软件质量保证与测试

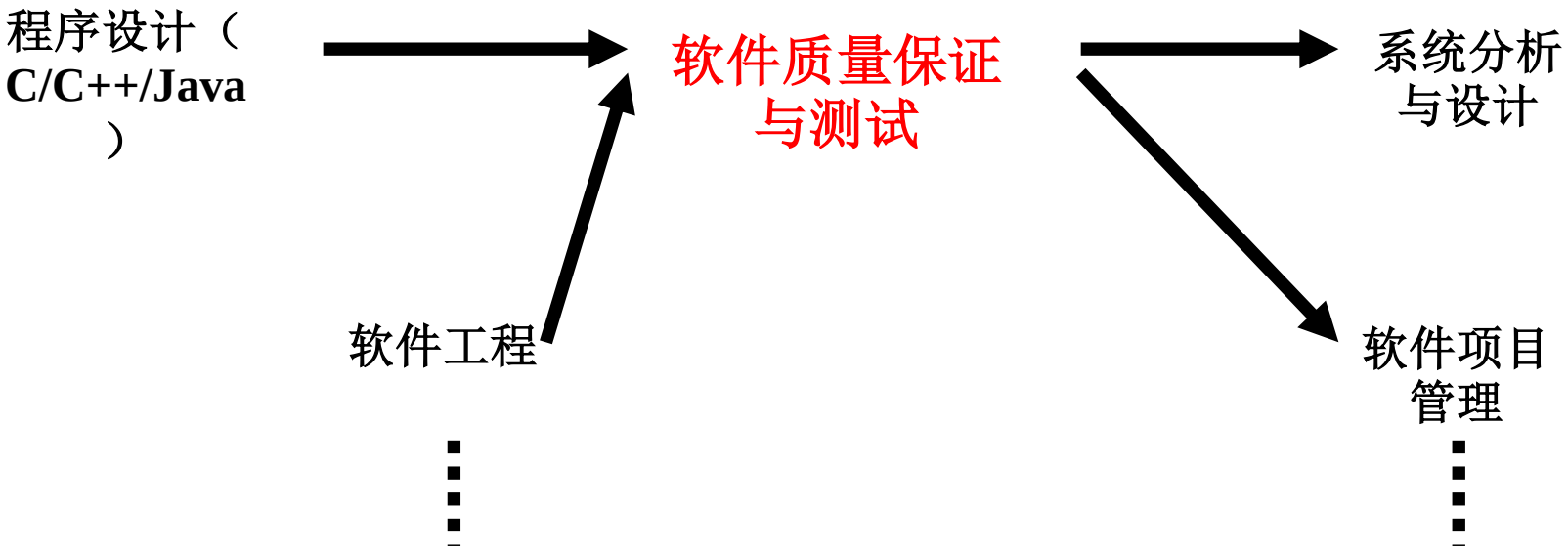
——软件测试、调试、重构、Design Patterns基础

User : testing_software@163.com
Password: testing

任课教师：侯鲲

blueblоodhk@163.com

课程体系



关于课程名称

软件质量保证——Software Quality Assurance (SQA)
一系列用于监控软件工程过程的方法，以保证软件的质量。

软件测试——Software Testing
一种用来促进被测软件的正确性、完整性、安全性、和品质的过程。

关于课程名称

软件质量保证与软件测试的关系

- 软件质量保证是建立达到良好软件质量的标准及开发过程，检查和评价当前软件开发过程，并设法达到防止软件出现错误的目标。
- 软件测试是软件质量保证的关键步骤。
- 软件质量保证与测试的关系就像“健康、长寿秘诀”与“到医院体检及看病”的关系。

