软件工程

内容

1. 软件工程产生背景

✓软件危机的表现及根源

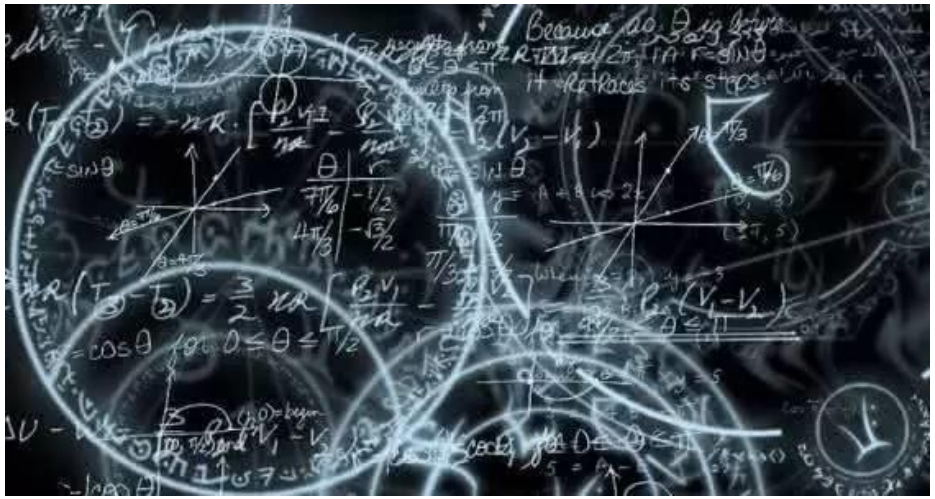
2. 软件工程基本内涵

✓思想、要素、目标和原则

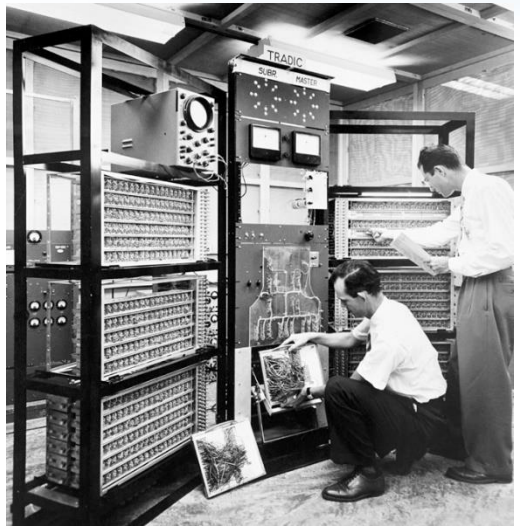
1.1 1950s-1960s的计算机软件应用

随着计算机硬件的发展，软件应用也迅速发展

- **应用领域变化**
  - ✓最早满足军方应用，如科学计算
  - ✓逐步走向商业应用等新领域，如银行、航空等领域的事务处理
- **应用数量增长**
  - ✓计算机软件的需求量不断上升
- **应用复杂性增加**
  - ✓多样化的用户
  - ✓多样化的需求



