



软件工程与实践

授课教师：邱元杰 电子邮箱：yuanjiq@126.com， 微信电话：13679081552

1.1 软件的概念与特点

1.2 软件危机

1.3 软件工程概念与发展过程

1.4 软件工程知识体系

1.5 软件工程师的特质与职业道德

第一章：软件工程概述

介绍软件、软件危机、软件工程的**概念**，软件工程的**知识体系**和软件工程师的**职业道德**。

什么是软件?



电驴(VeryCD)	阿里旺旺	极点五笔	紫光
比特彗星	UUCall	QQ五笔	拼音
视频播放	音频播放	网络电视	其
暴风影音	酷狗音乐	PPS	WinRAR
RealPlayer	千千静听	风行	Nero
QQ影音	酷我音乐盒	UUSee	Adobe
pps影音	QQ音乐播放器	PPTV	Flash
KMPlayer	Foobar2000	暴风影音	Easy
驱动软件	图像处理	股票证券	木马
驱动精灵	光影魔术手	大智慧	金山
驱动人生	Photoshop	同花顺	瑞星



软件的定义

软件=程序+数据+文档

程序

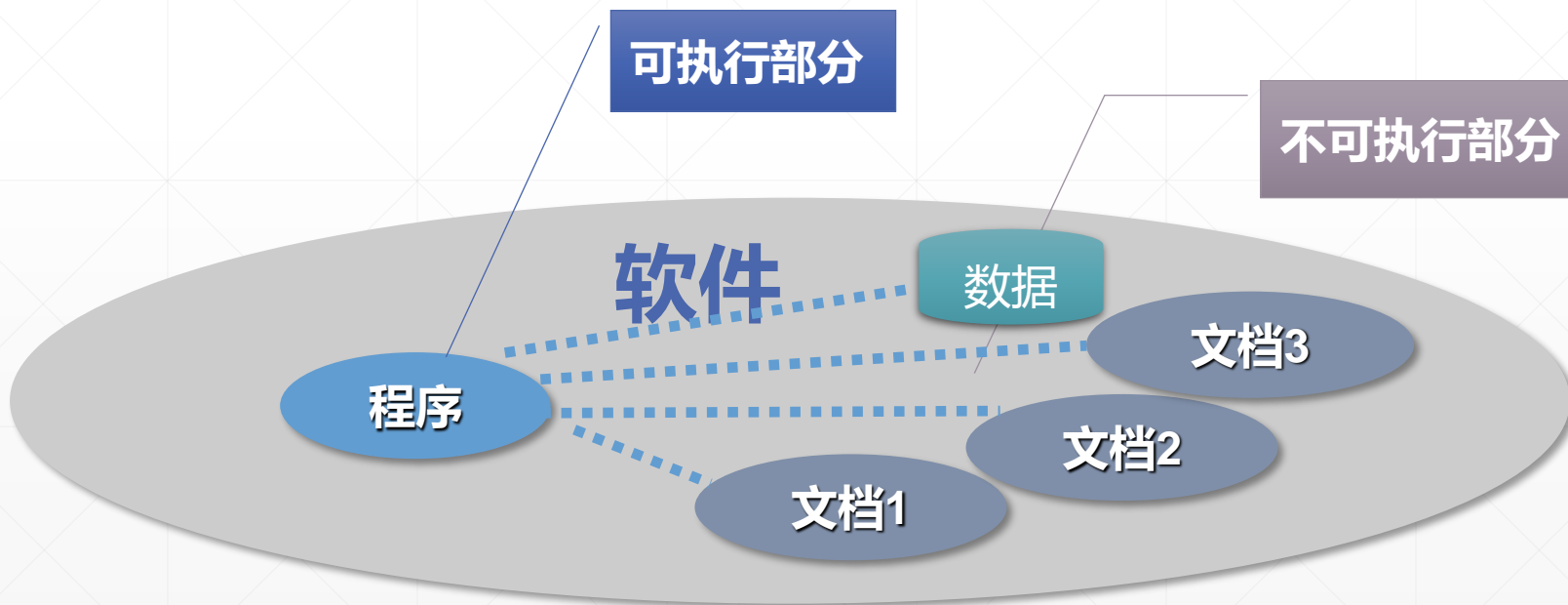
按事先设计的功能和性能需求执行的指令序列

数据

是程序能正常操纵信息的数据结构

文档

与程序开发、维护和使用有关的图文材料



软件开发的例子

- Microsoft Word 1.0开发，包含249,000行代码，投入660人月，前后历时5年，实际花费为预期时间的5倍。

提交报告日期	预期交付日期	预期到完成交付还需要的开发时间	实际到完成交付所需花费的开发时间	估算偏差率
84年9月	85年9月	365天	1887天	81%
85年6月	86年7月	395天	1614天	76%
86年1月	86年11月	304天	1400天	78%
86年6月	87年6月	334天	1235天	73%
87年1月	87年12月	334天	1035天	68%
.....
89年6月	89年9月	92天	153天	40%
89年7月	89年10月	92天	123天	25%
89年8月	89年11月	92天	92天	0%
89年11月	89年11月	0	0	0%

软件的特征

- **软件是开发的或者是工程化的，而不是制造的**
 - **软件生产是简单拷贝，而不是重复开发**
 - **软件产品易于多次修改，且总是要求修改**
 - **软件开发的环境对产品影响较大**
-

软件的特征（续）

- **软件开发的时间和工作量难以估计**
 - **软件开发进度难以客观衡量**
 - **软件的测试非常困难**
 - **软件不会磨损和老化，但会退化**
 - **软件维护不是简单更换元器件，变更容易产生新的问题**
-

软件双重作用

一方面是一种产品

- 提供计算能力
- 产生、管理、获取、修改、显示或传输信息



另一方面是开发其他软件产品的工具

- 支持或直接提供系统所需的功能
- 控制其他程序（如操作系统）
- 改善通信（如网络软件）
- 帮助开发其它软件（如软件开发工具）
- 其它功能.....