关于课程名称

软件质量保证和测试的关系

QA + Testing = good software

质量保证+测试=好的软件

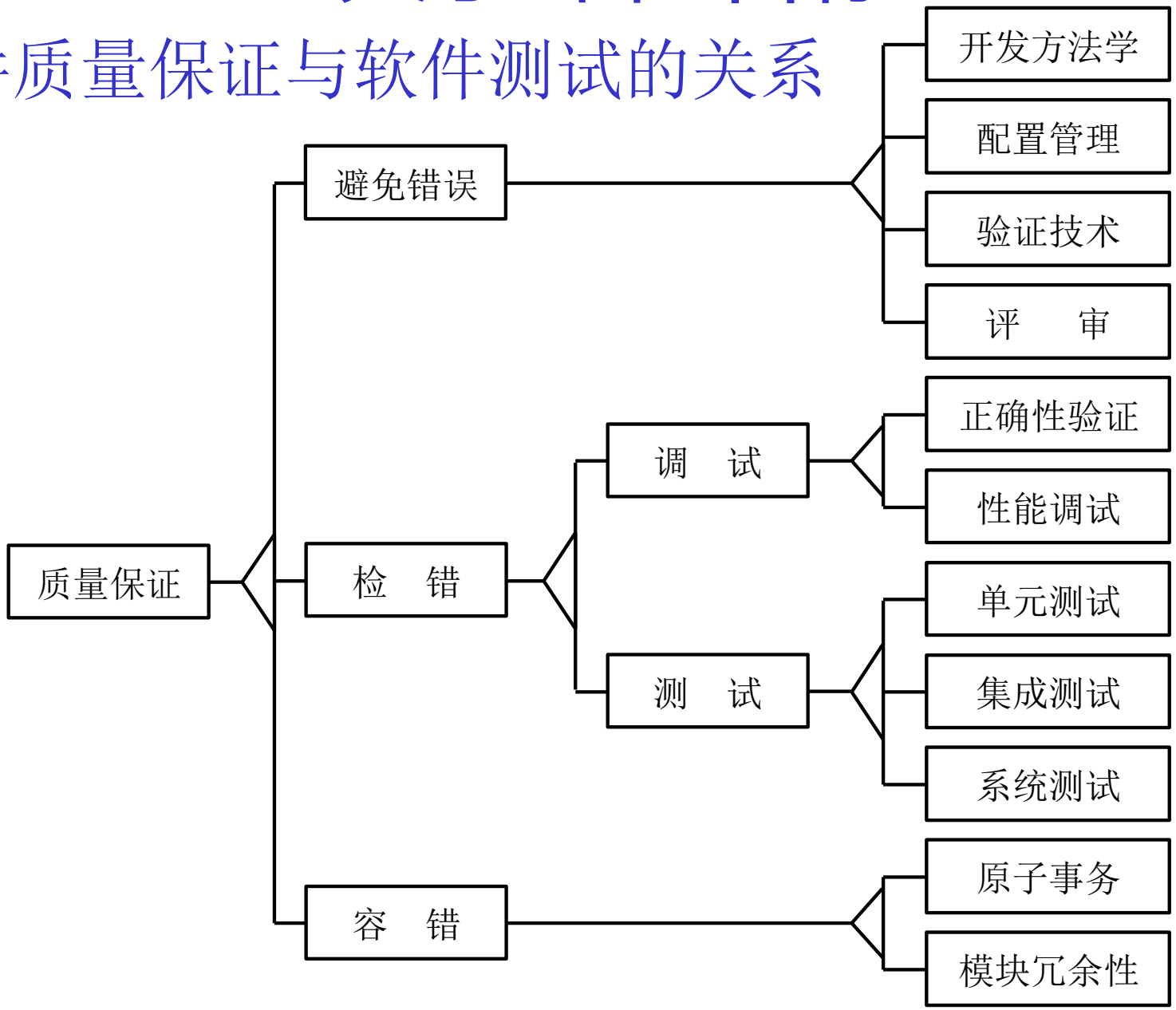
软件质量的特征，是每个人在某种条件之下需要它，每个人都觉得自己理解它，却又不愿意解释它。

