



哈爾濱工業大學(深圳)

HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN

软件工程 Software Engineering

张永兵

ybzhang08@hit.edu.cn

计算机科学与技术学院



课程简介

- 任课老师:

- 张永兵
- 信息楼1702室
- ybzhang08@hit.edu.cn

- 教材及参考书

- 《软件工程：实践者的研究方法》，机械工业出版社，原书第8版，2016年11月，ISBN 9787111548973；
- 《构建之法 现代软件工程》，邹欣，人民邮电出版社，第三版，2017年6月，ISBN 9787115460769；

- 考试

- 平时作业30%、期末考试70%



教学目标

课程目标：从工程学角度认识软件在大型系统中的设计和应用，具备作为软件工程师从事工程实践所需的专业能力

能力培养目标：

- 需求定义与分析
- 权衡和选择设计方案
- 使用UML建模
- 开发高质量软件
- 运用软件工程工具
- 团队协作开发
- 谈判与沟通

转变对软件开发的认识：程序 ➡ 系统

转变思维定式：程序员 ➡ 软件工程师

教学方法

课堂：讲授知识点和基本思想，提供相关阅读资料

课后：

- 通过作业真正体会和运用软件工程的原则、方法和技术
- 在实践过程中内化和提升理论知识
- 培养产品意识、工程思维、解决问题能力及团队协作精神

课程章节安排

- 第一部分 软件工程概论
- 第二部分 软件项目开发过程与管理
- 第三部分 软件需求工程
- 第四部分 软件设计
- 第五部分 软件编码、测试与质量保障
- 第六部分 软件实施、维护与演化
- 第七部分 案例分析
- 课程作业

理论课：课堂

课后作业

第一部分 软件工程概论

- 1.1 软件工程的产生与发展
- 1.2 软件工程的基本概念

1.1 软件工程的产生与发展

- 什么是软件工程
- 软件的本质与工程化的方法
- 软件开发面临的挑战
- 软件的本质特性
- 软件开发问题的解决途径

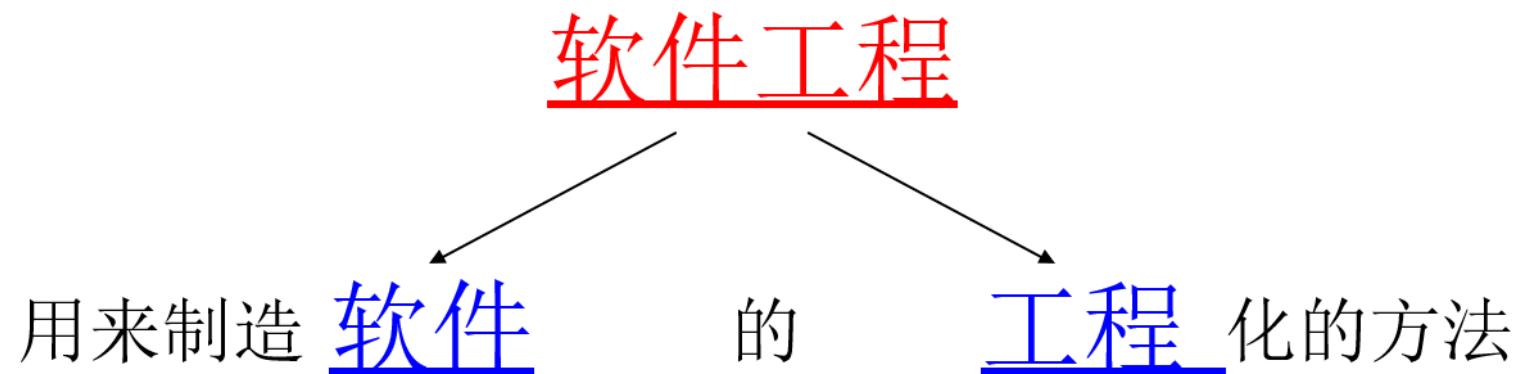
软件无处不在

软件无处不在，为我们的生活创造了无限精彩。在当今的信息时代，世界正在变得更加“智慧”，万事万物间感知化、互联化和智能化的程度不断加深。

软件工程为这一切做出了巨大贡献，随着该学科的成熟发展，其未来的贡献将不可限量。



什么是“软件工程”？



- 传统的理解：
 - 软件：控制计算机硬件功能及其运行的指令、程序和符号语言
 - 工程：将科学及数学原理运用于实际用途的应用手段，即高效率、低成本的设计、制造和运行各类结构、机器、进程和系统
- 软件工程：应用软件领域的理论方法来解决软件系统“从无到有”、“从有到好”的过程。

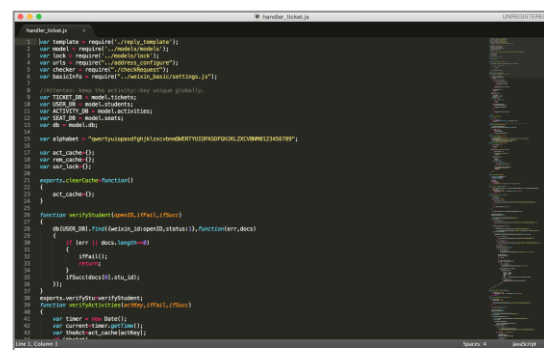
什么是“软件”

- 程序=? 软件
- 程序 (**Program**) 是为实现特定目标或解决特定问题而用计算机语言编写的命令序列的集合。
- 软件 (**Software**) 软件是程序、数据以及开发、使用和维护所需要的所有文档的总称。

软件的定义？

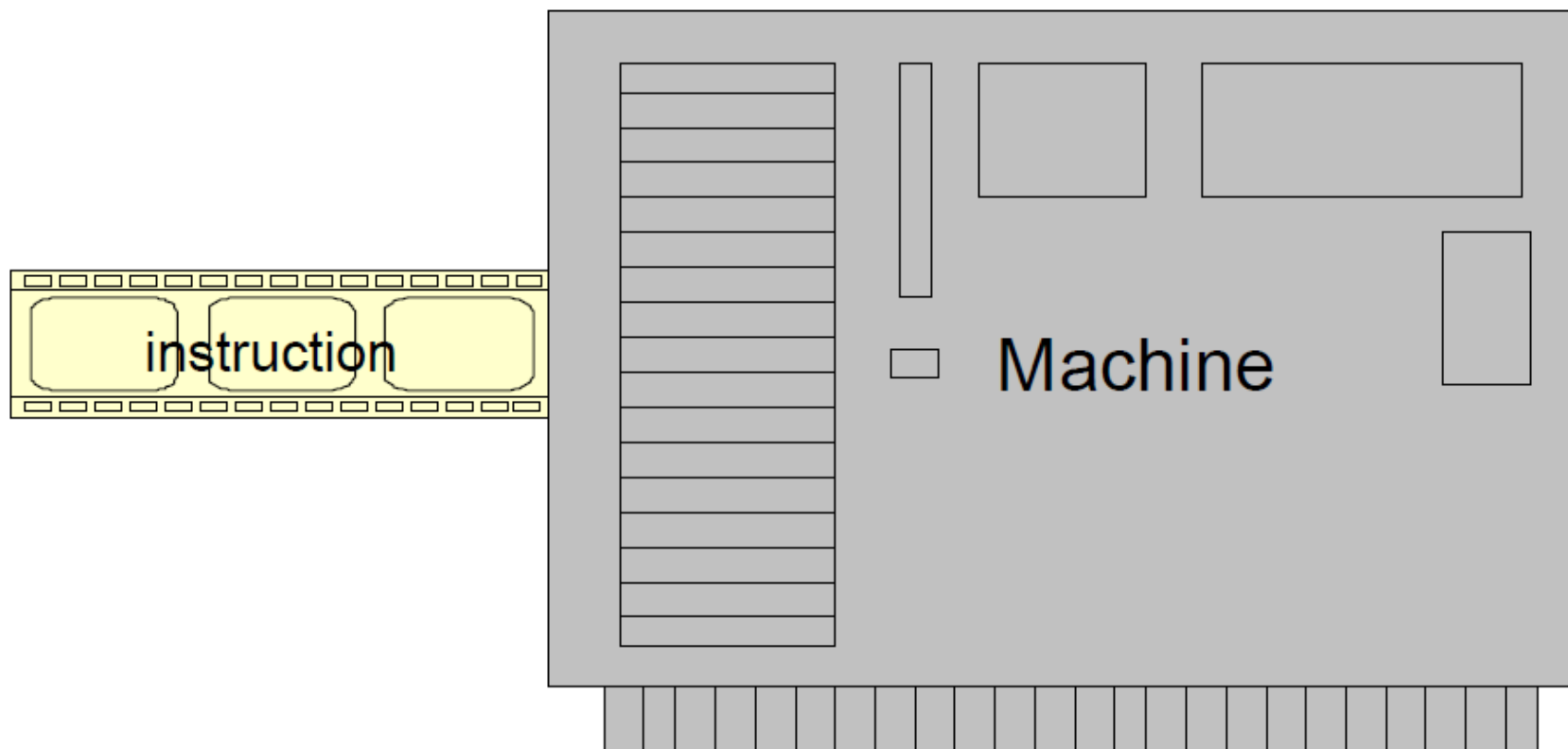
软件 = 程序 + 数据 + 文档

- 程序：计算机可以接受的一系列指令，运行时可以提供所要求的功能和性能。
- 数据：使得程序能够适当地操作信息的数据结构。
- 文档：描述程序的研制过程、方法和使用的图文资料。