1960s的个体作坊式软件开发



作坊式的个人创作

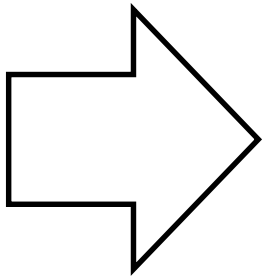
第二代晶体管计算机：
TRADIC（1954）
IBM 1401（1958）

- 依靠个人的能力
- 相互之间缺乏合作
- 关注计算存储时空利用
- 程序规模小且功能单一
- 无系统性方法和标准流程

1983年，严援朝一人历时3月，开发了CCDos，2万行代码

1.2 个体作坊式开发面临挑战

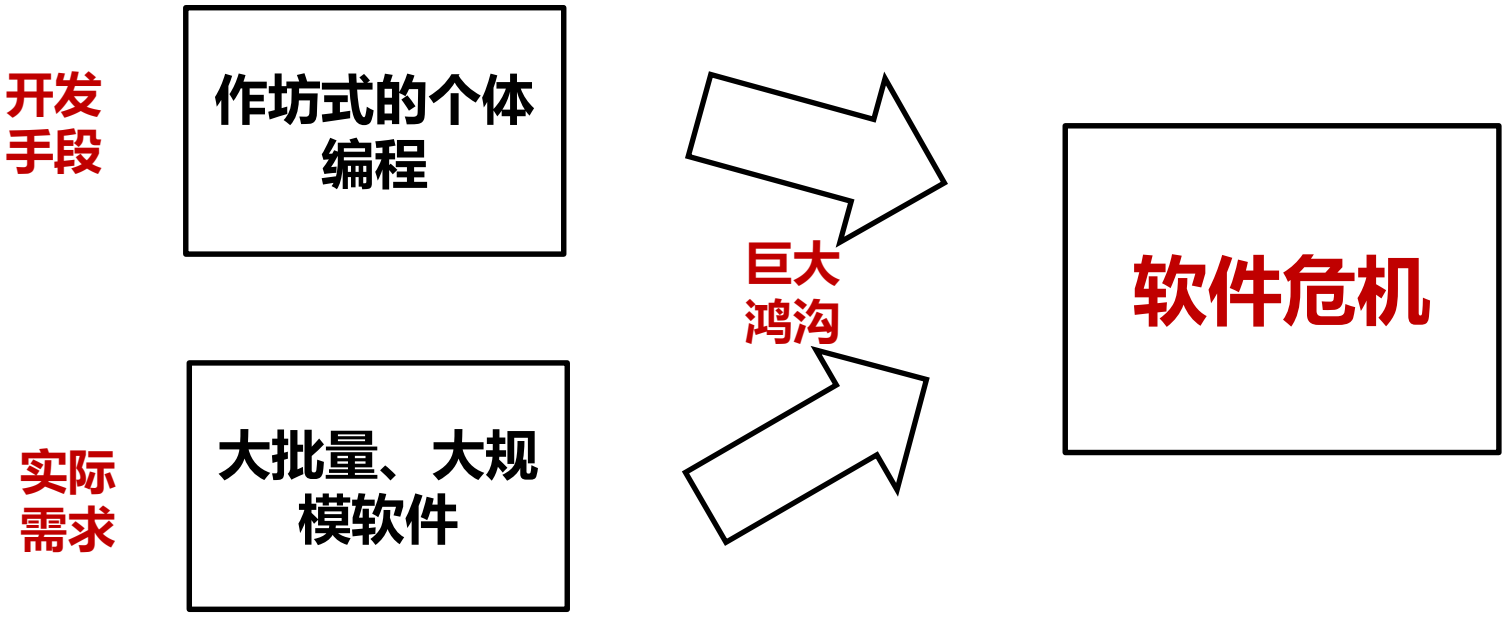
作坊式的个体编程开发



大批量和大规模软件系统

随着软件的复杂性和规模日益增大，传统的个体开发方法开始显得力不从心。

1.3 软件危机的出现

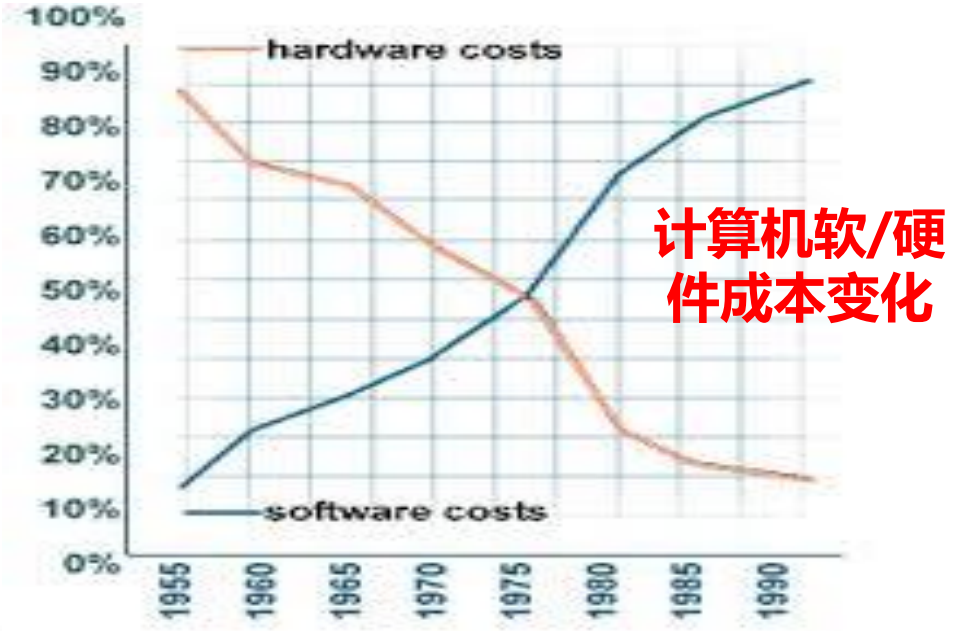


❑软件危机是指**落后的软件生产方式**无法满足迅速增长的**软件需求**，从而导致软件开发与维护过程中出现一系列的严重问题的现象。

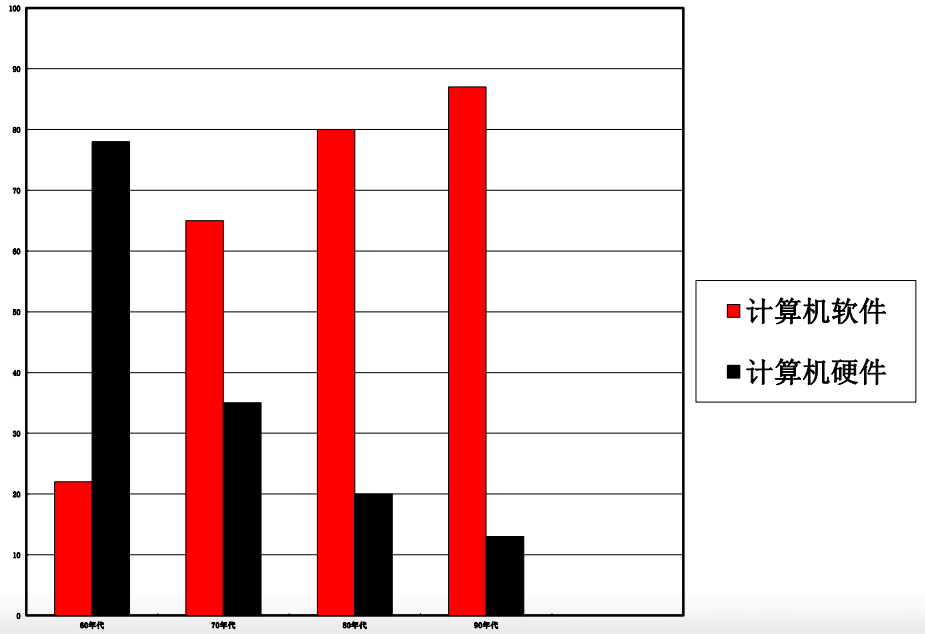