软件的分类（按软件功能）

系统软件

- 操作系统
- 数据库管理系统
- 设备驱动程序
- 通信处理程序等

支撑软件

- 文本编辑程序
- 文件格式化程序
- 磁盘或磁带间数据传输的程序
- 程序库系统
- 支持需求分析、设计、实现、测试和支持管理的软件

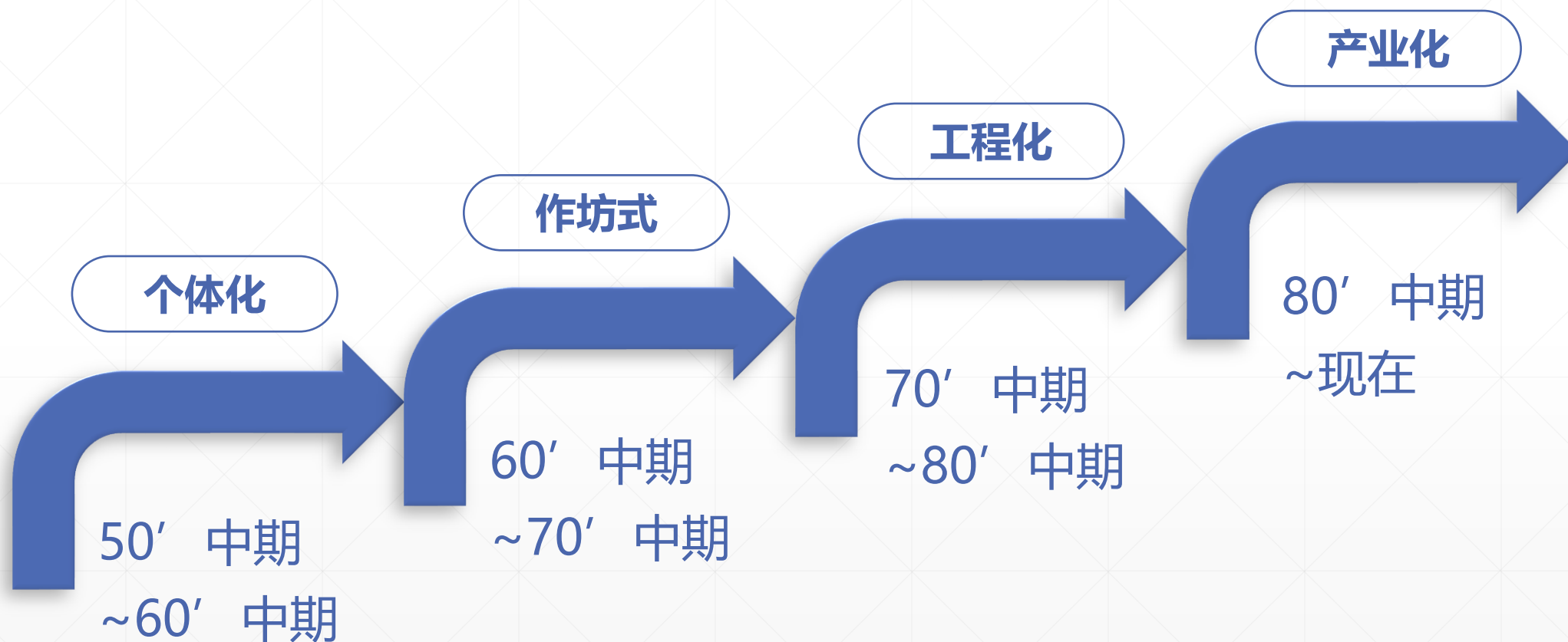
应用软件

- 商业数据处理软件
- 工程与科学计算软件
- 计算机辅助设计 / 制造软件
- 系统仿真软件
- 智能产品嵌入软件
- 医疗、制药软件
- 事务管理、办公自动化软件
- 计算机辅助教学软件

软件的分类 (按服务对象)



软件开发的发展



软件规模越来越大



应用软件系统也越来越复杂，规模迅速增长，动辄数百万行代码已是常见现象（谷歌：20亿行）

为什么软件发展如此之快？

计算需求

软件必须适应新的计算环境或技术

业务需求

软件必须改善，以实现新的业务需求



嵌入需求

软件必须扩展，以满足和新一代系统或数据库之间的互操作性

架构需求

软件必须重新设计，使其在新的网络环境是可用的

1.1 软件的概念与特点

1.2 软件危机

1.3 软件工程概念与发展过程

1.4 软件工程知识体系

1.5 软件工程师的特质与职业道德

第一章：软件工程概述

介绍软件、软件危机、软件工程的概
念，软件工程的
知识体系和软件工程师
的职业道德。

什么是软件危机

效率和质量下降

定义

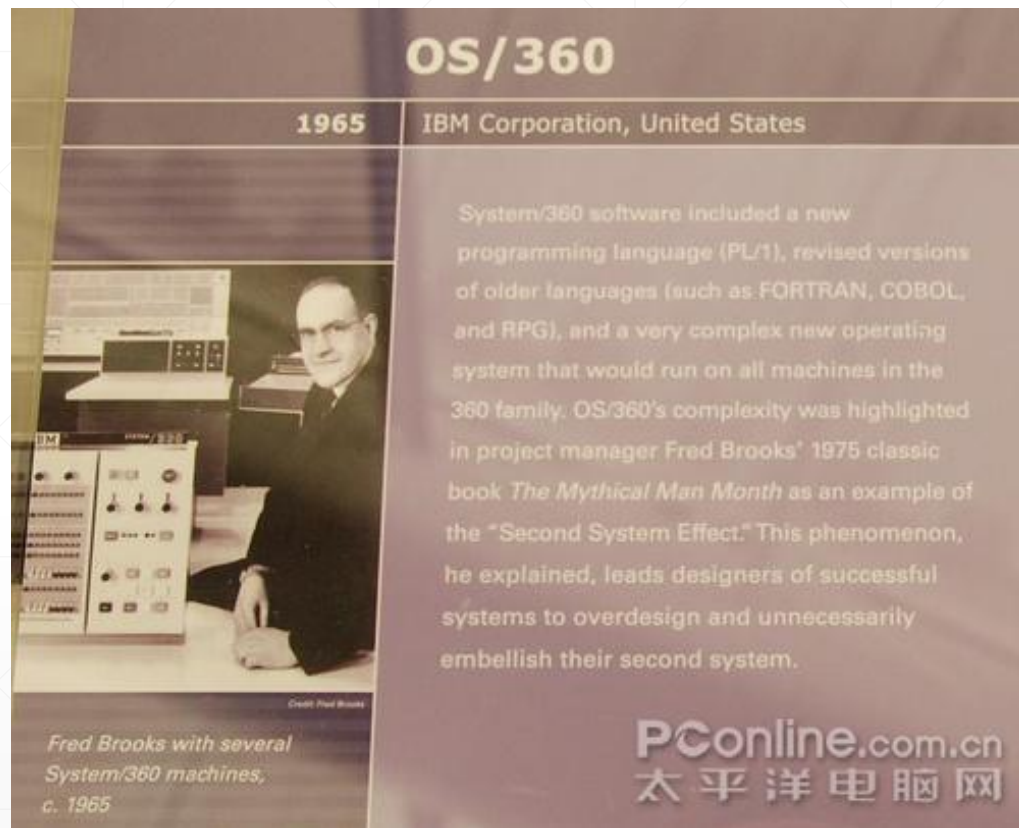
在计算机软件的开发和维护过程中所遇到的一系列严重问题。

1968年NATO会议
(Garmisch, Germany)
提出“软件危机”

- ❖ 项目超出预算
- ❖ 项目超过计划完成时间
- ❖ 软件运行效率很低
- ❖ 软件质量差
- ❖ 软件通常不符合要求
- ❖ 项目难以管理并且代码难以维护
- ❖ 软件不能交付

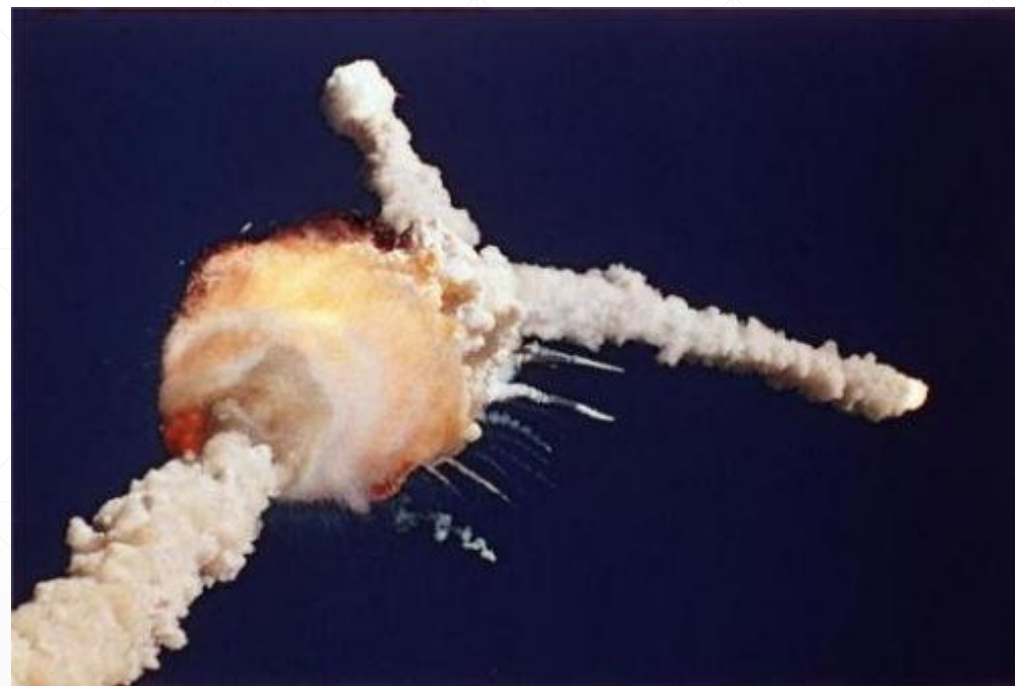
软件危机案例：OS/360

- 1961-1964年，IBM公司的 OS/360，共约100万条指令，花费了5000个人年，开发总投资5亿美元，达到当时美国研究原子弹的曼哈顿计划投资20亿美元的1/4，而结果却令人沮丧，错误多达2000个以上，系统根本无法正常运行。