软件角色的转变：软件=程序

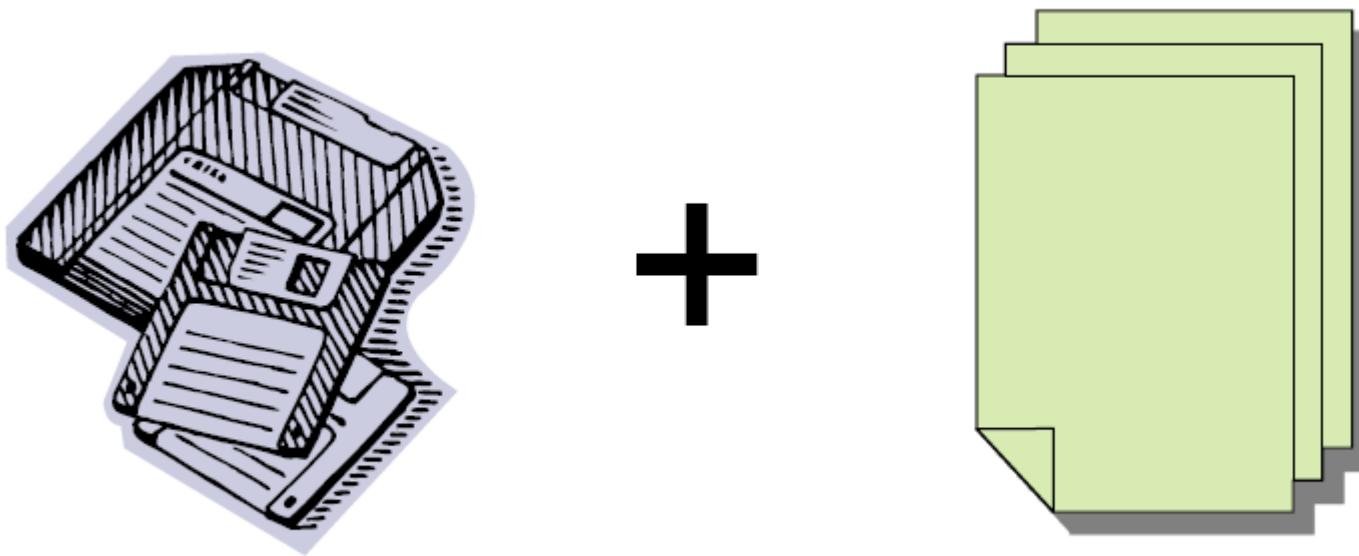
Before 1960' s



软件角色的转变：软件=工具

Software = programs + documents + data

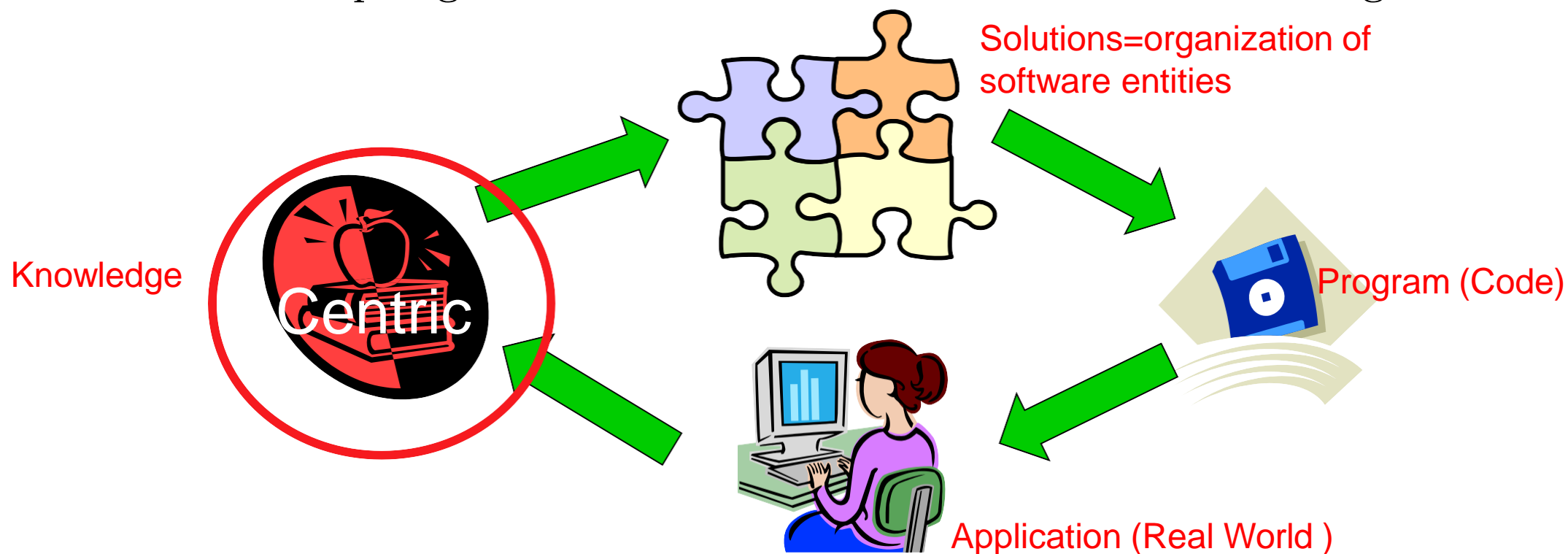
Before 1980' s



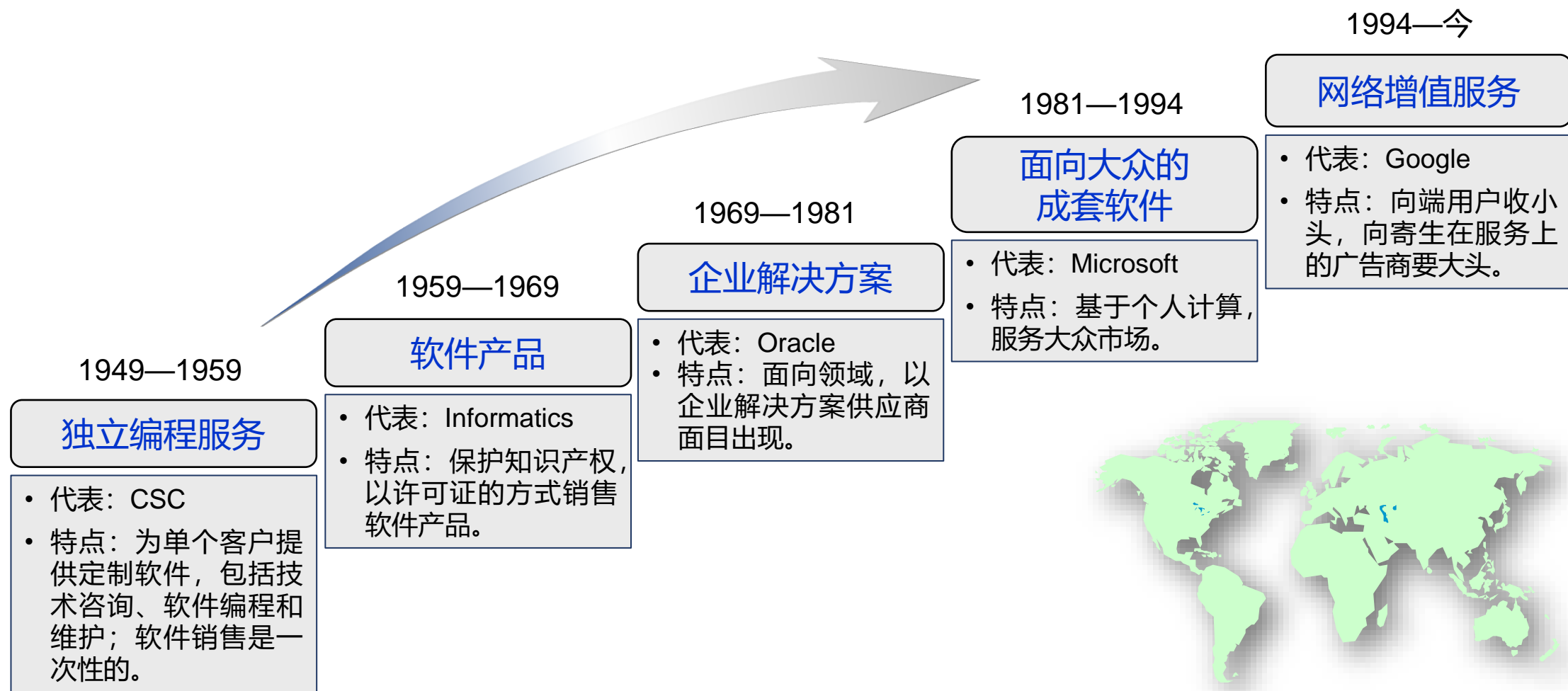
软件角色的转变：软件=服务

软件中承载知识

Software = programs + documents + data + knowledge



软件发展历程



软件角色的演化

- 信息（知识）时代软件的作用
 - 知识的管理者
 - 知识的传递者
 - 知识的生产者

软件的本质特性



一架客机由数百万个单独的部件组成，需要上千人组装，但通常都能够按时按预算交付使用



微软于**1989**年**11**月发布的**Word**最初版本，花费了**55**人年，大约有**249,000**行源代码，却晚了**4**年交付使用。

软件的本质特性



开发软件与建造客机有
什么本质的区别？

软件的本质特性

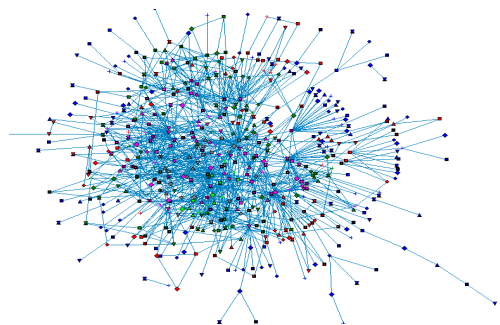
软件具有**复杂性**、**一致性**、**可变性**和**不可见性**等固有的内在特性，这是造成软件开发困难的根本原因。



Brooks, F. P., “No silver bullet: essence and accidents of software engineering”, *IEEE Computer*, Vol. 20, No. 4, pp.10-19, 1987

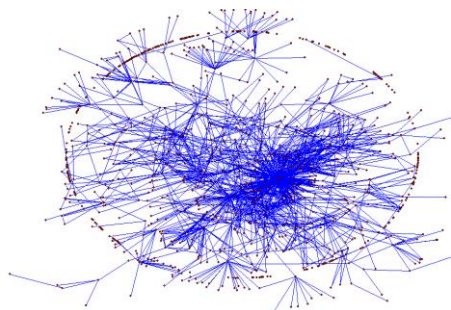
软件的本质特性：复杂性

软件是人类思想的外延，人们将自己的思想传送给计算机，当产生的可执行文件被激活运行时，软件便重现人类的意图。



Linux内核有630个函数，存在1814个函数调用。

注：图中将函数表示为节点，调用关系表示为边。

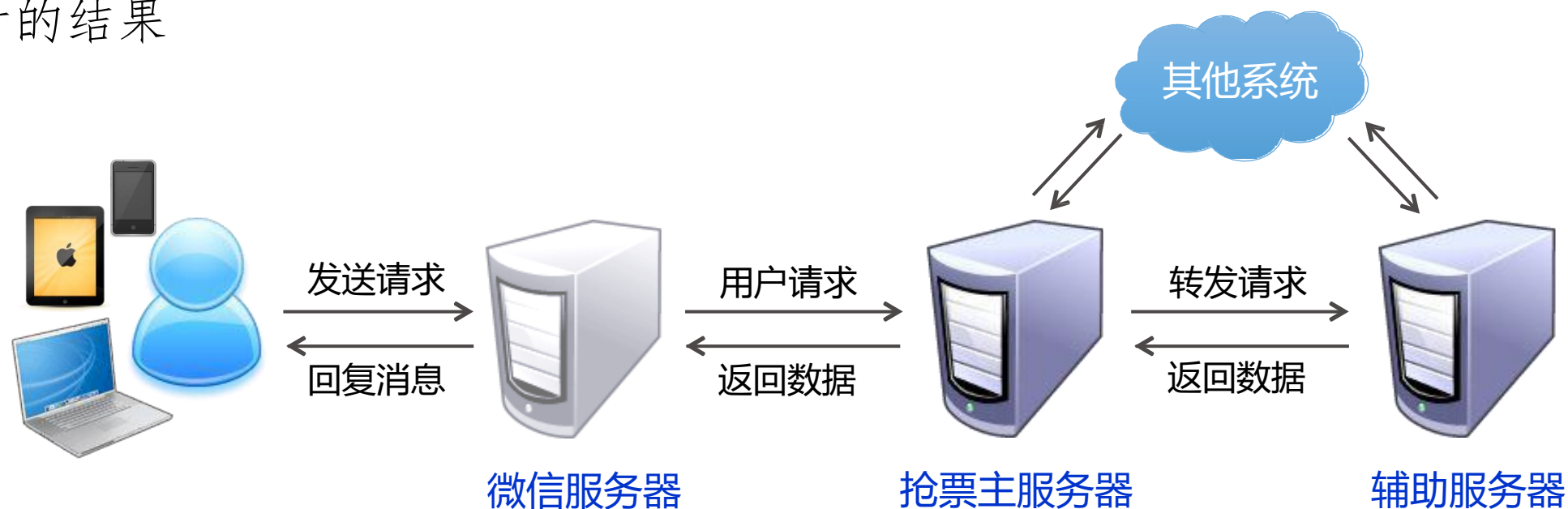


Tomcat 软件含1019个类，类与类之间有2109个继承或聚合关系。

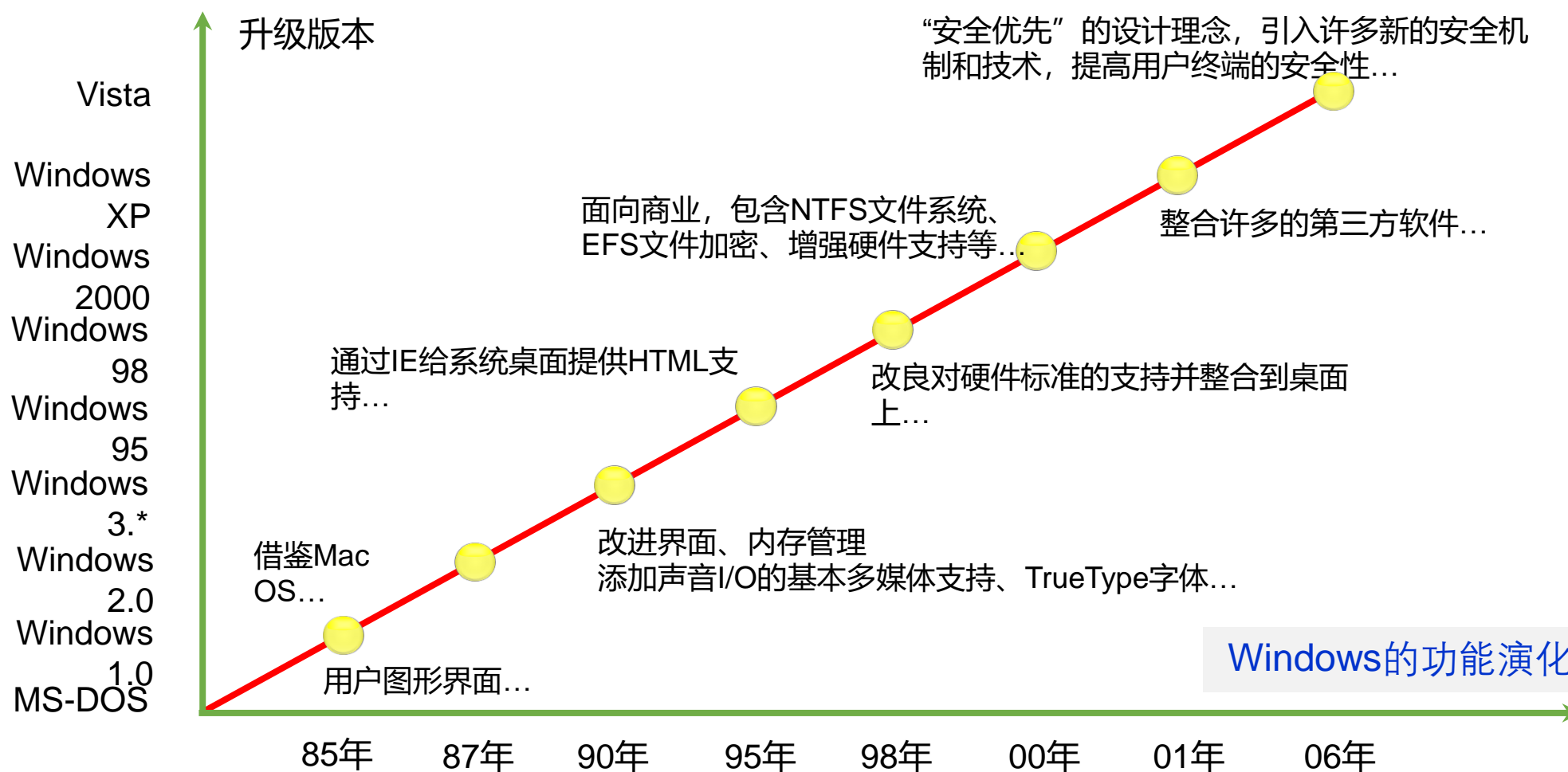
注：图中将类表示为节点，关系表示为边。

软件的本质特性：一致性

- 软件不能独立存在，需要依附于一定的环境（如硬件、网络以及其他软件）
- 软件必须遵从人为的惯例并适应已有的技术和系统
- 软件需要随接口不同而改变，随时间推移而变化，而这些变化是不同人设计的结果