1.3.1 开发成本高

❑软件成本高，软硬件投资比发生急剧变化

- ✓美国空军：1955年软件占总费用(计算机系统)的18%，70年60%，85年达到85%
- ✓IBM 360 OS：5000+人年，耗时4年(1963-1966)，花费2亿多美元



计算机软/硬件成本变化



1.3.2 进度难以控制

- 项目延期比比皆是
- 由于进度问题而取消的软件项目较常见
- 只有一小部分的项目能够按期完成

- ✓ 丹佛国际机场行李处理系统，1989年动工，预计1993年10月启用。
- ✓ 最终在1995年1月启用，落后进度13个月，超出预算近20亿美元，总造价52亿美元。

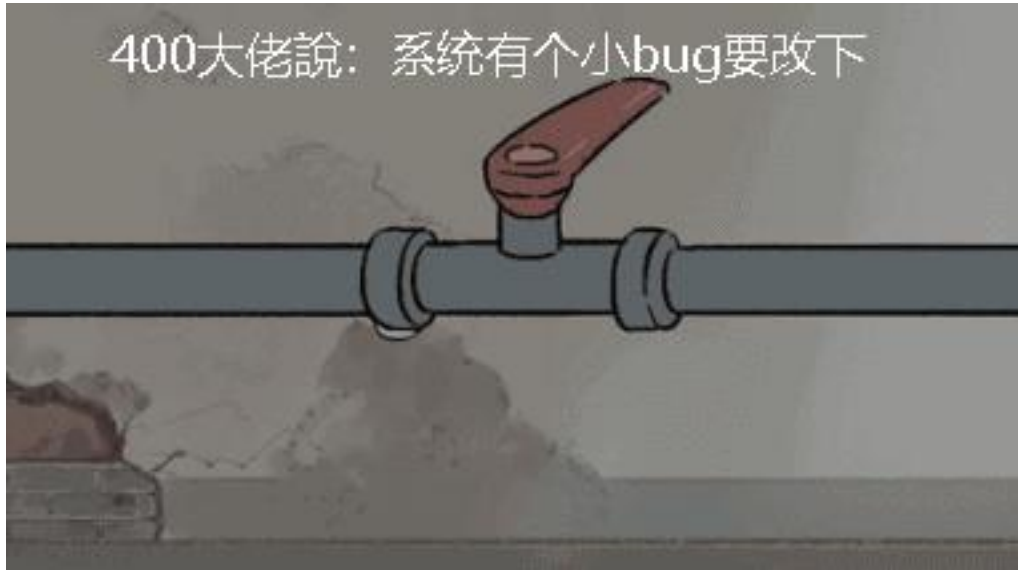
1.3.3 质量难以保证

- 人总是会犯错误的
- 软件开发的错误表现为多种形式
  - ✓没有按照要求（需求）来开发
  - ✓编写的代码在功能上存在错误
  - ✓实现了功能但是性能达不到要求
  - ✓……
- 有些软件错误可能是致命的