软件危机案例：美国火箭爆炸

- 谣言：1963年在美国，由于一个FORTRAN程序的循环语句“DO 5 I=1,3”误写为“DO 5 I=1.3”，“,”被误写为“.”，**一点之差**导致飞往火星的火箭爆炸，造成了1000万美元的损失。
- 实际情况：根据NASA的报告，是1962年的飞往金星的宇宙飞船天线发生故障，致使地面无法控制火箭，于是火箭上的计算机开始控制火箭，然而在**导航系统软件中又存在一个bug**（打印错误，在平滑半径导数表达式“ $R\text{-dot-bar sub } n$ ”中漏掉了代表平滑的“bar”），致使火箭迅速偏离航道。于是发射场的安全官员下令摧毁飞船。



软件危机案例：美国银行信托软件系统

- 美国银行1982年进入信托商业领域，并规划发展信托软件系统。
- 项目原订预算2千万美元，开发时程9个月，预计于1984年12月31日以前完成，后来至1987年3月都未能完成该系统，期间已投入6千万美元。
- 美国银行最终因为此系统不稳定而不得不放弃，并将340亿美元的信托账户转移出去，并失去了6亿美元的信托生意商机。



现在的软件危机

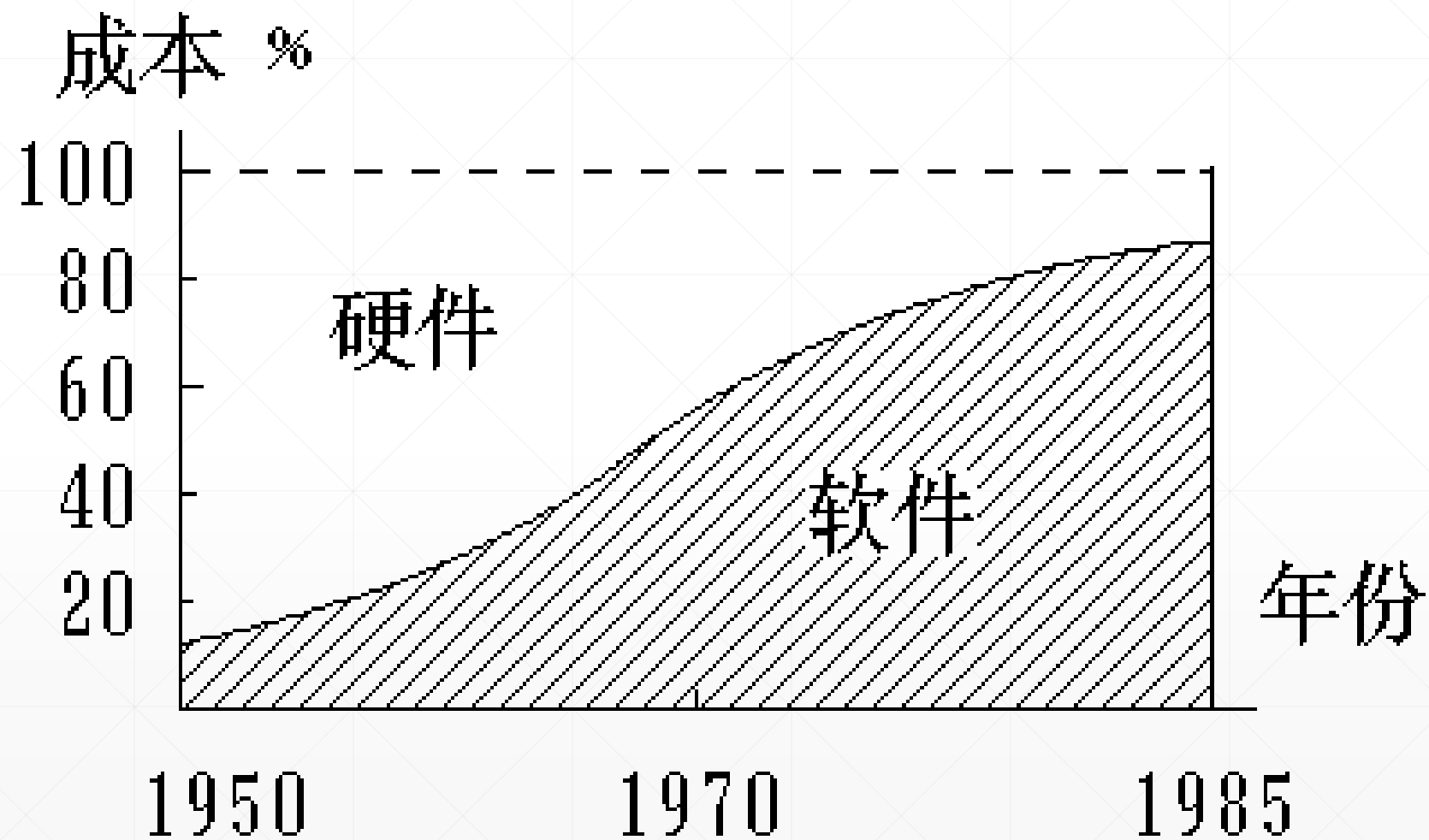
➤ Standish Group Report: CHAOS Report 2015