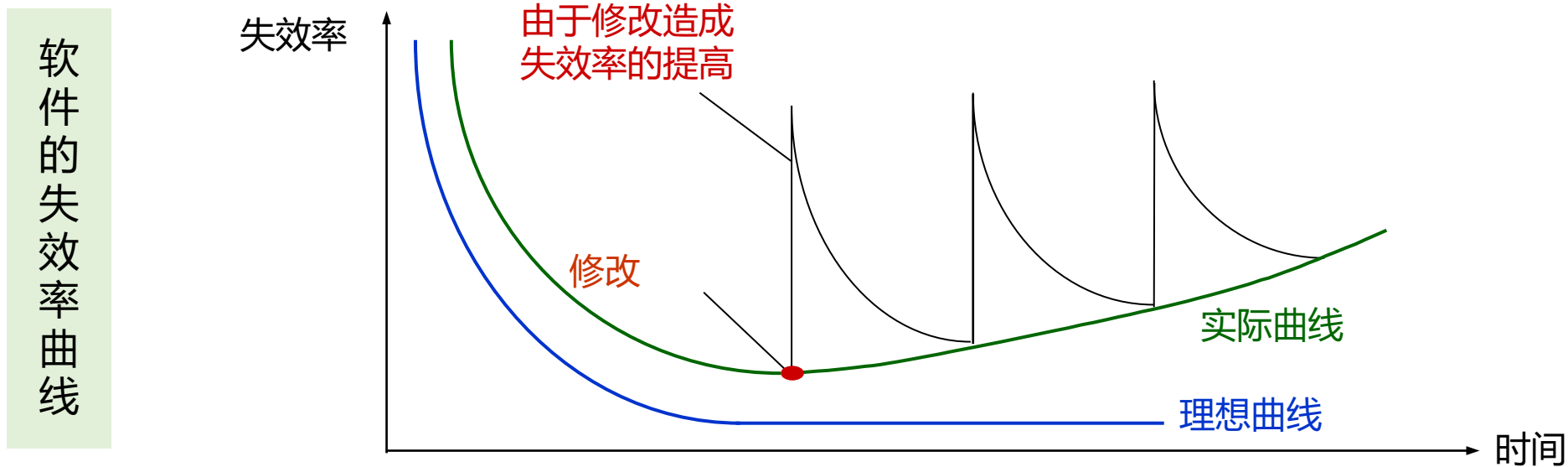


软件的本质特性：演化性

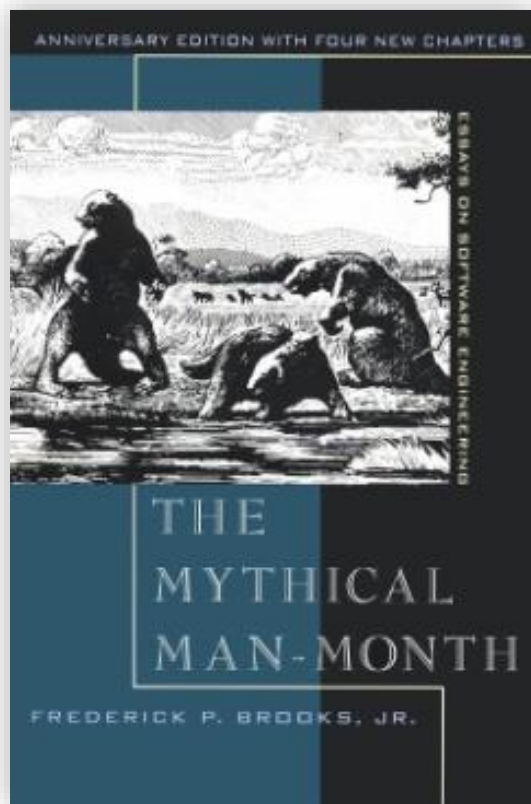


软件的本质特性：演化性

- 人们总是认为软件是容易修改的，但忽视了修改所带来的副作用
- 不断的修改最终导致软件的退化，从而结束其生命周期



软件的本质特性：不可见性



软件人员太像“皇帝的新衣”故事中的裁缝了。当我来检查软件开发工作时，所得到的回答好象对我说：我们正忙于编织这件带有魔法的织物。只要等一会儿，你就会看到这件织物是极其美丽的。但是我什么也看不到，什么也摸不到，也说不出任何一个有关的数字，没有任何办法得到一些信息说明事情确实进行得非常顺利，而且我已经知道许多人最终已经编织了一大堆昂贵的废物而离去，还有不少人最终什么也没有做出来。

软件的本质特性

- 软件所具有的复杂性、一致性、可变性、不可见性等特性，使得软件开发过程变得难以控制，开发团队如同在焦油坑中挣扎的巨兽。



举例：ARIANE 5 火箭

1996年6月4日，Ariane 5火箭在发射37秒之后偏离其飞行路径并突然发生爆炸，当时火箭上载有价值5亿美元的通信卫星。



原因：

- 程序中试图将64位浮点数转换成16位整数时产生溢出
- 缺少对数据溢出的错误处理程序
- 备份软件通过复制而成

举例：Windows Vista系统



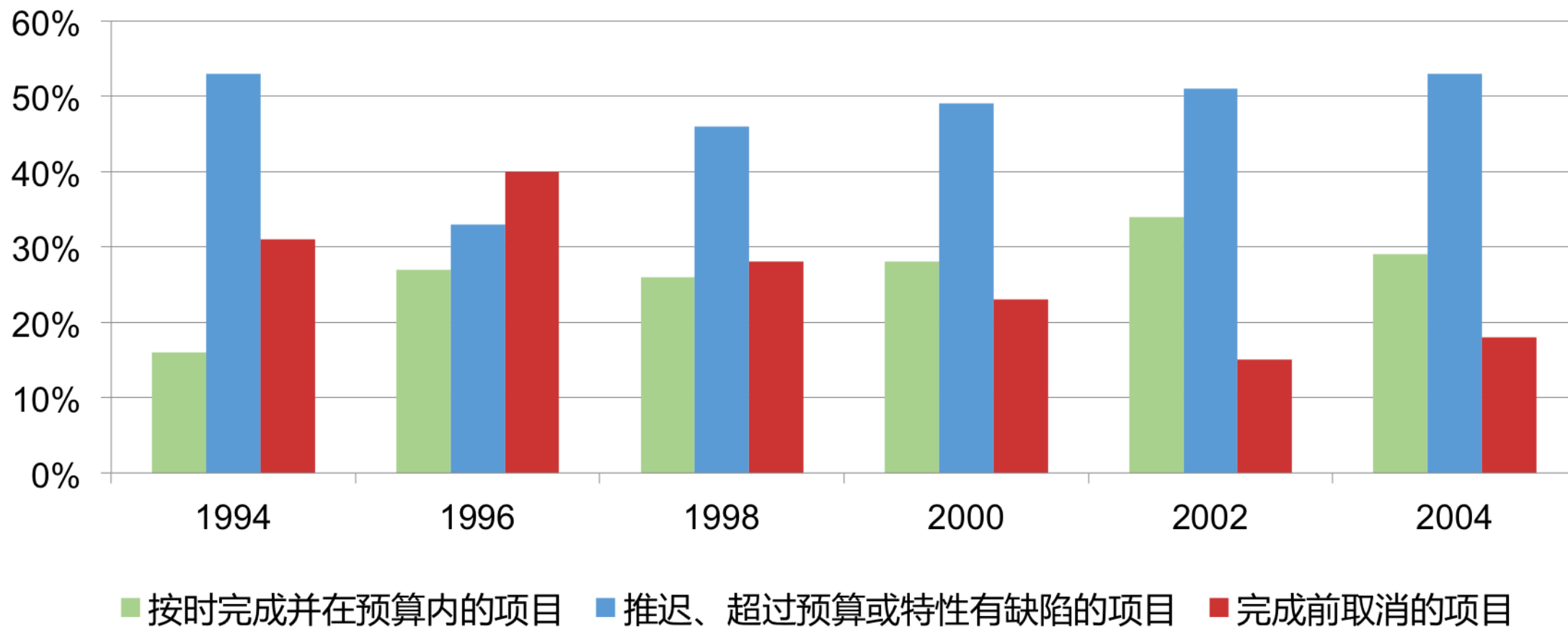
- 该系统从2001年开始研发，整个过程历时5年半，先后有9000位开发人员投入其中，耗资60亿美元，代码规模超过5000万行。
- 按照微软公司最初的计划，该系统面世时间应该在2003年，之后推迟到2004年下半年再到2005年初，最终在取消一些高级功能后于2006年11月正式发布。
- 由于整个系统过于庞杂，其开发管理相当混乱，以致于很多时间用在互相沟通和重新决定上。
- 从Vista Beta 1进入公开测试以来，程序错误总数已经超过2万个，这其中还不包括微软内部未公开的一些错误。

举例：12306 网络购票系统



- 12306网络购票系统历时两年研发成功，耗资3亿元人民币，于2011年6月12日投入运行。
- 2012年1月8日春运启动，9日网站点击量超过14亿次，出现网站崩溃、登录缓慢、无法支付、扣钱不出票等严重问题。
- 2012年9月20日，由于正处中秋和“十一”黄金周，网站日点击量达到14.9亿次，发售客票超过当年春运最高值，出现网络拥堵、重复排队等现象。

美国Standish集团的调查报告



软件开发面临的挑战

- 交付的许多功能不是客户需要的
- 交付的日期没有保障
- 客户使用时发现许多Bug

客户
不满意

- 开发团队专注技术，忽视风险
- 无能力预测成本，导致预算超支

风险与
成本问题

- 客户需求变化频繁，无力应对
- 无法预见软件的交付质量
- 对流程盲目遵从，忽视客户业务价值

项目过程
失控

无力管理
团队

- 无法评估开发人员能力及工作进度
- 困扰于如何提升团队的能力与效率

一个值得思考的问题

为什么一个看似简单的东西，却很有可能变成一个落后进度、超出预算、存在大量缺陷的怪物？

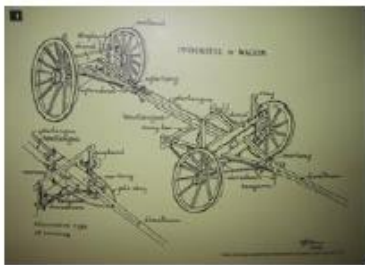


Brooks, F. P., *The Mythical Man Month*, Addison-Wesley, 1975

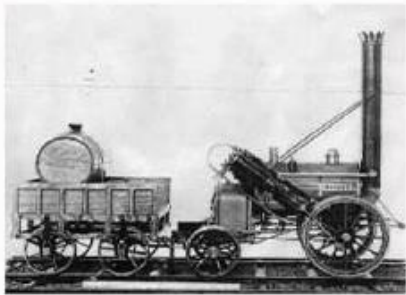
《人月神话》源于Brooks在IBM公司担任OS/360系统项目经理时的实践经验

什么是工程化的方法？

工程是将理论和所学的知识应用于实践的科学，以便经济有效地解决实际问题。

A technical drawing of a mechanical device, possibly a steam engine component, with various parts labeled. The drawing is on aged paper with a grid.

- craft
 - personal skill
 - experience
 - flexible materials
- engineering
 - tables and tools
 - recorded knowledge
 - controlled materials

A black and white photograph of a steam locomotive, showing its boiler, wheels, and smokestack.

■ 规模上的差异

- 花园小道 vs. 汽车高速公路
- 树上小屋 vs. 摩天大楼
- 加法程序 vs. 医院档案系统

■ 手工 (Craft)：小规模的设计与建造

- 简单问题与单一目标
- 个人控制与个人技能

■ 工程 (Engineering)：大规模的设计与建造

- 复杂问题与目标分解
- 团队协作，需要考虑运营、管理、成本、质量控制、安全等

什么是工程化的方法？

需求分析和定义

软件设计

编码实现

软件测试

软件交付与维护

工程的特征

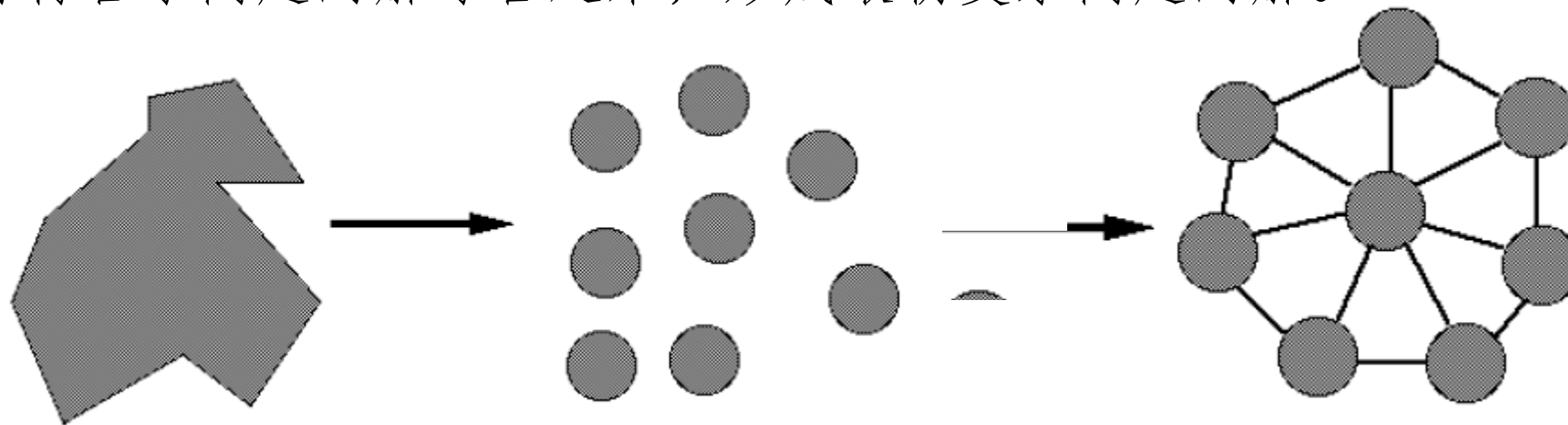
- 平衡与决策
- 度量与验证
- 训练有素的过程
- 团队协作与角色分工
- 系统地运用工具
- 工程原则、标准和实践
- 重用设计和设计制品

注重系统的构建过程

工程化思想

分而治之 (Divide and Conquer):

- 将复杂问题分解为若干可独立解决的简单子问题，并分别独立求解，以降低复杂性；
- 然后再将各子问题的解综合起来，形成最初复杂问题的解。



Rome was not built in a day !