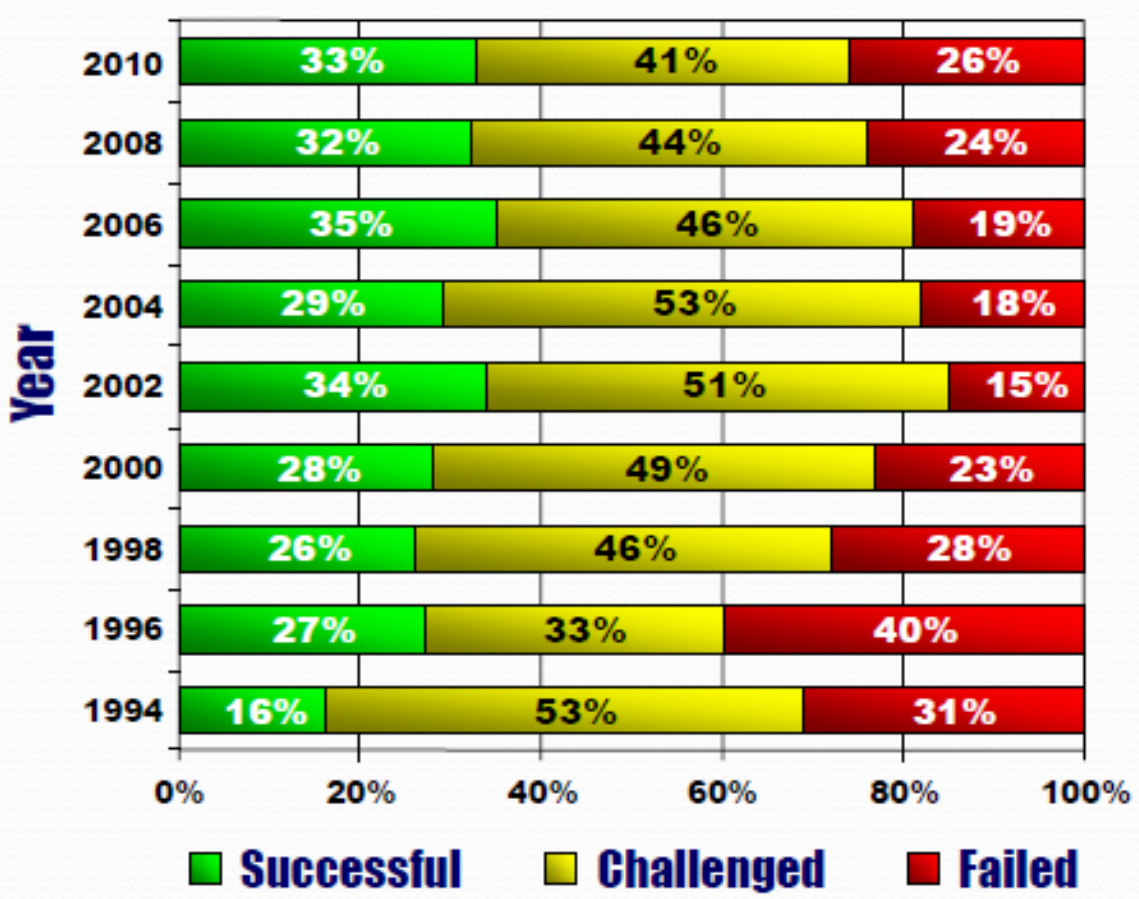
2018年和2019年，两架波音737MAX飞机相继发生坠机事故，导致数百人丧生。调查发现，飞机的自动防失速系统（MCAS）存在软件缺陷。