MODERN RESOLUTION FOR ALL PROJECTS

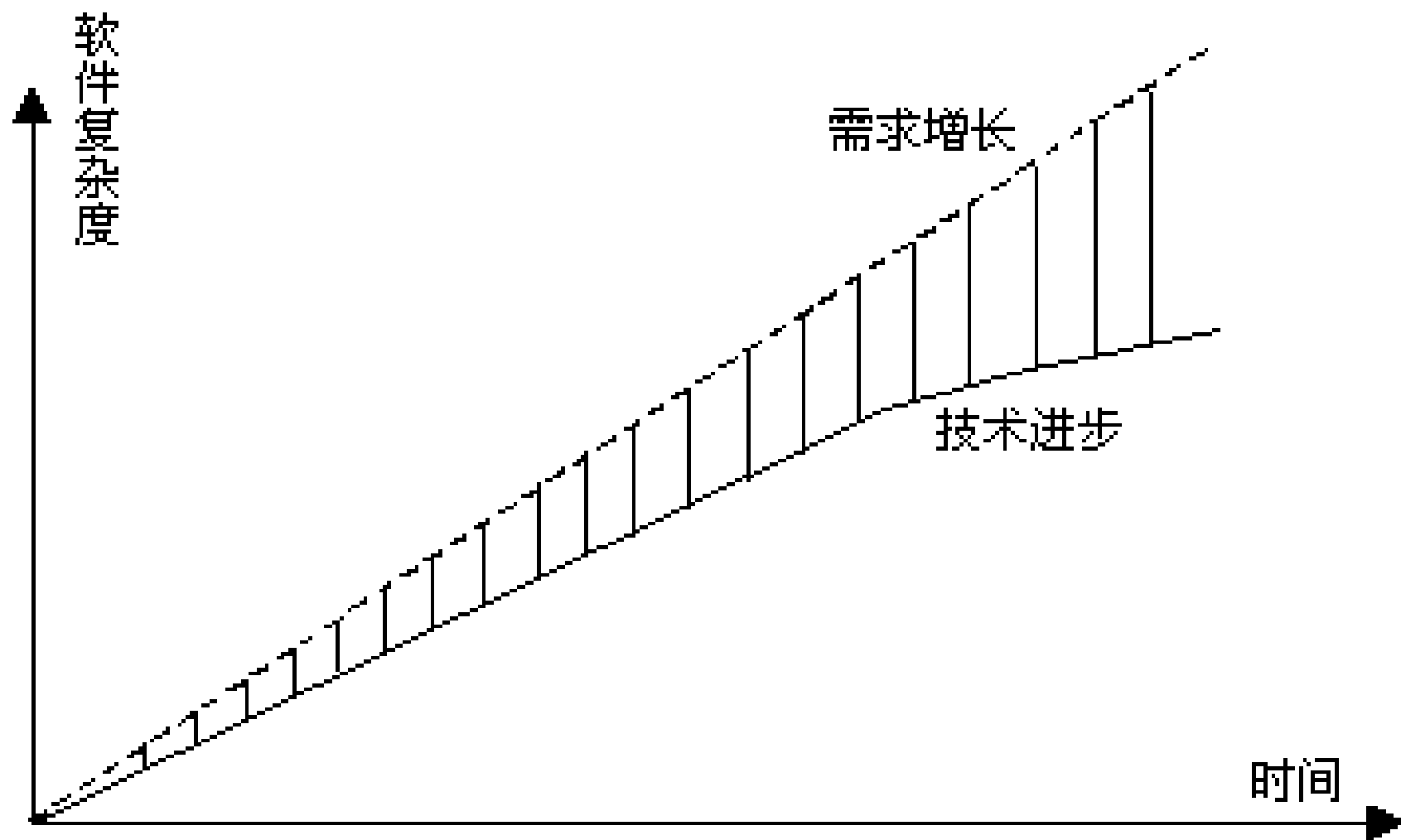
	2011	2012	2013	2014	2015
SUCCESSFUL	29%	27%	31%	28%	29%
CHALLENGED	49%	56%	50%	55%	52%
FAILED	22%	17%	19%	17%	19%

The Modern Resolution (OnTime, OnBudget, with a satisfactory result) of all software projects from FY2011-2015 within the new CHAOS database. Please note that for the rest of this report CHAOS Resolution will refer to the Modern Resolution definition not the Traditional Resolution definition.