核心问题：如何的分解策略可以使得软件更容易理解、开发和维护？

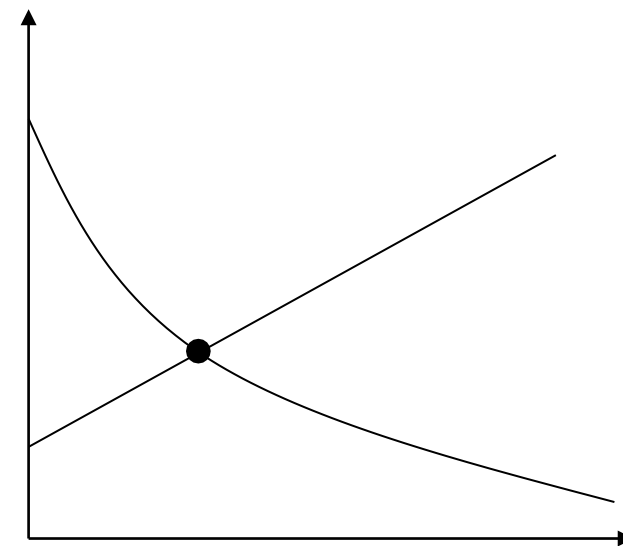
工程化思想

折中 (Trade-off)

不同的需求之间往往存在矛盾与冲突，需要通过折中来作出合理的取舍，找到使双方均满意的点。

例如：

- 在算法设计时要考虑空间和时间的折中；
- 低成本和高可靠性的折中；
- 安全性和速度的折中；



The doctrine of the mean, the art of balance

核心问题：如何调和矛盾(需求之间、人与人之间、供需双方之间，等等)

工程化思想

不同类型的软件对质量目标的要求各有侧重：

- 实时系统：侧重于可靠性、效率；
- 生存周期较长的软件：侧重于可移植性、可维护性；

多个目标同时达到最优是不现实的：目标之间相互冲突

	功能性	可靠性	可用性	效率	可维护性	可移植性
功能性		↑			↑	
可靠性				↓		↑
可用性				↓	↑	↑
效率		↓			↓	↓
可维护性		↑		↓		↑
可移植性		↓		↓		

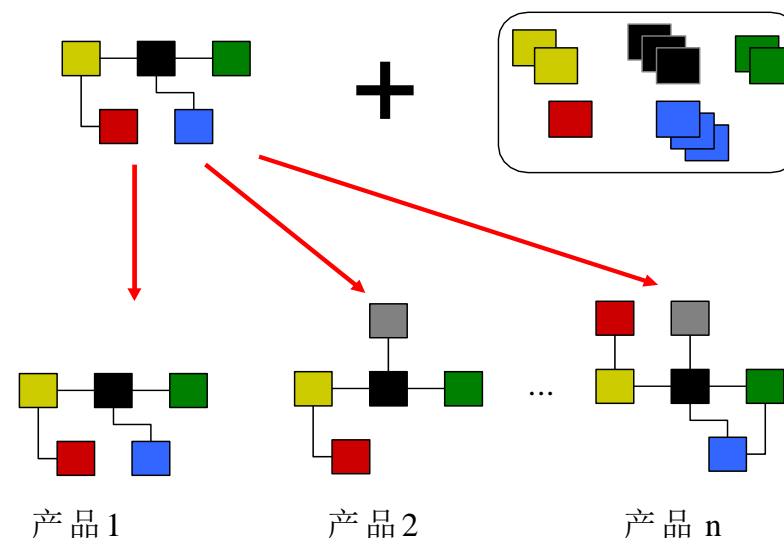
工程化思想

复用 (Reuse)

- 在一个新系统中，大部分的内容是成熟的，只有小部分内容是全新的。
- 构造新的软件系统可以不必每次从零做起；
- 直接使用已有的软构件，即可组装成新的系统；
- 复用已有的功能模块，既可以提高开发效率，也可以改善新开发过程中带来的质量问题；

产品线体系结构

可复用资产库



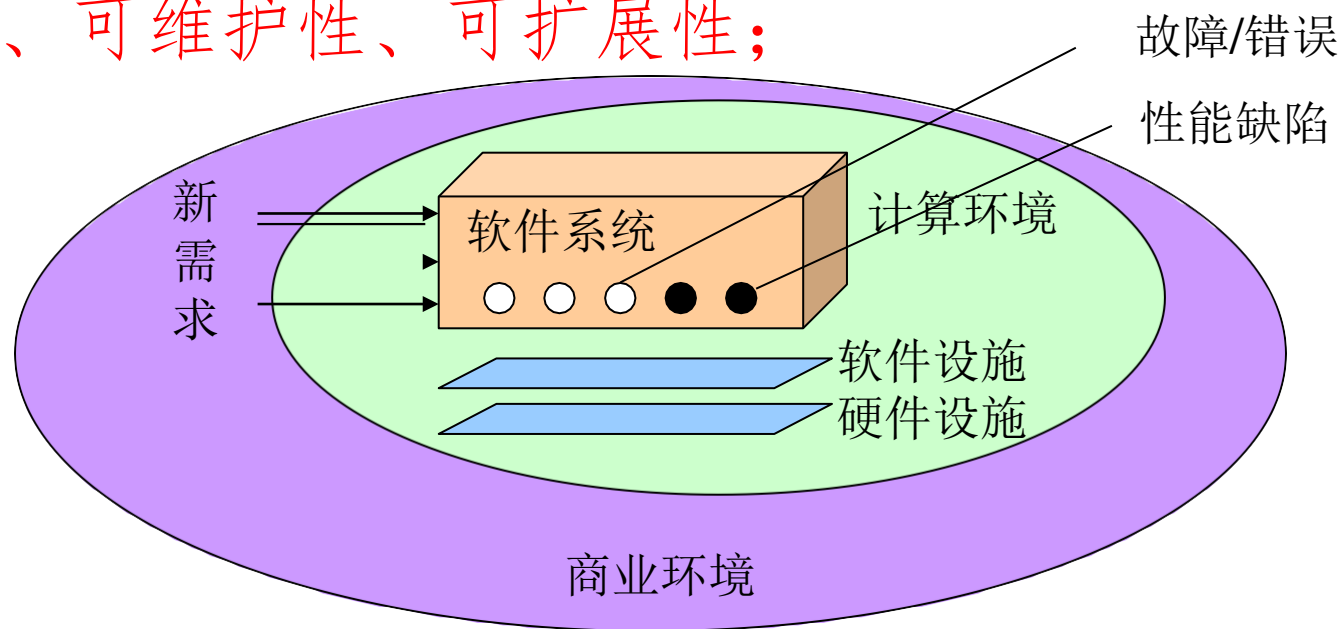
Don't re-invent the wheel !
核心问题：如何可以让你的软件变得更有价值？

工程化思想

软件系统在其生命周期中面临各种变化；

- 核心问题：在设计软件的初期，就要充分考虑到未来可能的变化，并采用恰当的设计决策，使软件具有适应变化的能力。
- 即：可修改性、可维护性、可扩展性；

Continuing change
and increasing
complexity



归纳起来，“软件工程”是...

软件工程是 ① 将系统性的、规范化的、可量化的方法应用于软件的开发、运行和维护，即工程化应用到软件上；② 对①中所述方法的研究。

软件工程的基本目标：

- 较低的开发成本
- 按时完成开发任务并及时交付
- 实现客户要求的功能
- 所开发软件具有良好的性能
- 较高的可靠性、可扩展性、可移植性
- 软件维护费用低



软件开发面临的挑战

软件创新的最大障碍：复杂性

- 复杂的基础架构
- 缺乏适用的开发流程
- 存在众多的应用系统
- 每年扩展现存系统并开发新系统
- 软件生产环境本身的复杂度



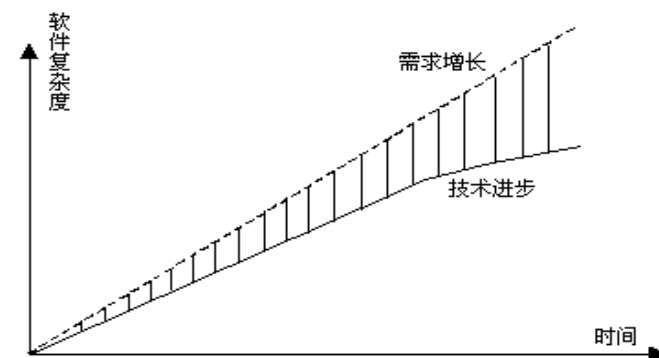
软件生产过程中面临的问题

■ 软件生产过程面临的问题

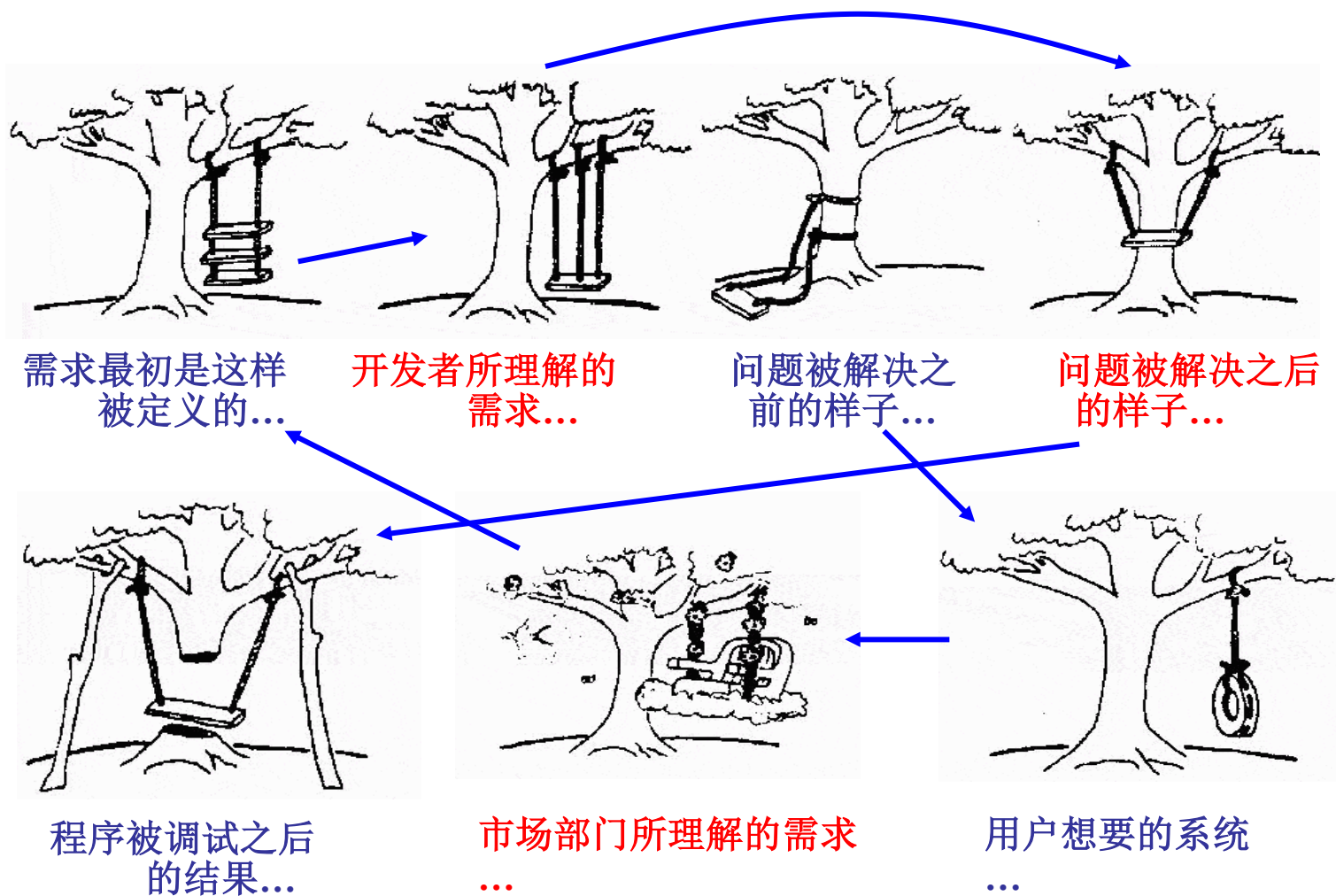
- 为什么软件需要如此长的开发时间？
- 为什么开发成本居高不下？
- 为什么在将软件交付顾客之前，我们无法找到所有的错误？
- 为什么维护已有的程序要花费高昂的时间和人力代价？
- 为什么软件开发和维护的过程仍旧难以度量？