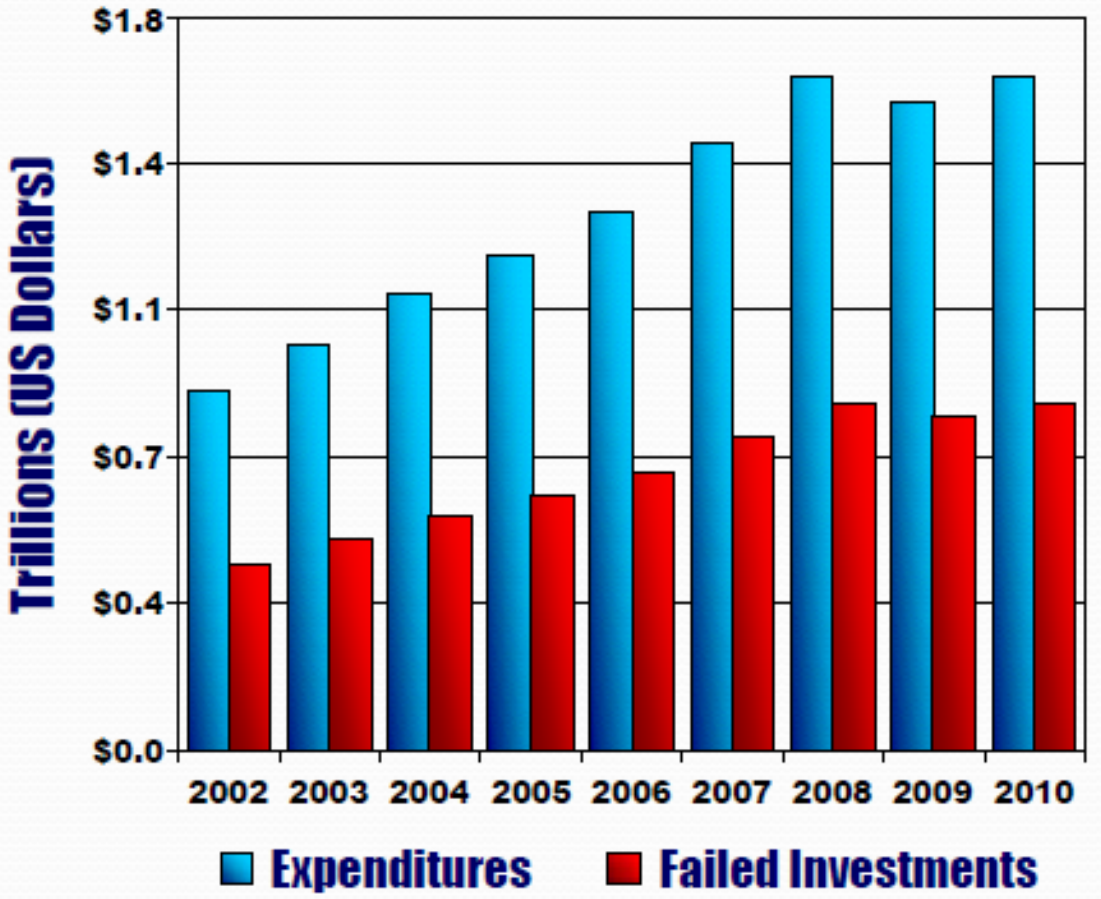
1.3.4 软件维护困难

- **理解**
  - ✓读懂程序比较困难，尤其是他人程序
- **修改**
  - ✓程序非常脆弱，牵一发而动全身
- **出错**
  - ✓改了以后易引入错误
- **发现**
  - ✓有了错误后难以发现



1.3.5 失败风险很大





计算机软件开发的成功比例和失败投资

David F. Rico, PMP, ACP, CSM, Lean & Agile Software Methods, 2012.

软件危机的产生根源

- 对软件这样一类**复杂和特殊系统**的认识不清
- 没有找到支持软件系统开发的**有效方法**
  - ✓基础理论、关键技术、开发过程、支撑工具等
- 缺乏成功软件开发**实践**以及相应的开发**经验**
  - ✓系统总结、认真分析、充分借鉴、吸取教训

1.4 如何来解决软件危机?

❑如何来解决软件危机?

- ✓策略、方法、理论、技术等

❑多方共同关注的问题

- ✓用户(如美国军方)
- ✓工业界(如IBM)
- ✓学术界（如研究学者）

软件开发迫切需要理论和方法指导，软件工程应运而生！

内容

1. 软件工程产生背景
   - ✓软件危机的表现及根源

2. 软件工程基本内涵
   - ✓思想、要素、目标和原则

2.1 软件工程的诞生

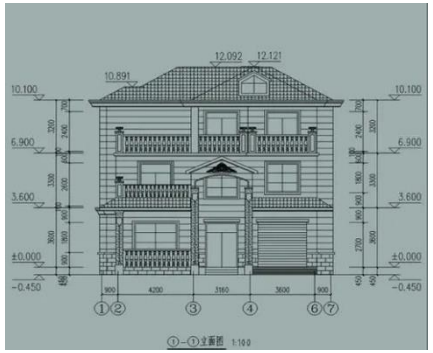
- 时间：1968年
- 地点：西德南部小城
- 事件：NATO(北大西洋公约组织)召开的会议
- 主题：如何解决软件危机
- 成果：提出了软件工程