软件成本日益增加



软件技术进步 < 需求增长



产生软件危机的原因

客观：软件本身特点


- 逻辑部件
- 规模庞大

主观：不正确的开发方法


- 忽视需求分析
- 错误认为：软件开发=程序编写
- 轻视软件维护

消除软件危机的途径：软件工程！

对计算机软件有一个正确的认识：
软件≠程序



必须充分认识到软件开发不是某种个体劳动的神秘技巧，而应该是一种组织良好、管理严密、各类人员协同配合、共同完成的工程项目。



推广使用在实践中总结出来的开发软件的成功技术和方法。
开发和使用更好的软件工具。

1.1 软件的概念与特点

1.2 软件危机

1.3 软件工程概念与发展过程

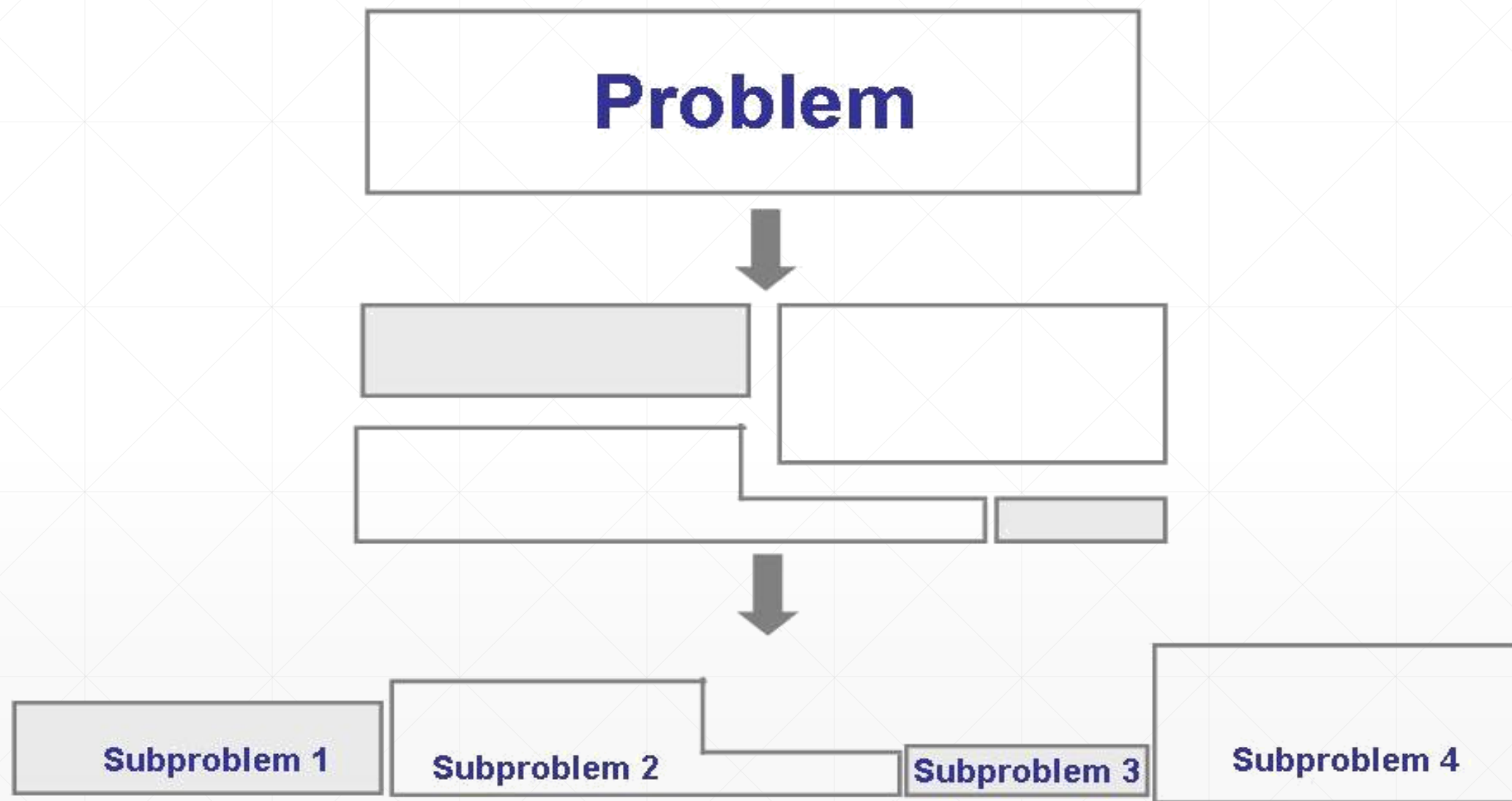
1.4 软件工程知识体系

1.5 软件工程师的特质与职业道德

第一章：软件工程概述

介绍软件、软件危机、软件工程的**概念**，软件工程的**知识体系**和软件工程师的**职业道德**。

软件工程有什么用？



软件工程的定义

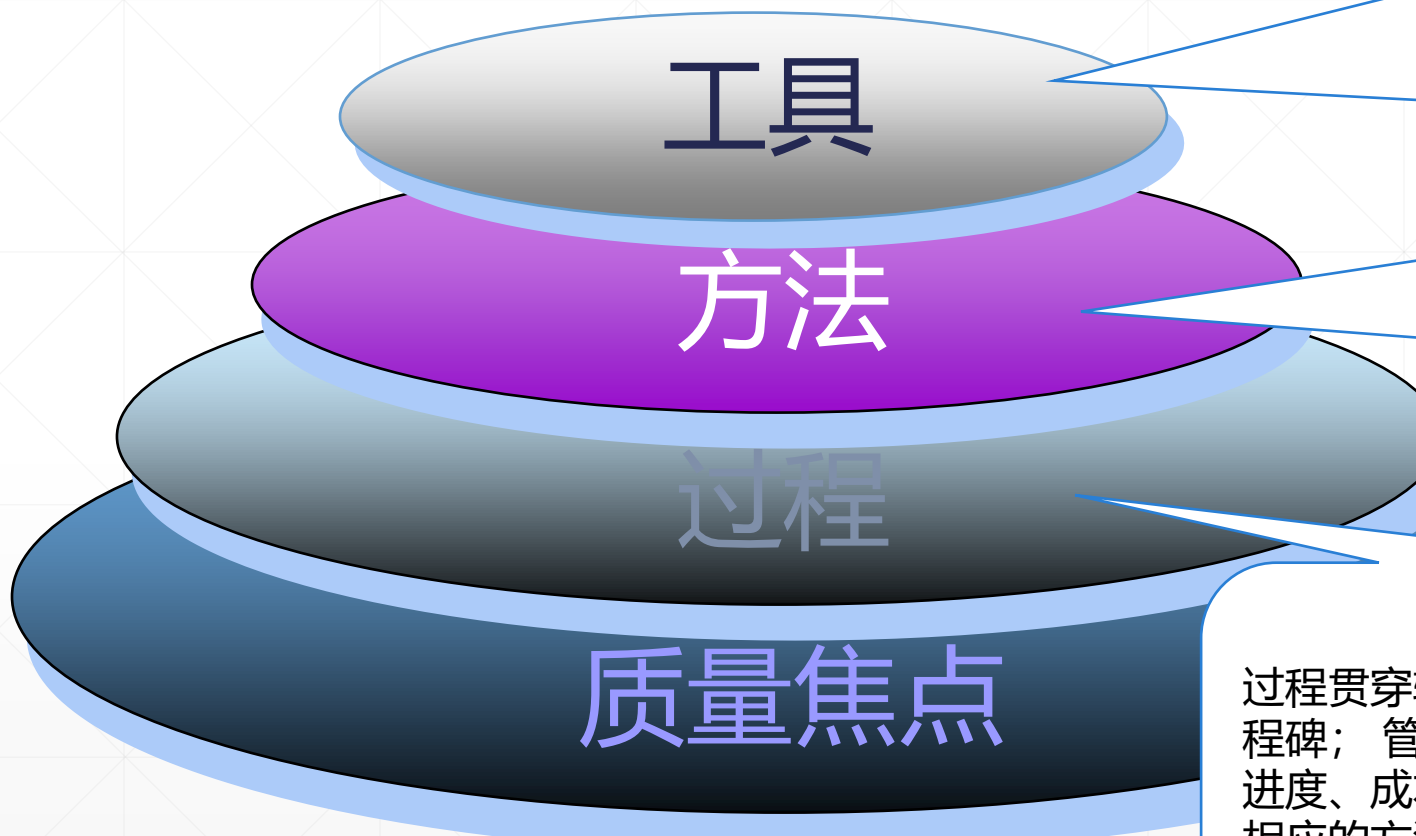
定义

IEEE计算机协会将软件工程定义为：（1）应用系统化的、学科化的、定量的方法，来开发、运行和维护软件，即，将工程应用到软件。（2）对（1）中各种方法的研究。

目标

软件工程的目标是在给定的时间和预算内，按照用户的需求，开发易修改、高效、可靠、可维护、适应力强、可移动、可重用的软件。

软件工程三要素：方法、工具、过程



软件工程层次图

它为软件工程师的过程和方法提供自动化或半自动化的工具支持。将若干工具集成起来，与软件工程数据库和计算机系统构成一个支持软件开发的系统称“计算机辅助软件工程(CASE)”，系统中某一工具的信息加工结果可以作为另一工具的输入。集成的软件工程工具再加上人的因素构成了软件工程环境。

软件工程方法是完成软件工程项目的手段。它支持项目计划和估算、系统和软件需求分析、设计、编程、测试和维护。软件工程方法依赖一组原则，它贯穿软件工程的各个环节。软件工程方法分两类：结构化方法和面向对象方法。

过程贯穿软件开发的各个环节，在各环节之间建立里程碑；管理者在软件工程过程中对软件开发的质量、进度、成本进行评估、管理和控制；技术人员采用相应的方法和工具生成软件工程产品（模型、文档、数据、报告、表格等）。

软件工具

- 它为软件工程的过程和方法提供自动化或半自动化的工具支持。
 - 将若干工具集成起来，与软件工程数据库和计算机系统构成一个支持软件开发的系统称“计算机辅助软件工程(CASE)”，系统中某一工具的信息加工结果可以作为另一工具的输入。
 - 集成的软件工程工具再加上人的因素构成了软件工程环境。
-

软件工程方法

- **软件工程方法是完成软件工程项目的手段。**
 - **它支持项目计划和估算、系统和软件需求分析、设计、编程、测试和维护。**
 - **软件工程方法依赖一组原则，它贯穿软件工程的各个环节。**
 - **软件工程方法分两类：结构化方法和面向对象方法。**
-

软件过程

- **过程贯穿软件开发的各个环节，在各环节之间建立里程碑；**
 - **管理者在软件工程过程中对软件开发的质量、进度、成本进行评估、管理和控制；**
 - **技术人员采用相应的方法和工具生成软件工程产品（模型、文档、数据、报告、表格等）。**
 - **软件过程不是标准，是可裁剪，可改进的，是要适应具体项目的**
-

软件工程的发展已经历了四个重要阶段：

1.第一代软件
工程 —
传统的软件
工程



2.第二代软件
工程 —
对象工程



3.第三代软件
工程 —
过程工程



4.第四代软件
工程 —
构件工程


第一代软件工程 — 传统的软件工程

60年代末到70年代为了克服“软件危机”
提出“软件工程”的名词, 将软件开发纳入
工程化的轨道, 基本形成软件工程的概念、
框架、技术和方法。称为传统的软件工程。



第二代软件工程 — 对象工程

60年代末到70年代为了克服“**软件危机**”提出“软件工程”的名词, 将软件开发纳入工程化的轨道, 基本形成软件工程的概念、框架、技术和方法。称为传统的软件工程。

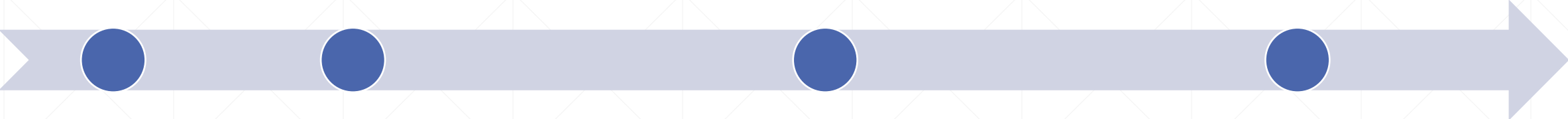


80年代中到90年代, 面向对象的方法与技术得到发展, 研究的重点转移到面向对象的分析与设计, 演化成为一种完整的软件开发方法和系统的技术体系, 称为对象工程。

第三代软件工程 — 过程工程

60年代末到70年代为
了克服“**软件危机**”
提出“**软件工程**”的
名词, 将软件开发纳入
工程化的轨道, 基本
形成软件工程的**概念、
框架、技术和方法**。
称为传统的**软件工程**。

