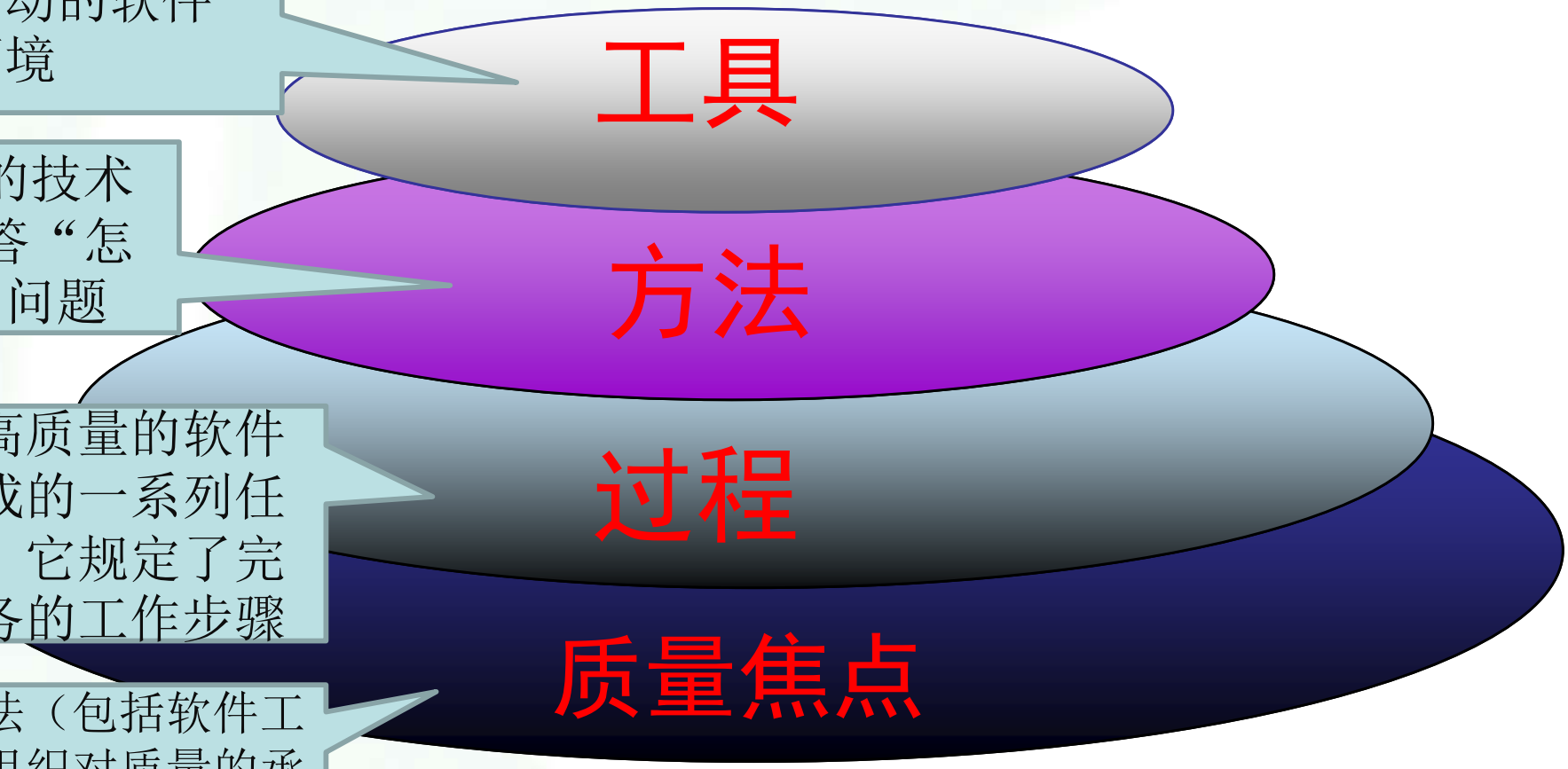
软件工程：一种层次化技术

为运用方法而提供的自动或半自动的软件工程支撑环境

各项任务的技术方法，回答“怎么做”的问题

为了获取高质量的软件所需要完成的一系列任务的框架，它规定了完成各项任务的工作步骤

任何工程方法（包括软件工程）必须以组织对质量的承诺为基础



1.4 软件过程

- 软件过程是工作产品构建时所执行的一系列活动、动作和任务的集合。
- 过程框架定义了若干个框架活动，为实现完整的软件工程过程建立了基础。
- 通用过程框架包含以下5个活动：
  - **沟通、策划、建模、构建、部署**
- 过程框架以普适性活动做补充：
  - 项目跟踪和控制、风险管理、软件质量保证、技术评审、测量、软件配置管理、可重复管理、工作产品的准备和生产

不同软件过程的差异

- 活动、动作和任务的总体流程，以及相互依赖关系
- 在每一个框架活动中，动作和任务的细化程度
- 工作产品的定义和要求的程度
- 质量保证活动应用的方式
- 过程描述的详细程度和严谨程度
- 客户和利益相关者项目参与的程度
- 软件团队所赋予的自主性
- 队伍组织和角色明确程度

1.5 软件工程实践

- 软件工程的实践如何融入框架？
- 实践的精髓：
  1. 理解问题（沟通和分析）
  2. 计划解决问题（建模和软件设计）
  3. 实施计划（代码生成）
  4. 检查结果的正确性（测试和质量保证）

方法：通过基本问题对应各个阶段

①理解问题（沟通和分析）

- 谁将从问题的解决中获益？
- 有哪些数据、功能和性能是解决问题所必须的？
- 问题可以划分吗？
- 问题可以图形化描述吗？

方法：通过基本问题对应各个阶段

② 策划解决方案（建模和软件设计）

- 以前曾经见过类似问题吗？
- 类似问题是否解决过？
- 可以定义子问题吗？
- 能用一种可以很快实现的方式来描述解决方案吗？