为何出现这种情况？

- **客观上：软件产品开发的复杂度和难度随软件规模呈指数级别增长**
 - 随着软件规模的急速增长，传统的软件开发方法已经不可用了
- **主观上：软件开发人员缺乏工程性的、系统性的方法论**
 - 程序员具有编程的能力，但对软件开发这一过程性较强的任务却缺乏足够的工程化思维；
 - 对软件开发的一些认识的误区：软件神话 (Software myths)；
 - 没有将“软件产品研发”与“程序编码”区分清楚；
 - 忽视需求分析、轻视软件维护；

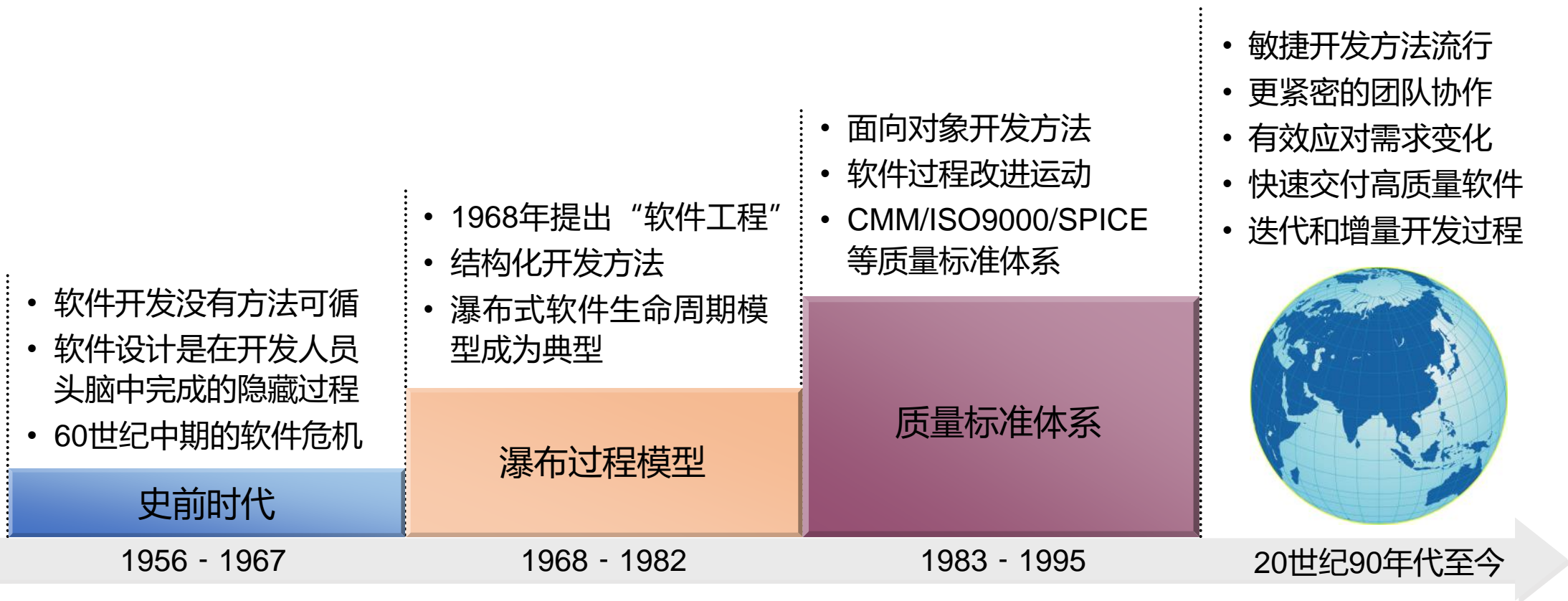


软件通常是按这样的方式构造出来的

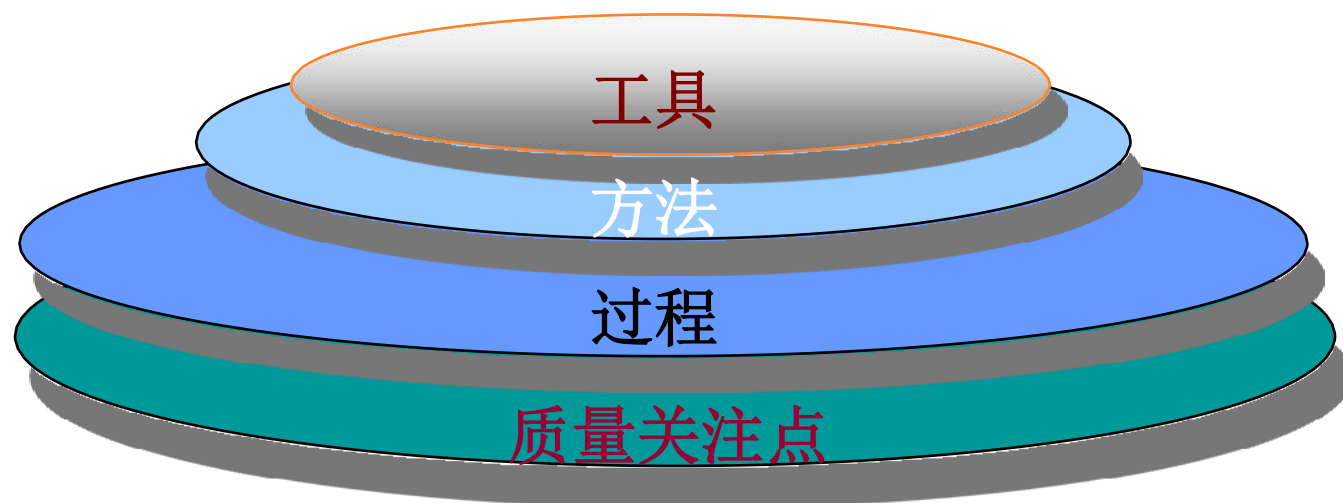


软件开发问题的解决途径

软件工程诞生：1968年北大西洋公约组织（NATO）召开国际会议，提出“**软件工程**”概念和术语。



归纳起来, 软件工程的内容



工具： 为过程和方法提供自动或半自动支持，如：建模工具，测试工具

方法： 技术上的解决办法，如：面向对象分析、构件设计

过程： 确定若干框架活动，软件项目管理的基础

质量关注点： 质量是基础，如：可靠性、易用性

1.2 软件工程的基本概念

- 软件工程的基本要素
- 软件工程所关注的目标
- 软件工程学科发展
- 什么是好的软件

什么是软件工程

软件工程是一项受软件工程原理指导的活动，软件工程师需要捕捉和理解一个系统的基本原理模型，并根据评价标准提出合理的开发决策。



举例：千年虫问题

一个基本原理是决策背后的动机：

- 描述要解决的问题（例如年份如何表示？）
- 可选择解决方案（例如用2个字符还是4个字符表示年份）
- 用于指导决策的准则（例如使内存所占空间最小）
- 软件工程师针对可选的解决方案进行讨论
- 为了解决问题而做出的决策（例如决定用2个字符表示年份）

软件工程的基本要素

为软件开发过程
提供“如何做”的技术

方法

支持软件生命周期的
所有活动

过程

为软件开发方法
提供自动的或半自动的
软件支撑环境

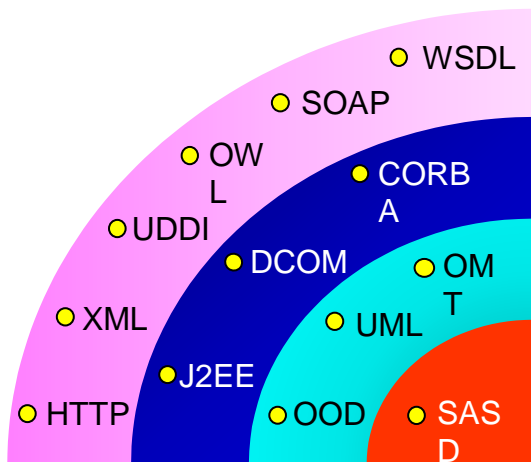
工具

构造软件的
特定方法或者风格

范型

软件
质量

软件工程方法（范型）



面向服务：在应用表现层次上将软件构件化，即应用业务过程由服务组成，而服务由构件组装而成。

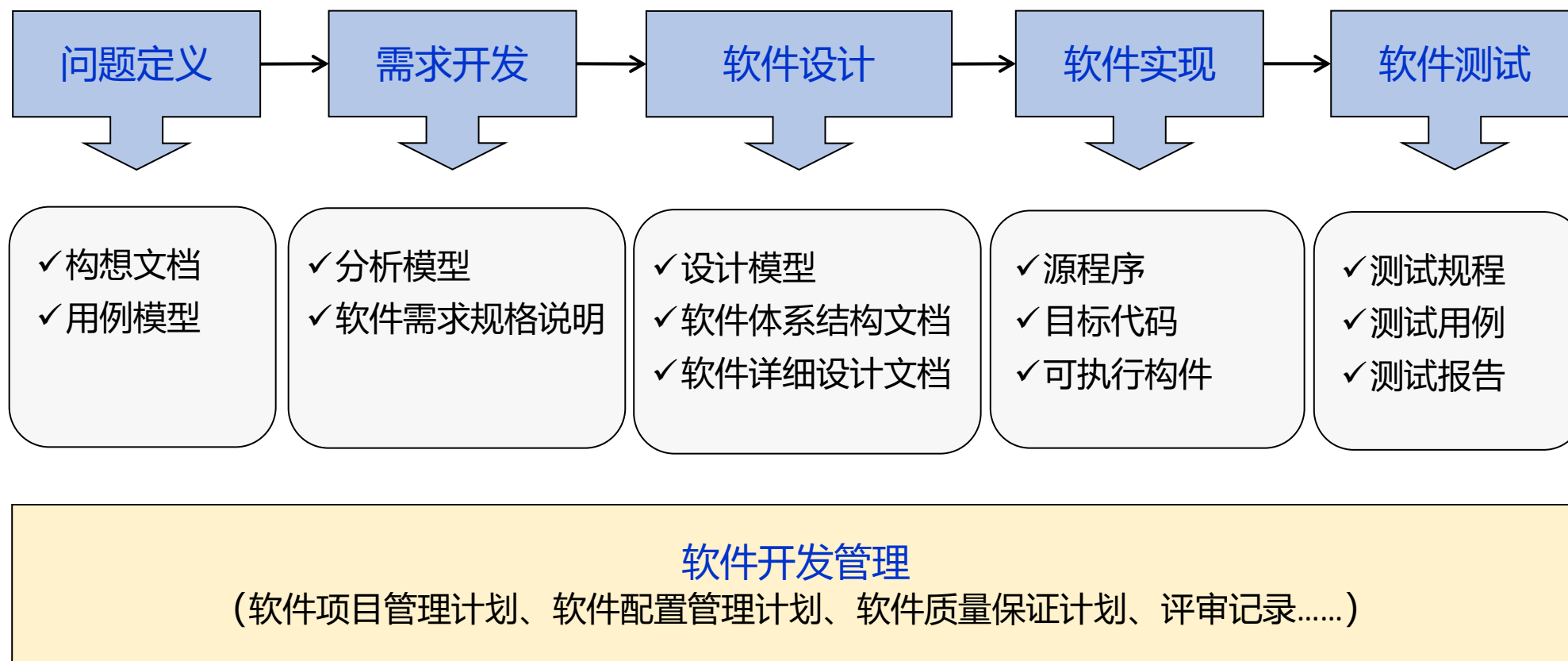
面向构件：寻求比类的粒度更大的且易于复用的构件，期望实现软件的再工程。

面向对象：以类为基本程序单元，对象是类的实例化，对象之间以消息传递为基本手段。

面向过程：以算法作为基本构造单元，强调自顶向下的功能分解，将功能和数据进行一定程度的分离。

SASD	Structure Analysis, Structure design	结构化设计方法
OOD	Object-Oriented Programming	面向对象设计
UML	Unified Modeling Language	统一建模语言
OMT	Object Modeling Technology	对象建模方法
CORBA	Common Object Request Broker Architecture	公共对象请求代理体系结构
DCOM	Distribute Component Object Model	分布式组件对象模型
J2EE	Java2 Platform Enterprise Edition	JAVA平台企业版
HTTP	Hypertext Transfer Protocol	超文本传输协议
XML	Extensible Markup Language	可扩展标记语言
UDDI	Universal Description, Discovery and Integration	统一描述、发现和集成
OWL	Ontology Web Language	本体Web语言
SOAP	Simple Object Access Protocol	简单对象访问协议
WSDL	Web Service Describe Language	网络服务描述语言