提出了软件工程的概念及思想，标志着软件工程的诞生

2.2 何为软件工程

- 将**系统的**、**规范的**、**可量化**的方法应用于软件的开发、运行和维护的过程；　以及上述方法的研究 -- [*IEEE 93*]
  - ✓**系统化**：提供全生命周期、完整的方法指导
  - ✓**规范化**：为各项开发活动提供标准化的指南
  - ✓**可量化**：对成本、进度、质量等要素进行量化

获取需求



设计图纸

施工建造

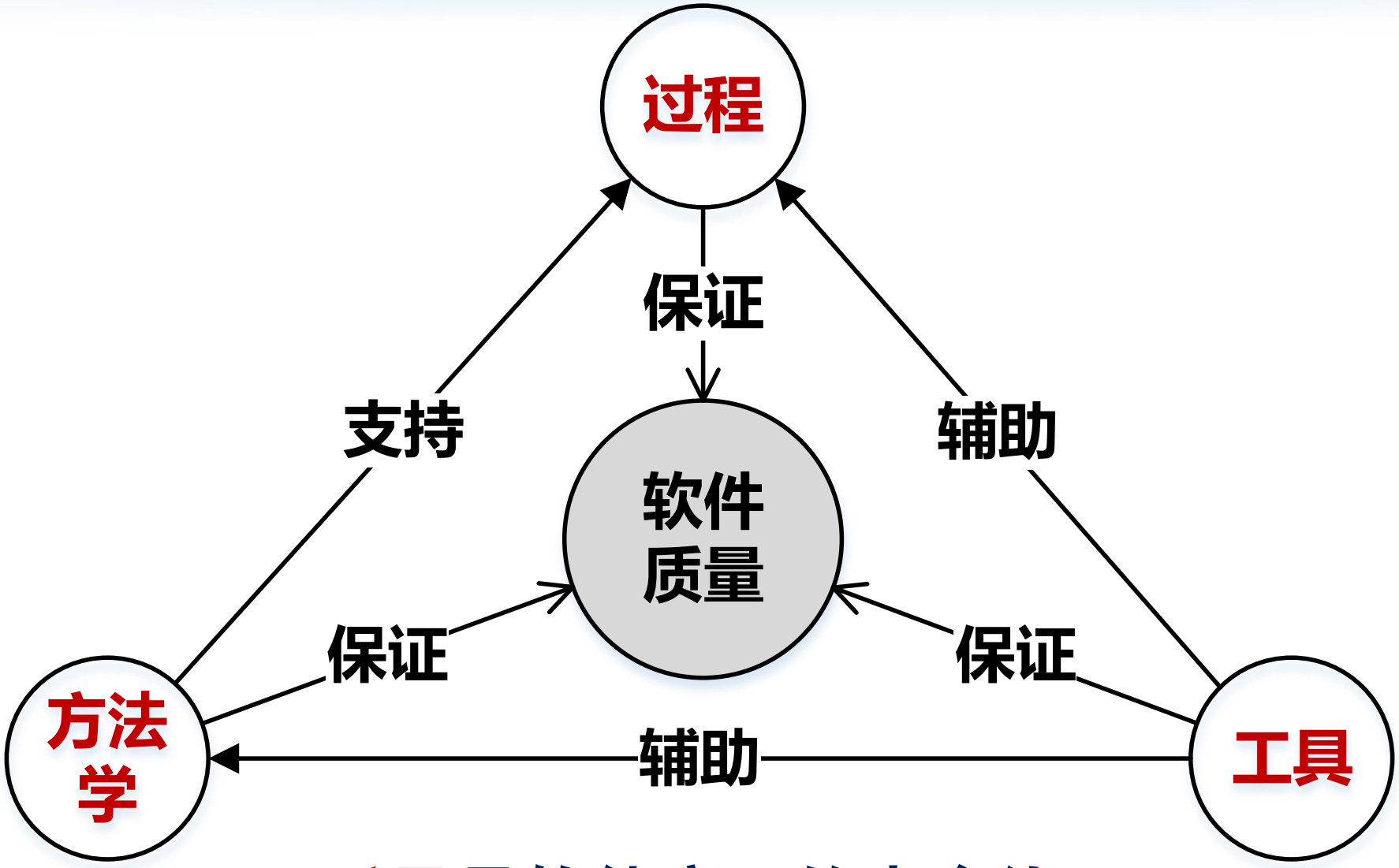
类似于建筑工程，　采用工程化的方法来解决软件开发、运维中的问题

2.3 软件工程的目标

- 在成本、进度等**约束**下，采用**工程化**的原理和**方法**，指导软件开发和运维，开发出**满足用户要求**的**足够好**软件

- 目标总结：**成本、质量、开发效率**

2.4 软件工程的三要素



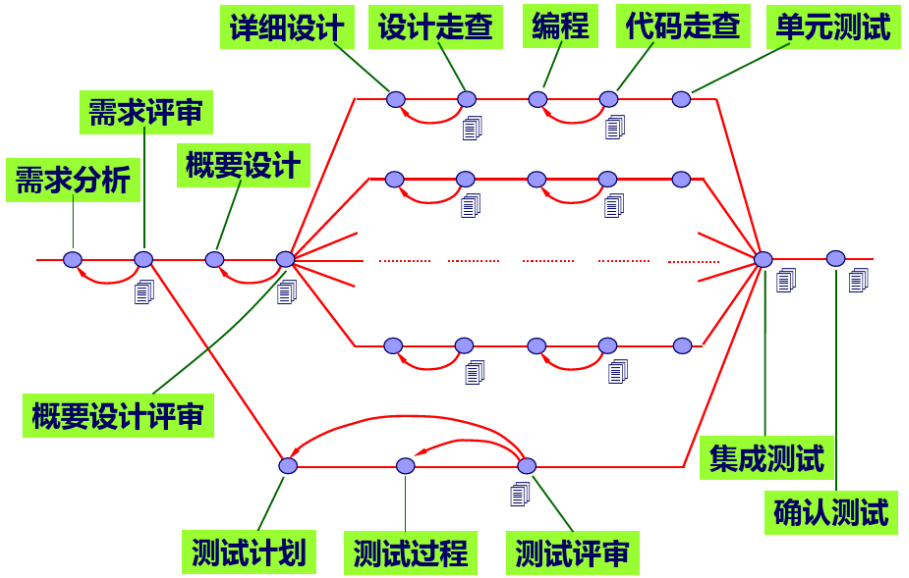
质量是软件产品的生命线

2.4.1 过程(Process)

- 为什么全世界肯德基的汉堡都是同样口味?

- 过程定义软件开发、运行和维护需要开展**哪些工作**、按照什么样的**步骤和次序**进行,以及**交付的内容**等
  - ✓回答“何时做”的问题

- 典型过程模型