80年代中开始, 人们在软件开发的实践过程中
认识到: 提高软件生产率, 保证软件质量的关键是“**软件过程**”, 是软件开发和维护中的**管理和支持能力**, 逐步形成**软件过程工程**。



80年代中到90年代,
面向对象的方法与技
术得到发展, 研究的
重点转移到面向对象
的分析与设计, 演化
为一种完整的软件开
发方法和系统的技术
体系, 称为**对象工程**。

第四代软件工程 — 构件工程

60年代末到70年代为了克服“**软件危机**”提出“**软件工程**”的名词, 将软件开发纳入工程化的轨道, 基本形成软件工程的**概念、框架、技术和方法**。称为**传统的软件工程**。

80年代中开始, 人们在软件开发的实践过程中认识到: 提高软件生产率, 保证软件质量的关键是“**软件过程**”, 是软件开发和维护中的**管理和支持能力**, 逐步形成**软件过程工程**。

80年代中到90年代, 面向对象的方法与技术得到发展, 研究的重点转移到面向对象的分析与设计, 演化成为一种完整的软件开发方法和系统的技术体系, 称为**对象工程**。

90年代起, 基于**构件 (Component)**的开发方法取得重要进展, 软件系统的开发可通过使用现成的可复用构件组装完成, 而无需从头开始构造, 以此达到**提高效率和质量, 降低成本的目的**。称为**构件工程**。

软件工程的7个原则 (B. W. Boehm, 1983)