软件工程过程



举例：麦当劳的过程管理

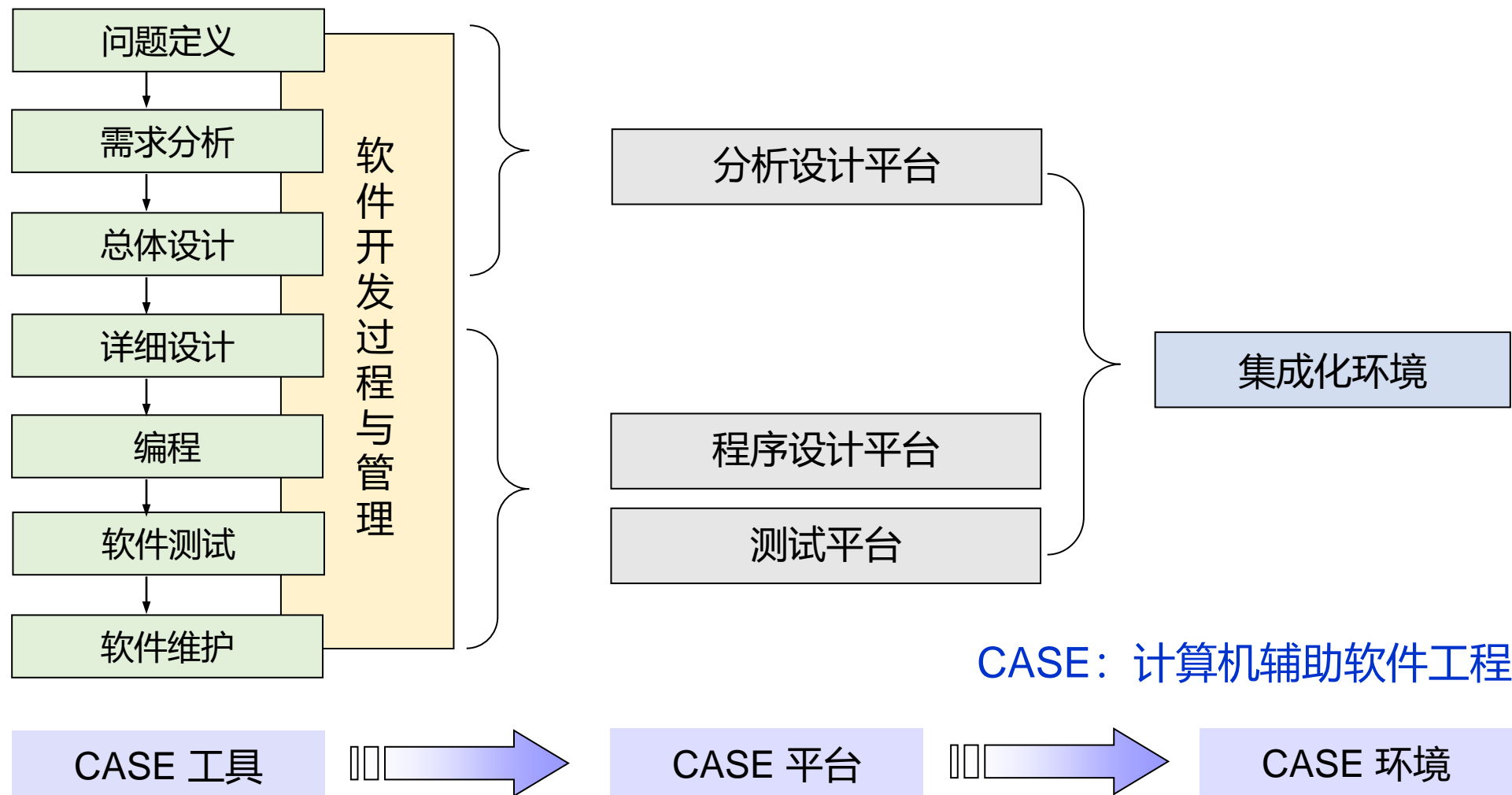


- 流水线作业：机械化、批量化、标准化
- 四无主义：无论何人，无论何时，无论何地，无品质差异
- 质量控制：在过程细节上精益求精

精益求精的细节管理：

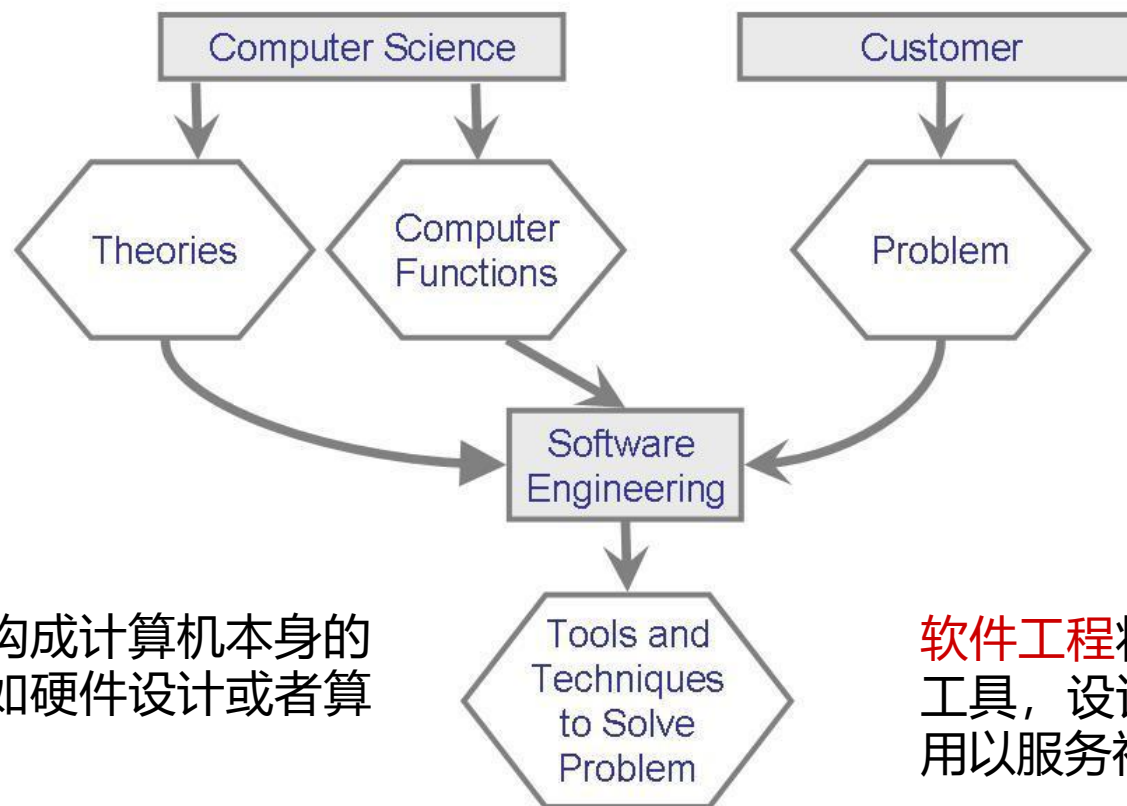
- 面包直径均为17cm，这种尺寸入口最佳
- 面包中的气泡全部为0.5cm，这种尺寸味道最佳
- 牛肉食品的品质检查有40多项内容，从不懈怠
- 肉饼成分讲究，由83%肩肉与17%五花肉混制而成
- 牛肉饼重量在45克时其边际效益达到最大
- 汉堡包从制作到出炉时间严格控制在5秒钟
- 一个汉堡包净重1.8盎司，其中洋葱重量为0.25盎司
- 汉堡包出炉后超过10分钟，薯条炸好后超过7分钟，不准售出
- 汉堡包饼面上如有人工手压轻微痕迹，不准售出
- 与汉堡包一起卖的可口可乐必须是4℃，这个温度最可口
- 柜台高度92cm，绝大多数顾客付账取物最方便
- 避免顾客等待30秒以上，这是人与人对话时产生焦虑的临界点

软件工程工具



软件工程与计算机科学的区别

科学是发现世界上已经存在的事物，回答“是什么”和“为什么”的问题。
工程是创造世界上从未存在的事物，回答“做什么”和“怎么做”的问题。



计算机科学研究构成计算机本身的理论和结构，诸如硬件设计或者算法的理论证明等。

软件工程将计算机作为问题求解的工具，设计和实施尚未存在的方案用以服务社会。

举例：电子商务网站商品推荐

学术界：

- 基于数据本身评价推荐效果，如准确率等；
- 如果一个用户本来打算购买《软件工程》这本书，那么推荐系统准确预测该结果是否有价值？

工业界：

- 基于商业应用评价推荐效果，如受推荐影响的转化率和购买率等；
- 对于用户而言，希望推荐系统使其购买更多并非预想的书；对于出版社而言，希望推荐系统增加图书的潜在购买人群。



软件工程所关注的目标

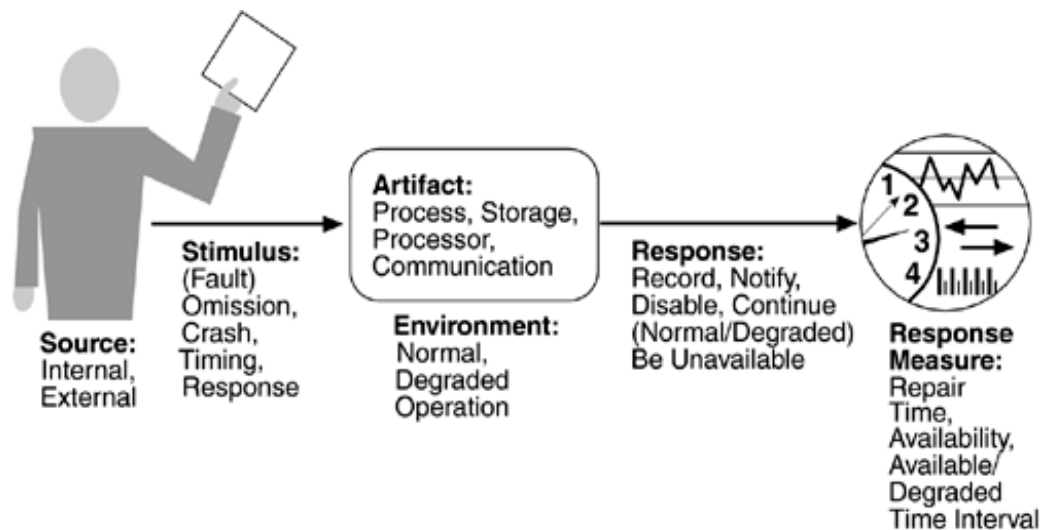
- **功能性需求(Functional Requirements):** 软件所实现的功能达到它的设计规范和满足用户需求的程度
 - 功能1、功能2、...、功能n。
 - 完备性: 软件能够支持用户所需求的全部功能的能力;
 - 正确性: 软件按照需求正确执行任务的能力;
 - 健壮性: 在异常情况下, 软件能够正常运行的能力
 - 容错能力;
 - 恢复能力;
 - 正确性描述软件在需求范围之内行为, 而健壮性描述软件在需求范围之外行为。
 - 可靠性: 在给定的时间和条件下, 软件能够正常维持其工作而不发生故障的能力。

软件工程所关注的目标

- **非功能性需求(Non-Functional Requirements):** 系统能够完成所期望的工作的性能与质量
 - **效率:** 软件实现其功能所需要的计算机资源的大小, “时间-空间”;
 - **可用性:** 用户使用软件的容易程度, 用户容易使用和学习;
 - **可维护性:** 软件适应“变化”的能力, 系统很容易被修改从而适应新的需求或采用新的算法、数据结构的能力;
 - **可移植性:** 软件不经修改或稍加修改就可以运行于不同软硬件环境(CPU、OS和编译器)的能力;
 - **清晰性:** 易读、易理解, 可以提高团队开发效率, 降低维护代价;
 - **安全性:** 在对合法用户提供服务的同时, 阻止未授权用户的使用;
 - **兼容性:** 不同产品相互交换信息的能力;
 - **经济性:** 开发成本、开发时间和对市场的适应能力。
 - **商业质量:** 上市时间、成本/受益、目标市场、与老系统的集成、生命周期长短等。

典型NFR举例：可用性(availability)

- 当系统不再提供其规格说明中所描述的服务时，就出现了系统故障，即表示系统的可用性变差。
- 关注的方面：
 - 如何检测系统故障、故障发生的频度、出现故障时的表现、允许系统有多长时间非正常运行、如何防止故障发生、发生故障后如何消除故障、等等。



典型NFR举例：可用性(availability)

■ 错误检测(Fault Detection)

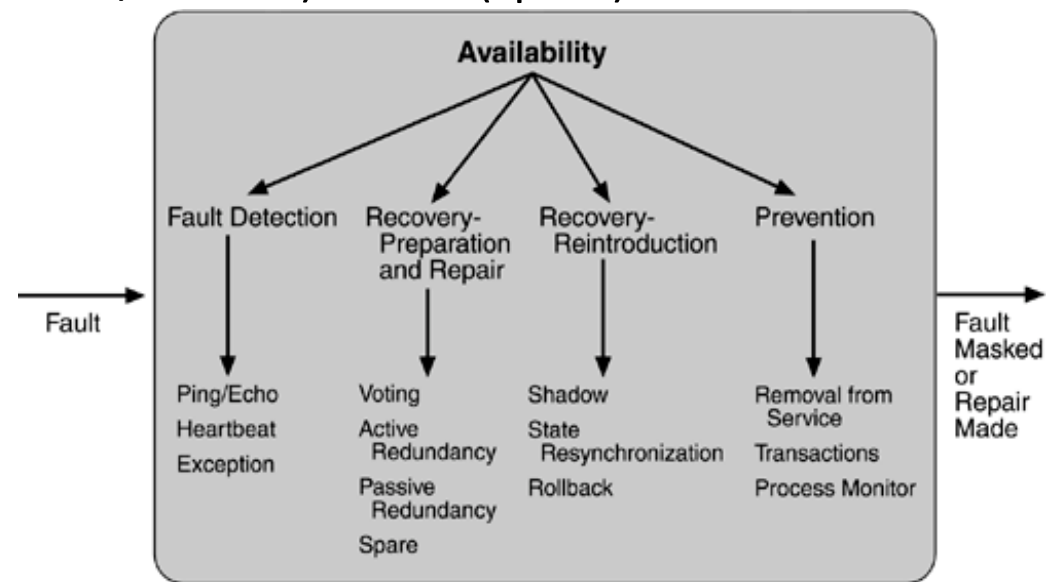
- 命令-响应机制(ping-echo)、心跳(heartbeat)机制、异常机制(exception);