  - ✓如瀑布模型、增量模型等



2.4.1 过程(Process)

（1）过程影响产品质量；（2）不同过程适用于不同项目

1、连皮猪肉，去毛洗净，放入冷水锅内煮熟。
2、将煮好的猪肉冲洗干净放凉，切成薄而大的肉片。
3、锅内放一茶匙油，放入肉片和姜片，用小火慢慢煎炒至出油，表面变金黄色。
4、加入红油豆瓣酱。翻炒至出红油后，再放入一匙生抽，料酒翻炒。
5、投入青蒜白及红、青椒，翻炒至断生即可。
6、最后放入青蒜叶，翻炒片刻即可出锅。

1. 猪肉切丝，加老抽和淀粉抓匀，放一旁备用；
2. 子姜，红椒切丝；
3. 锅里下油，油热后下少许子姜丝（因为肉抓了淀粉，容易粘锅，先下少许子姜丝，会有所改善）
4. 把肉丝放下去，翻炒至8分熟，放入剩余的子姜丝，翻炒至肉全熟，下红椒翻炒几下，放盐，翻炒均匀即可出锅。

2.4.2 方法学(Methodology)

- **从技术的视角，回答软件开发、运行和维护“如何做”的问题**
  - ✓例如：需求分析方法、设计方法、测试方法等

- **典型方法**
  - ✓结构化软件开发方法学
  - ✓面向对象软件开发方法学
  - ✓基于构件的软件开发方法学

2.4.2 工具(Tool)

- **工具为方法的运用提供自动或半自动的软件工程支撑环境**
  - ✓工欲善其事，必先利其器
  - ✓利用工具辅助，可以提高软件开发效率和质量，促进团队协作，加快软件交付进度。