使用阶段性生命周期计划的管理

进行连续的验证

保证严格的产品控制

使用现代编程工具工程实践

保持清晰的责任分配

用更好更少的人

保持过程改进

1.1 软件的概念与特点

1.2 软件危机

1.3 软件工程概念与发展过程

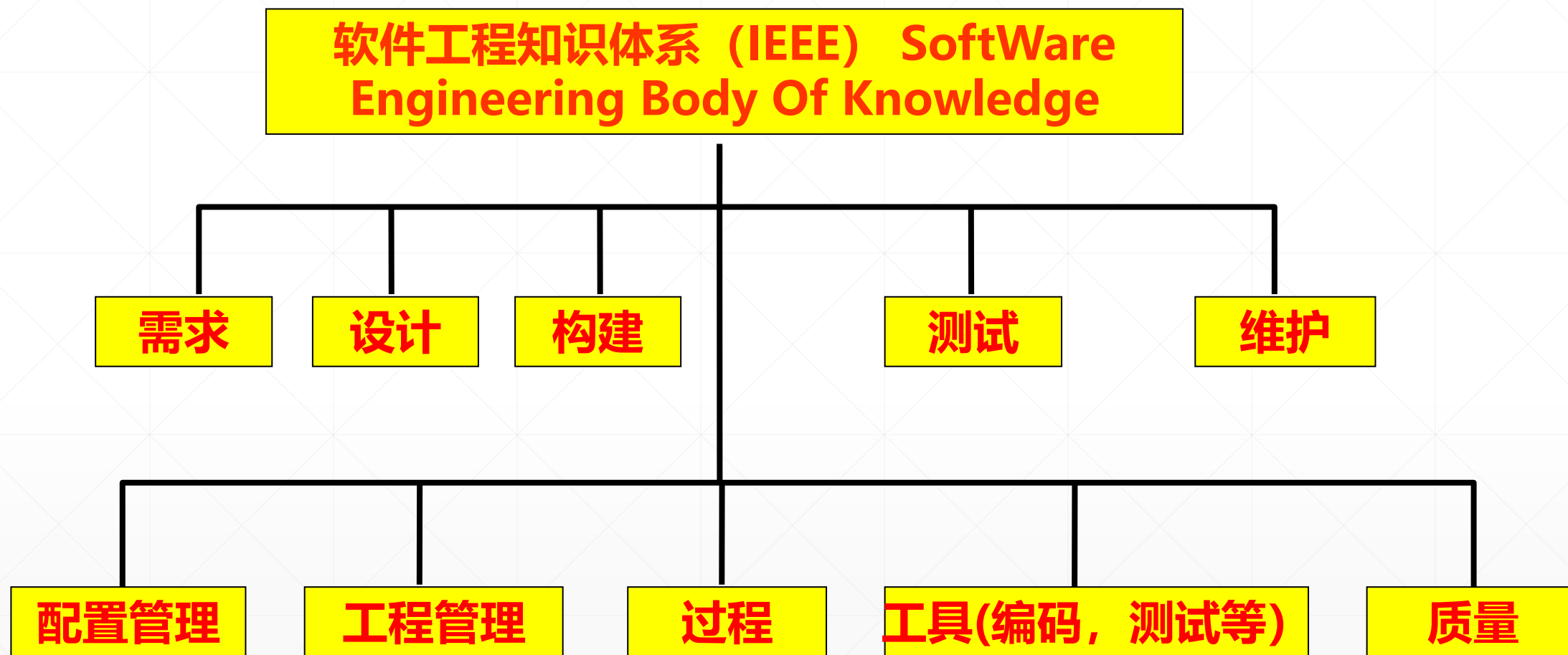
1.4 软件工程知识体系

1.5 软件工程师的特质与职业道德

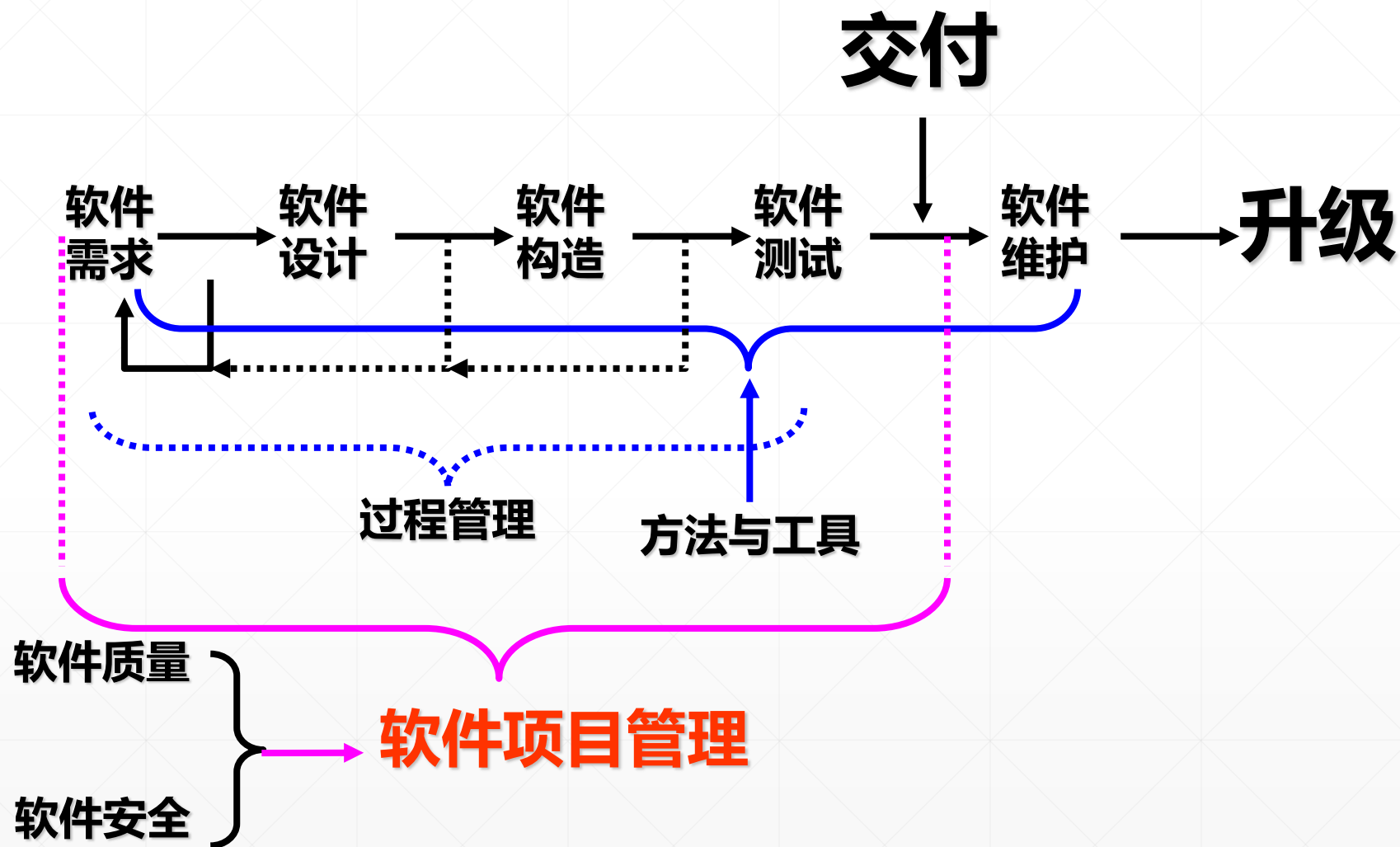
第一章：软件工程概述

介绍软件、软件危机、软件工程的概
念，软件工程的
知识体系和软件工程师
的职业道德。

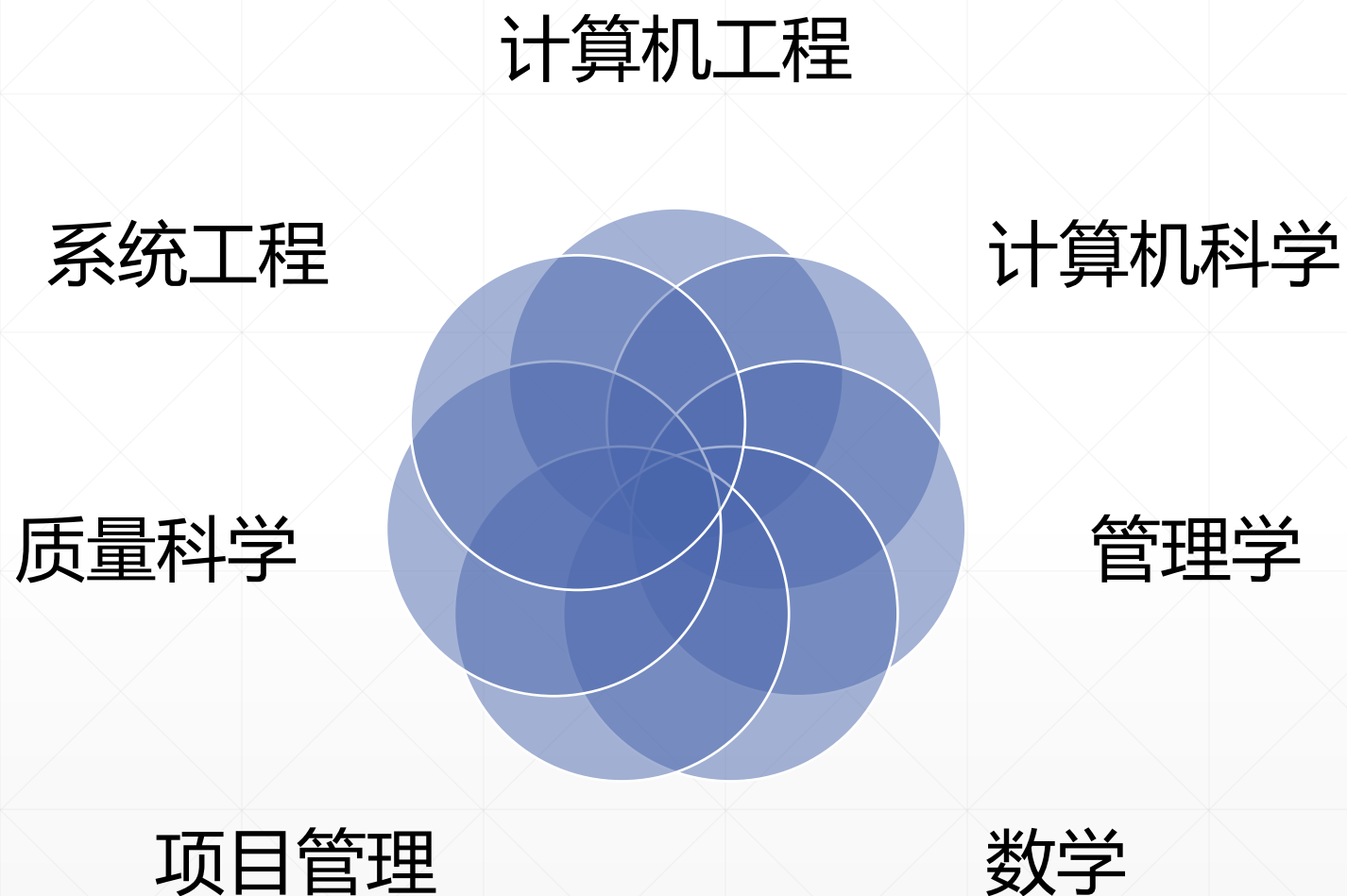
软件工程知识体系 (SWEBOK)