■ 错误恢复(Recovery)

- 表决、主动冗余(热重启)、被动冗余(暖重启/双冗余/三冗余)、备件(spare);
- Shadow操作、状态再同步、检查点/回滚;

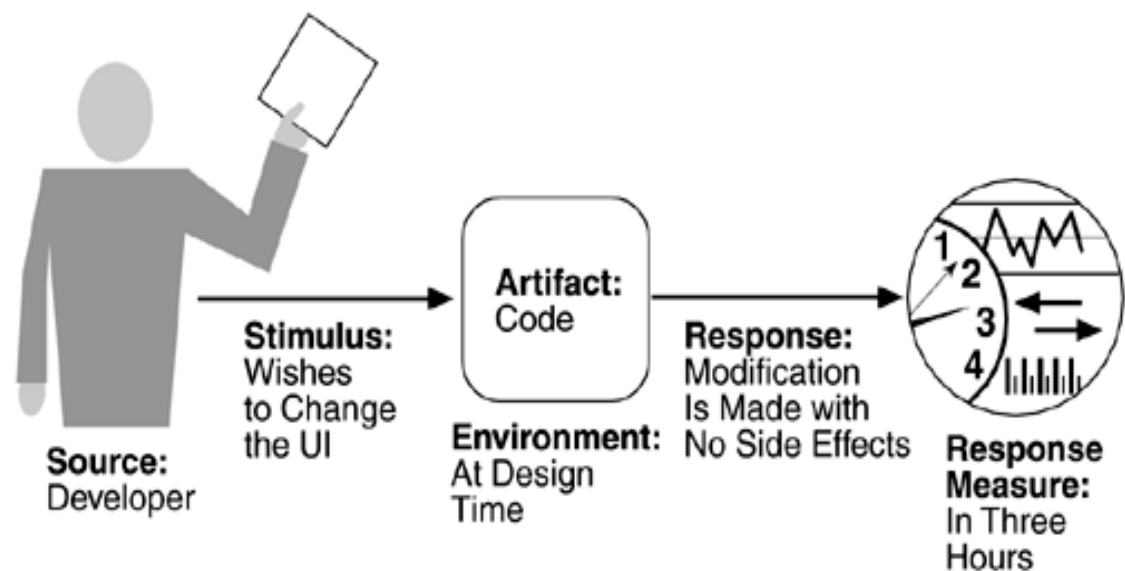
■ 错误预防(Prevention)

- 从服务中删除、事务、进程监视器。



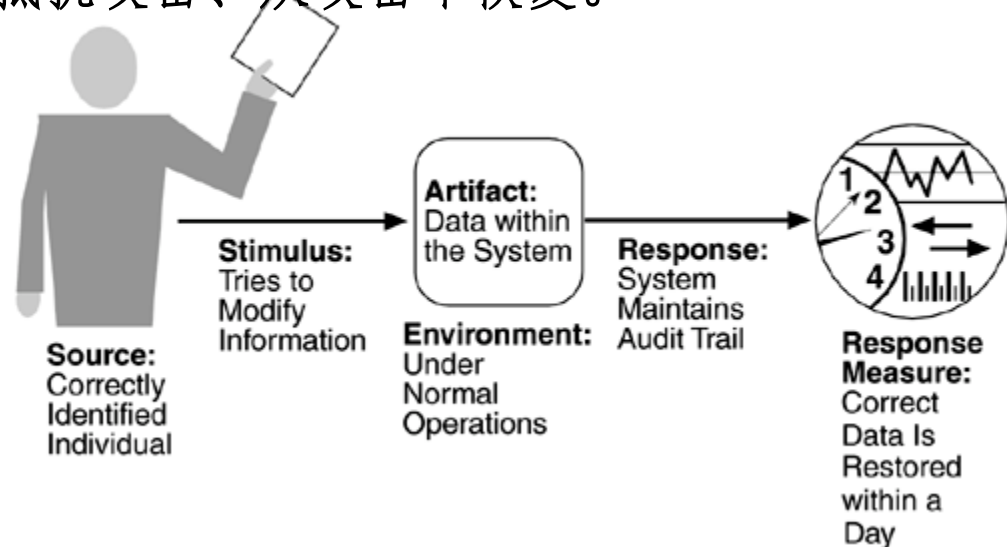
典型NFR举例：可修改性(modifiability)

- 可以修改什么——功能、平台(HW/OS/MW)、外部环境、质量属性、容量等；
- 修改的代价有多大？
- 修改的效率有多高？



典型NFR举例：安全性(security)

- 安全性：系统在向合法用户提供服务的同时，阻止非授权使用的能力
 - 未经授权试图访问服务或数据；
 - 试图修改数据或服务；
 - 试图使系统拒绝向合法用户提供服务；
- 关注点：检测攻击、抵抗攻击、从攻击中恢复。



软件工程学科发展

Wasserman, Anthony I., "Toward a discipline of software engineering", *IEEE Computer*, Vol. 13, No. 6, pp.23-31, 1996

改变软件工程的关键因素：

交付时间的重要性

计算技术在经济中的转变

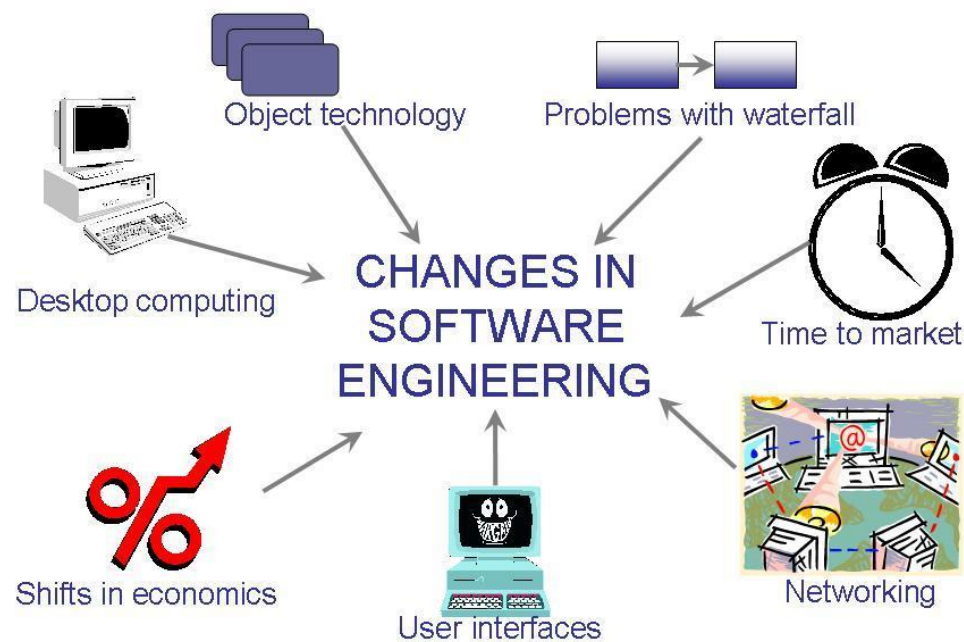
功能强大的桌面计算

互联网络

面向对象技术

图形用户界面

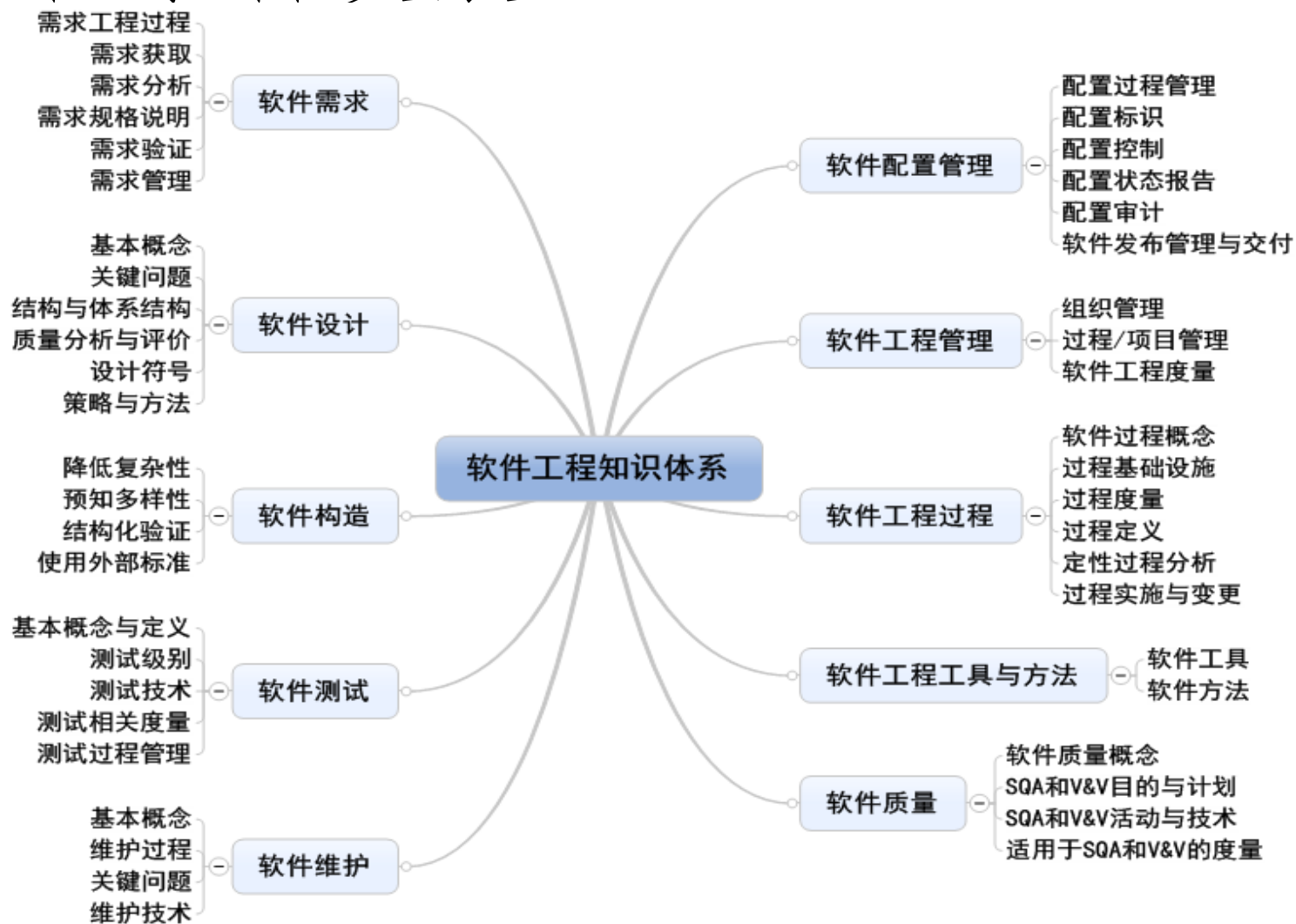
瀑布模型的不可预知性



软件工程学科发展



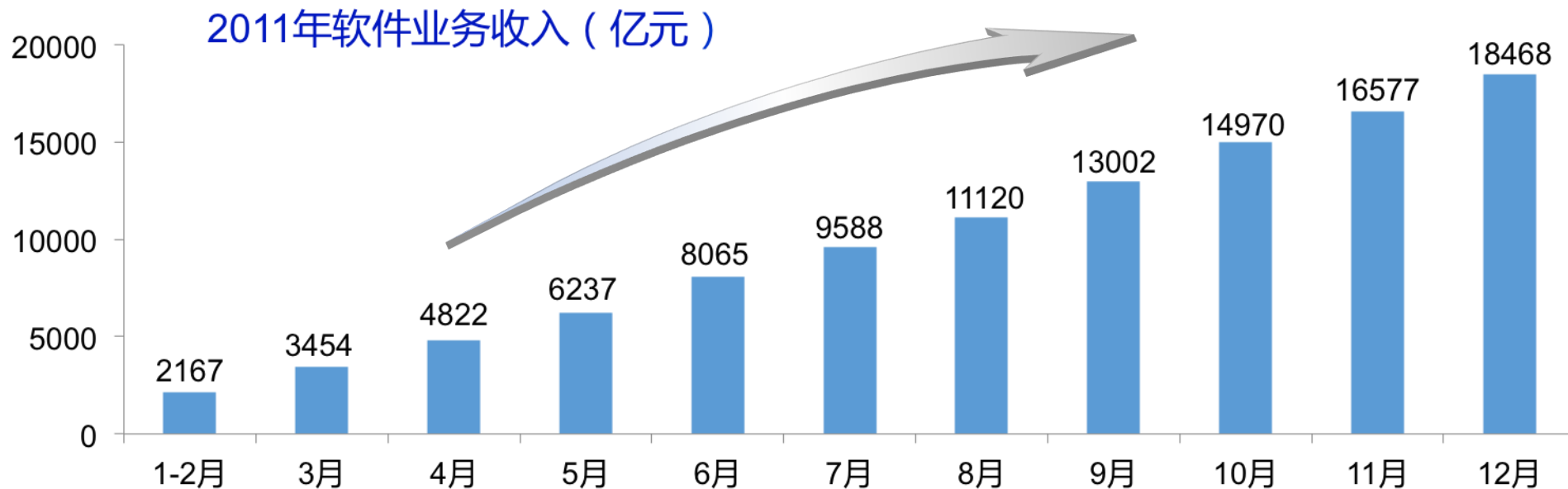
软件工程学科发展



中国软件产业的需要

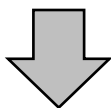


- 软件产业步入新的快速发展阶段，收入保持快速增长势头
- 软件行业结构调整步伐加快，骨干企业运行态势良好
- 基础软件产业化能力差，产业创新体系不健全，核心技术缺乏
- 人才结构矛盾突出，高层次、复合型、领军型人才依然缺乏