- **典型工具**
  - ✓JMeter、PowerDesigner、Eclipse、Visual Studio等

2.5 计算机辅助软件工程

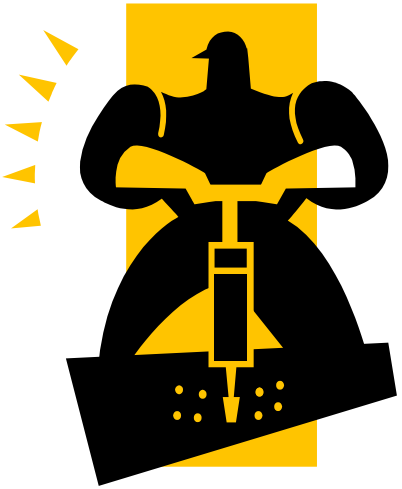
- **什么是计算机辅助软件工程(Computer-Aided Software Engineering, CASE)**
  - ✓借助**计算机软件**来辅助软件开发、运行、维护和管理的过程

- **为什么需要计算机辅助软件工程**
  - ✓提高效率。如代码生成、测试用例的创建和执行
  - ✓减少错误。如自动进行模型正确性检查，代码静态分析
  - ✓支持协作。如支持团队开发，多版本控制

CASE工具和环境

❑CASE工具

- ✓如编辑器、编译器等
- ✓具有功能单一性



❑CASE环境

- ✓将CASE工具按统一标准和接口组装起来，使工具间、人员间、各个过程间能方便交互的集成环境。
- ✓如华为CodeArts，可即开即用，随时随地在云端进行项目管理、代码托管、在线编码、代码检查、编译构建、部署、测试、发布等。

计算机辅助软件工程工具

- **代码编写**
  - ✓编辑、编译、分析、查找、代码生成等
- **项目管理**
  - ✓工作量和成本估算、制定和跟踪计划、配置和版本管理
- **软件建模**
  - ✓需求建模、UML建模、数据建模等
- **软件测试**
  - ✓测试用例自动生成、代码测试、缺陷报告等
- **软件运维**
  - ✓软件运行，管理和维护

典型的CASE工具和环境

类别	辅助活动	提供的功能	示例
软件分析与设计	业务建模、需求分析、软件设计等	可视化建模、模型分析和管理、 文档撰写等	Rational Rose, Office, StartUML, **Microsoft Visio**, ArgoUML
编码实现	编码、编译、调试等	编辑器、编译器、调试器、加载器、代码生成器等	Eclipse, Visual Studio, Android Studio, Jenkins, **Copilot**
软件质量保证	质量分析、软件测试等	代码质量分析、软件测试、缺陷管理和追踪等	**Sonarqube**, FindBugs, JUnit, ClearQuest, CheckStyles
项目管理	协同开发、配置管理、代码仓库管理等	软件仓库和版本管理、分布式协同开发、计划制定、规模和工作量估算等	**Git**, GitLab, Microsoft Project, CVS, IBM ClearCase
软件运维	软件部署、运行和维护	自动部署、运行支撑、状况监控、日志管理、权限管理等	**Docker,** K8S, Zagios

有没有你所熟悉的工具?

小结

- **软件工程产生的背景和目的**
  - ✓软件危机，持续存在，关注点不同
- **软件工程的本质**
  - ✓软件视为产品，软件开发视为工程、创作和生产相结合的过程
  - ✓三要素：过程、方法学和工具
  - ✓软件工程的基本原则

综合实践1

- **任务：理解和分析开源软件的整体情况**
- **方法**
  - ✓运行开源软件，理解软件功能；泛读开源代码，分析代码的构成，绘制出软件系统的体系结构图；利用SonarQube工具分析开源代码的质量情况
- **要求**
  - ✓理解开源软件提供的功能和服务，掌握软件系统的模块构成，分析开源软件的质量水平
- **结果：**
  - ✓（1）软件需求文档；（2）软件体系结构图，描述开源软件的模块构成；（3）SonarQube的开源软件质量报告

综合实践2

- **任务：分析相关行业和领域的状况及问题。**
- **方法**
  - ✓选择你所感兴趣的行业和领域（如老人看护、防火救灾、医疗服务、出行安全、婴儿照看、机器人应用等），开展调查研究，分析这些行业和领域的当前状况和未来需求，包括典型的应用、采用的技术、存在的不足和未来的关注。
- **要求**
  - ✓调研要充分和深入，分析要有证据和说服力。
- **结果**
  - ✓行业和领域调研分析报告。

课后实践

- **构想软件需求，形成一页纸（不超1000字）描述**
  - ✓名字：概括软件系统
  - ✓问题：描述该软件欲解决的实际问题
  - ✓方法：描述如何通过软件及相关系统来解决问题
  - ✓举例：举一个使用该软件解决问题的应用案例和具体场景
  - ✓功能：软件大致有哪些功能?
  - ✓设备：软件需要与哪些设备进行集成
- **文档格式可参见模板**

问题和讨论