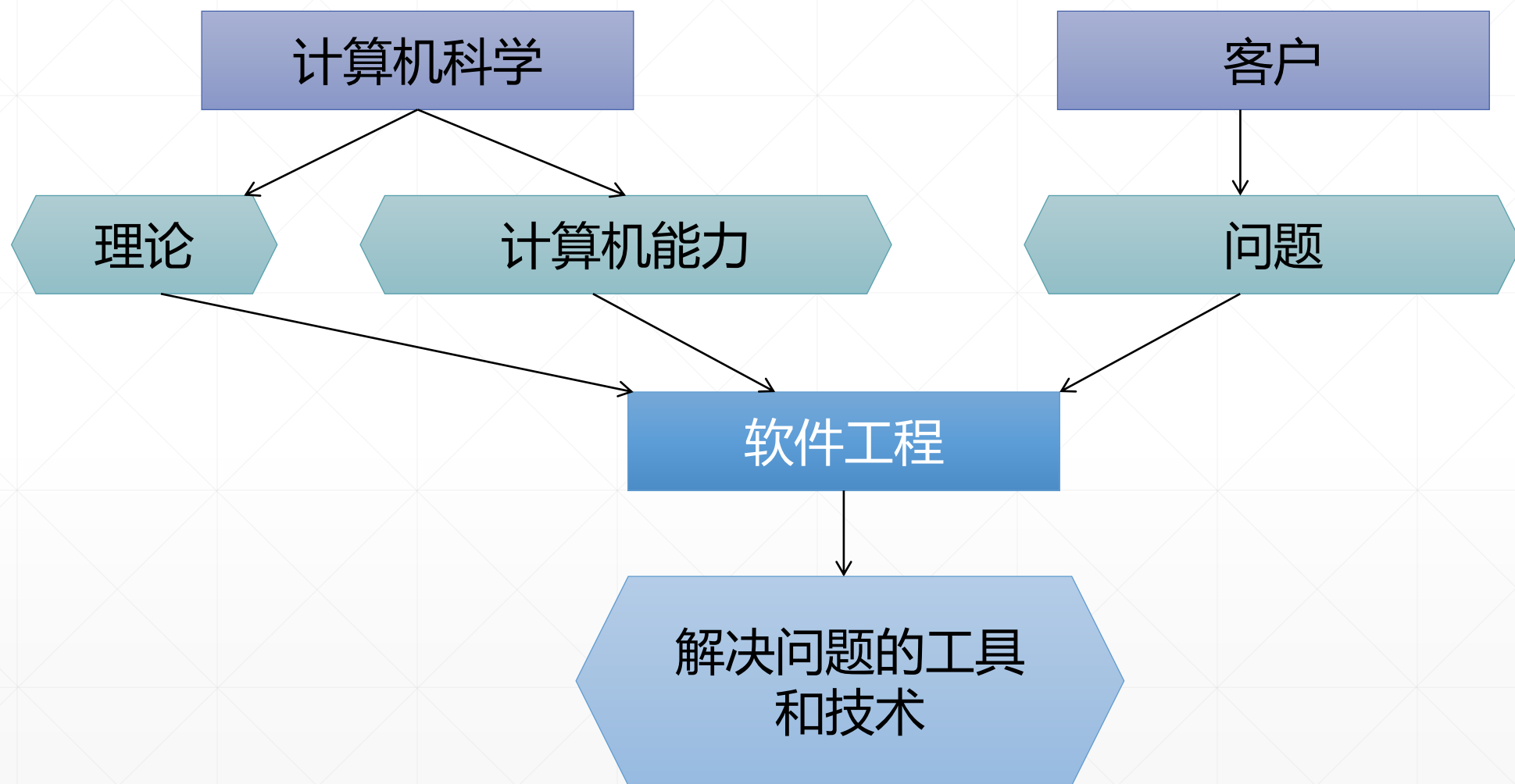
软件工程知识体系各主题之间的关联



软件工程是一门交叉学科



软件工程 VS 计算机科学



软件工程 VS 计算机科学

	软件工程	计算机科学
目标	在时间、资源、人员这3个主要限制条件下构建满足用户需求的软件系统	探索正确的计算和建模方法，从而改进计算方法本身
产品	软件（比如办公包和编译器）	算法（比如希尔排序法）和抽象的问题（比如哲学家进餐问题）
进度与时间表	软件项目都有特定的进度与时间表	研究项目一般不具有设置的进度与时间表
关注点	软件工程关注如何为用户实现价值	软件理论关注的是软件本身运行的原理，比如时间复杂度，空间复杂度，和算法的正确性
变化程度	随着技术和用户需求的不断变化，软件开发人员必须时刻调整自己的开发以适应当前的需求。同时软件工程本身也处于不断的发展中	对于某一种特定问题的正确解决方法将永远不会改变
需要的其他知识	相关领域的知识	数学

一些对软件工程的误解 (1) 管理方的误解:

- M1: 我们已经有一本关于软件生产的标准和过程的书, 这还不能让我们学习到需要的知识吗?
- R1: 相比最新的大型主机, 工作站和PC, 这会使我们在做高质量软件开发时花费更多时间。
- M2: 如果我们项目进度落后了, 可以加入更多的程序员来赶进度。
- R2: 软件开发的机制和手工作业不一样。在一个延迟了的软件项目中加入新的开发人员只会让它延迟更多。
- M3: 如果我们将软件项目外包给第三方, 我们就轻松了, 让那个公司去完成它吧。
- R3: 如果组织管理方不懂得如何从内部管理和控制软件项目, 即使将项目外包也无济于事。

(注: M 代表误解, R 代表现实。)

一些对软件工程的误解 (2)

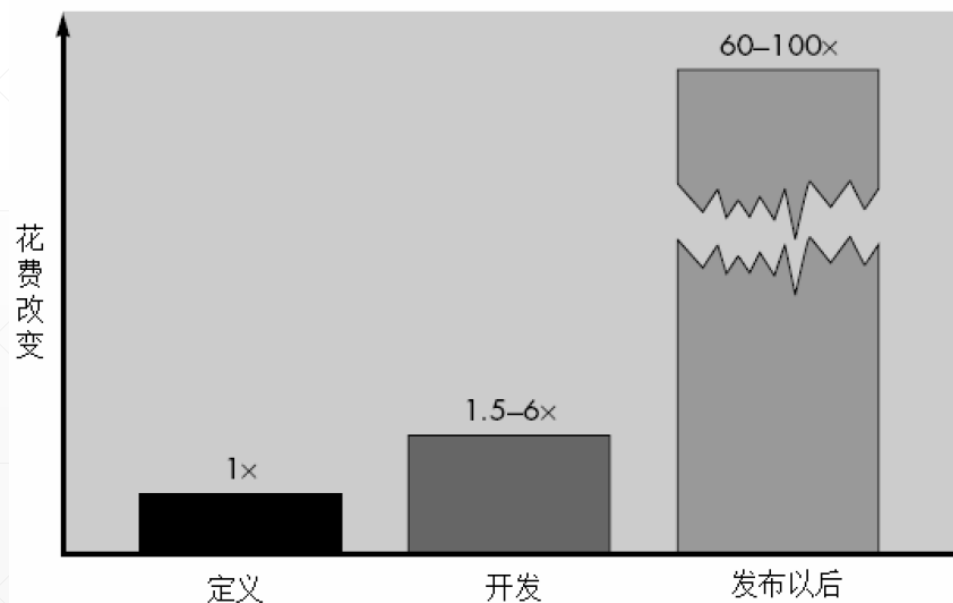
客户方的误解

M1: 对目标的一般陈述就足以开始编程，我们可以今后再补充细节。

R1: 前期糟糕的项目需求定义，是导致软件失败的主要原因。

R2: 项目需求的确在不断变化，但变化所产生的影响是根据变化提出的时间不同而不同的。

需求变化对变更成本的影响



一些对软件工程的误解 (3)

开发人员: 软件从业者积累了50 年的编程文化。在早期的软件开发中, 编程被视为一种艺术形式。

M1: 一旦我们编程完毕并成功运行, 我们的工作就结束了。

R1: “越早开始写代码, 我们就会花费越长的时间去完成它”。工业数据显示, 软件开发60%-80%的精力将耗费在软件首次提交给用户以后。

一些对软件工程的误解 (3)

- **M2: 当我的程序运行之前，我没有办法评估它的质量。**
 - **R2: 一个最有效的软件质量保证机制应当在项目的正式开始启动时——可以通过技术报告体现。**
 - **M3: 唯一可交付的工作成果是一个成功运行的项目程序。**
 - **R3: 一个可运行的程序只是软件结构的一部分，它还包含了许多其它因素。**
 - **M4: 软件工程将会让我们去创建大量不必要的文档，并且总是使我们的进度放慢。软件工程仅仅是文档而已。**
 - **R4: 软件工程并不是创建文档，而是创建质量。更好的质量减少返工的概率。更少返工会让项目更早交付。所有的文档都是提高团队沟通和质量所必须的。**
-

1.1 软件的概念与特点

1.2 软件危机

1.3 软件工程概念与发展过程

1.4 软件工程知识体系

1.5 软件工程师的特质与职业道德

第一章：软件工程概述

介绍软件、软件危机、软件工程的概
念，软件工程的
知识体系和软件工程师
的职业道德。

软件工程师的特质

- 个人责任感
 - 敏锐的眼光
 - 坦诚的态度
 - 高抗压能力
 - 高度的公平感
 - 注重细节
 - 务实的态度
-

软件工程职业道德和责任规范



1) 诚信：工程师们应当对他们的雇主和顾客时刻保持诚信而无论之前是否达成了关于诚信的协议。



2) 能力：工程师们不应该虚夸他们的能力水平。他们不应该故意接受一份超出自己能力范围的工作。



3) 知识产权：工程师们应该了解当地的知识产权法律法规，如专利权、版权等。他们应该小心确保雇主和客户的知识产权受到了保护。



4) 滥用计算机：软件工程师不以他们的工作职责为由滥用别人的电脑。滥用计算机的范围很广，从极小（在雇主的机器上玩游戏）到极其严重的（传播病毒）。