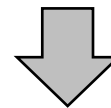
中国软件产业的需要

软件人才是软件企业最宝贵的财富，其工作特点是**创新**



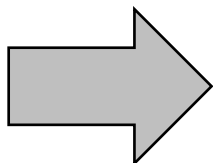
软件人才职业要求

- ✓ 良好的思想政治素质
- ✓ 恪守伦理和工程职业道德
- ✓ 掌握数学、自然科学和工程科学知识
- ✓ 具备解决复杂工程问题的能力
- ✓ 解决与工程有关经济、环境与社会问题
- ✓ 良好的人际交流与合作能力
- ✓ 具备国际视角和跨文化竞争与合作能力



软件人才素质要求

既要：细致、严谨、认真、甘于寂寞、乐于奉献、责任感强
又要：兴趣广泛、开阔思路、善于观察自然和社会、有创新创业精神
喻为：**动静结合**
还要：过硬的技术、良好的**文化底蕴**和**艺术素养**



面对软件技术的迅猛发展，软件人员要**不断学习**
才能保持**持续创新**，适应工作需要和软件技术发展的势态

什么是好的软件

Quality is never an accident; it is always the result of intelligent effort.

—— *John Ruskin*

Garvin 关于质量的各种不同视角：

- 先验论的观点：质量是通过经验来认识的，但无法精确定义。
- 用户的观点：质量是关于产品符合用户需求和期望的程度。
- 制造的观点：质量是与规格说明的一致性。
- 产品的观点：质量是与产品的内在特性相联系的。
- 基于价值的观点：质量取决于客户愿意支付的金额。

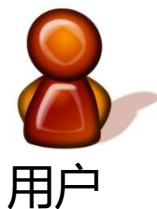
质量的三个方面：产品质量、过程质量、商业环境背景下的质量



什么是好的软件

功能质量

软件符合指定需求
软件几乎没有缺陷
软件性能正常
软件容易上手，操作方便



用户

软件质量



开发人员

结构质量

代码可测试性
代码可维护性
代码可读性
代码效率：高效管理资源
代码安全：可预防常见威胁



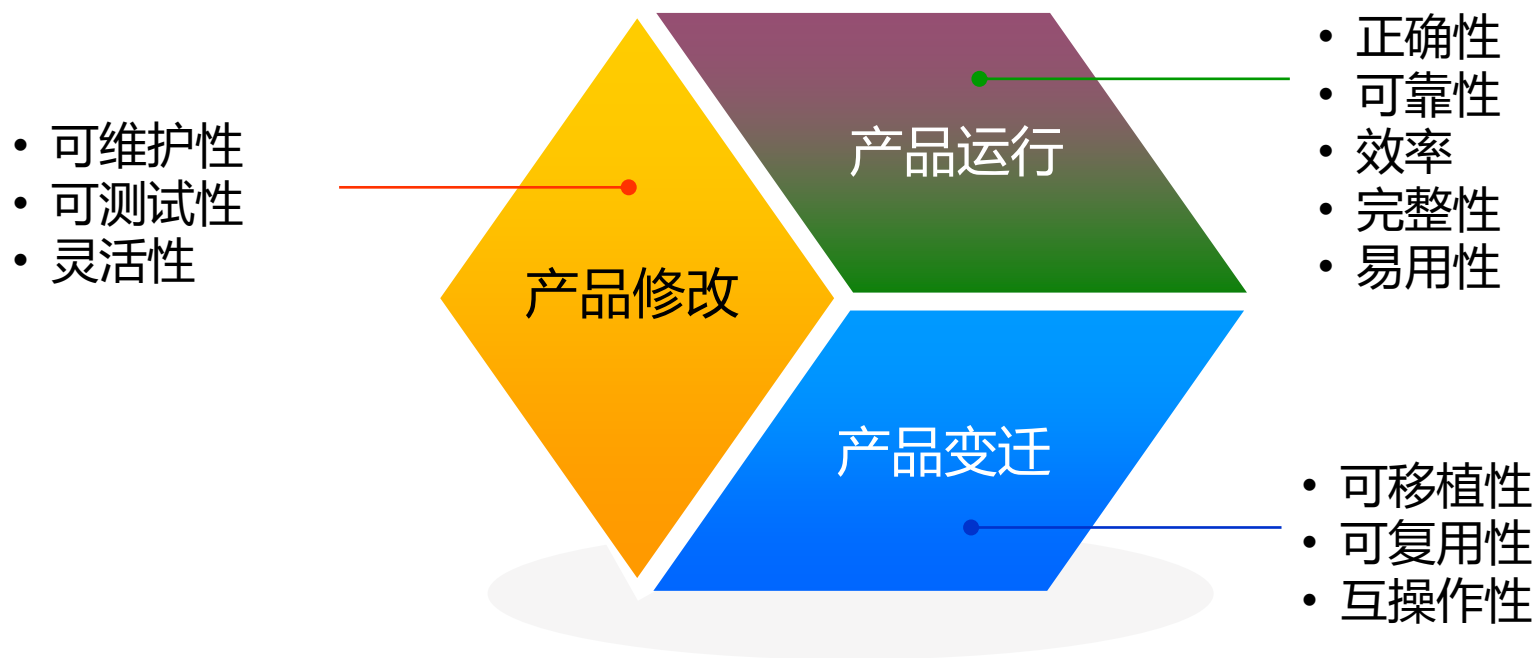
投资者

过程质量

软件按时交付
软件满足预算
可复用的开发过程，确保交付质量

McCall 质量模型

软件质量是许多质量属性的综合体现，各种质量属性反映了软件质量的不同方面。人们通过改善软件的各种质量属性，从而提高软件的整体质量。



McCall 质量模型

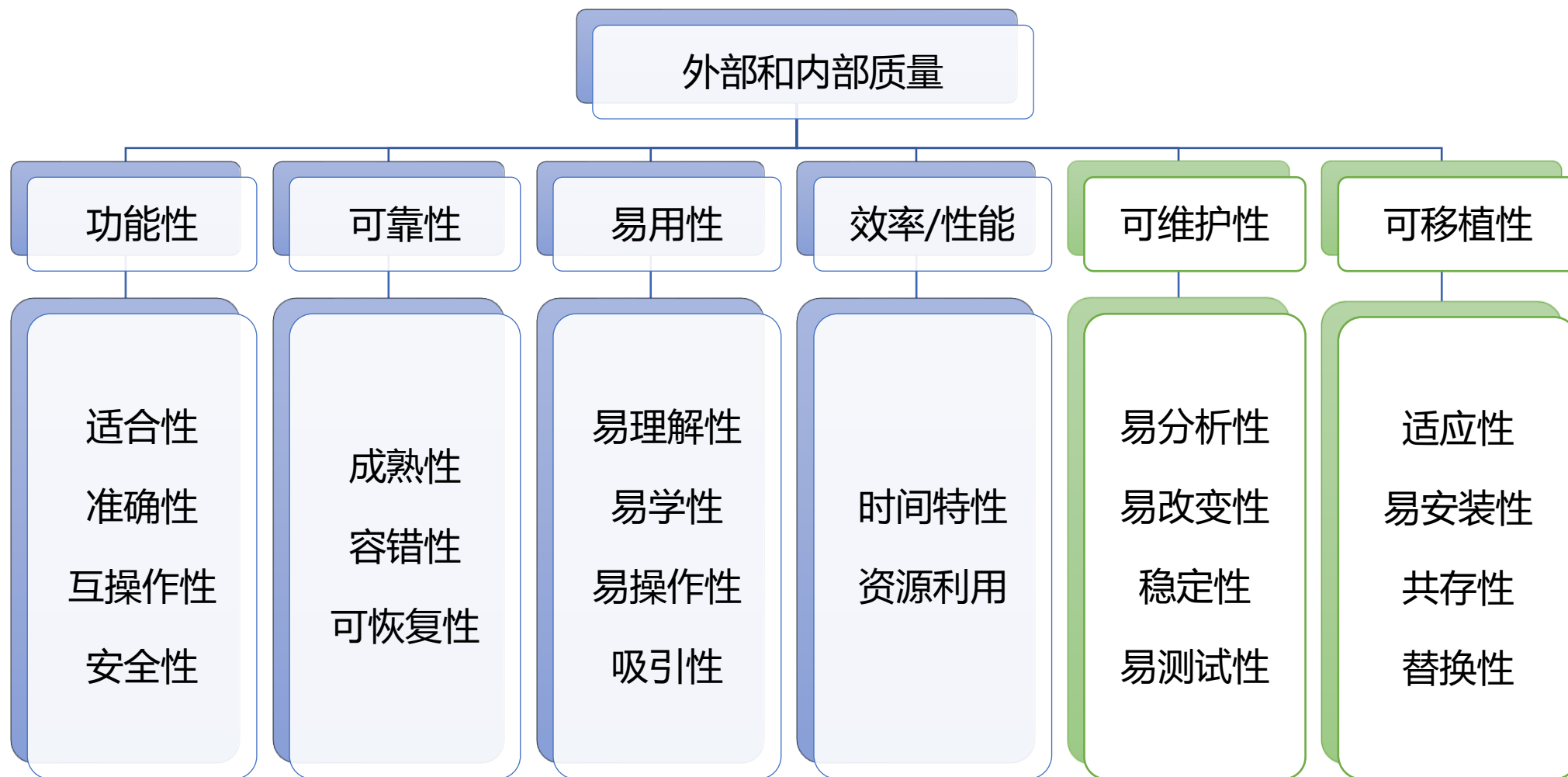
质量属性

- **正确性：** 软件满足需求规格说明和完成用户任务目标的程度。
- **可靠性：** 软件在规定时间和条件下无故障持续运行的程度。
- **效率：** 软件系统完成预定功能所需的计算机资源量。
- **完整性：** 对未授权人员访问软件或数据的可控程度。
- **易用性：** 用户学习、操作、准备输入和解释输出所需要的工作量。
- **可维护性：** 定位和修复软件中的一个错误所需的工作量。
- **可测试性：** 测试程序以确保它能完成预期功能所需的工作量。
- **灵活性：** 修改或改进一个已经投入运行的软件所需的工作量。
- **可移植性：** 将程序从一个硬件或软件环境移植到另一环境所需的工作量。
- **可复用性：** 程序可以再次用于另一个应用中的程度。
- **互操作性：** 将一个系统连接到另一个系统所需的工作量。

ISO9126 质量模型



ISO9126 质量模型



商业环境下的软件质量

讨论思考：

软件质量的重要性是无容置疑的，那么是不是质量越高就越好？
软件产品是否应该追求“零缺陷”？



举例：在航天器发射之前，只要发现任何异常，就会立即取消发射指令，直到异常被消除为止。前苏联甚至做得更过分，许多重大武器系统的负责人都签了生死状，系统研制成功则获得英雄勋章，失败则被枪毙。



许多互联网软件（例如新浪微博、百度导航等）在产品仍然存在一定缺陷的情况下就发布上线，之后再不断更新版本修复已有的缺陷。这种系统为什么不像航天系统一样，在发布前应修改所发现的任何缺陷？

商业环境下的软件质量

讨论思考：

软件质量的重要性是无容置疑的，那么是不是质量越高就越好？
软件产品是否应该追求“零缺陷”？

商业目标决定质量目标：

为了提高用户对产品的满意度，企业必须提高产品的质量，但是不可能为了追求完美的质量而不惜一切代价。当企业为提高质量所付出的代价超过收益时，这个产品就没有商业价值了。

企业必须权衡质量、效率和成本三个因素，产品质量太低了或者太高了，都不好。理想的质量目标不是“零缺陷”，而是恰好让广大用户满意，并且将提高质量所付出的代价控制在规定的预算之内。

软件工程=最佳实践

- 软件系统的复杂性、动态性使得：
 - 高深的软件理论在软件开发中变得无用武之地；
 - 即使应用理论方法来解决，得到的结果也往往难以与现实保持一致；
 - 因此，软件工程被看作一种实践的艺术：
 - 做过越多的软件项目，犯的错误就越少，积累的经验越多，随后作项目的成功率就越高；
 - 对新手来说，要通过多实践、多犯错来积累经验，也要多吸收他人失败的教训与成功的经验。
- 当你把所有的错误都犯过之后，你就是正确的了。