方法：通过基本问题对应各个阶段

③ 实施计划（代码生成）

- 解决方案和计划一致吗？
- 解决方案的每个组成部分是否可以证明正确？

方法：通过基本问题对应各个阶段

④ 检查结果的正确性（测试和质量保证）

- 能否测试解决方案的每个部分？
- 解决方案是否产生了与所要求的数据、功能和特性一致的结果？

1.5 软件工程实践

- 一般原则:
  1. 存在原则 --增加真正的价值
  2. 保持简洁 --易于维护和理解
  3. 保持愿景 --概念一致
  4. 关注使用者 --尽可能使其工作简化
  5. 面向未来 --通用，耐用
  6. 计划复用 --前瞻性的计划和设计
  7. 认真思考 --行动前清晰定位、完整思考

1.6 软件神话

- 管理神话：
  - 我们已经有了一本写满软件开发标准和规程的宝典。它无所不包，囊括了我们可能问到的所有问题
  - 如果我们未能按时完成计划，我们可以通过增加程序员人数而赶上进度
  - 如果将一个软件外包给另一家公司，则我们可以完全放手不管。
  - ……

1.6 软件神话（续）

- 用户神话
  - 有了对项目目标的大概了解，便足以开始编写程序，我们可以在之后的项目开发过程中逐步了解细节。
  - 虽然项目需求不断变更，但是因为软件是弹性的，因此可以很容易地适应变化
  - ……

1.6 软件神话（续）

- 从业者神话：
  - 当我们完成程序并将其交付使用之后，我们的任务就完成了。
  - 直到程序开始运行，才能评估其质量
  - 对于一个成功的软件项目，可执行程序是惟一可交付的成果。
  - 软件工程将导致我们产生大量无用文档，并因此降低工作效率。

作业

- 理解：
  - 什么是软件危机？
  - 软件危机的具体表现有哪些？
- 调研分析（交）：
  - 调研中国软件“卡脖子”问题主要有哪几个方面，深入一个领域说明现状及趋势（现状、难点、国家政策、发展趋势等）。