关于课程名称

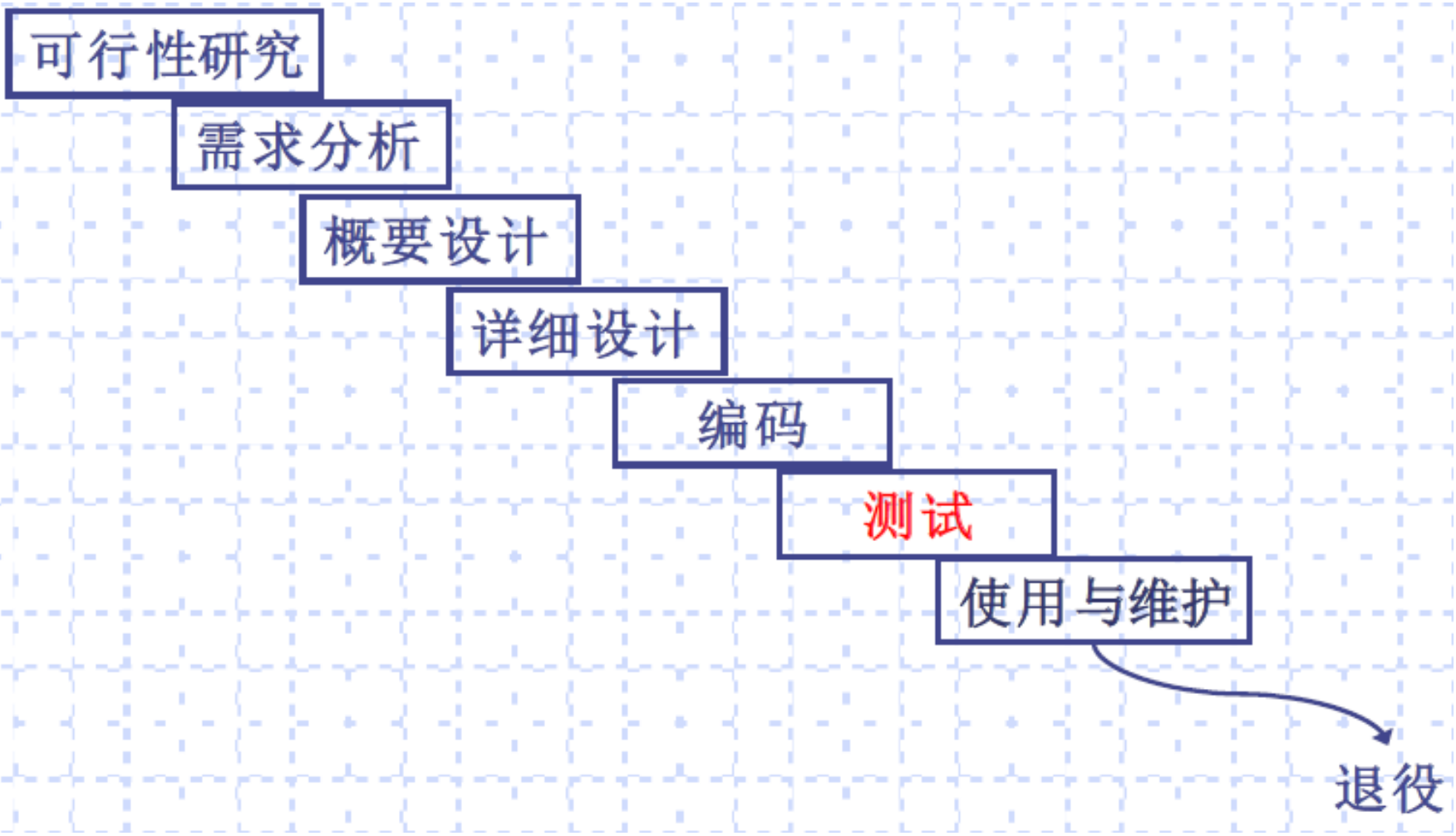
质量保证和质量控制

关于课程名称

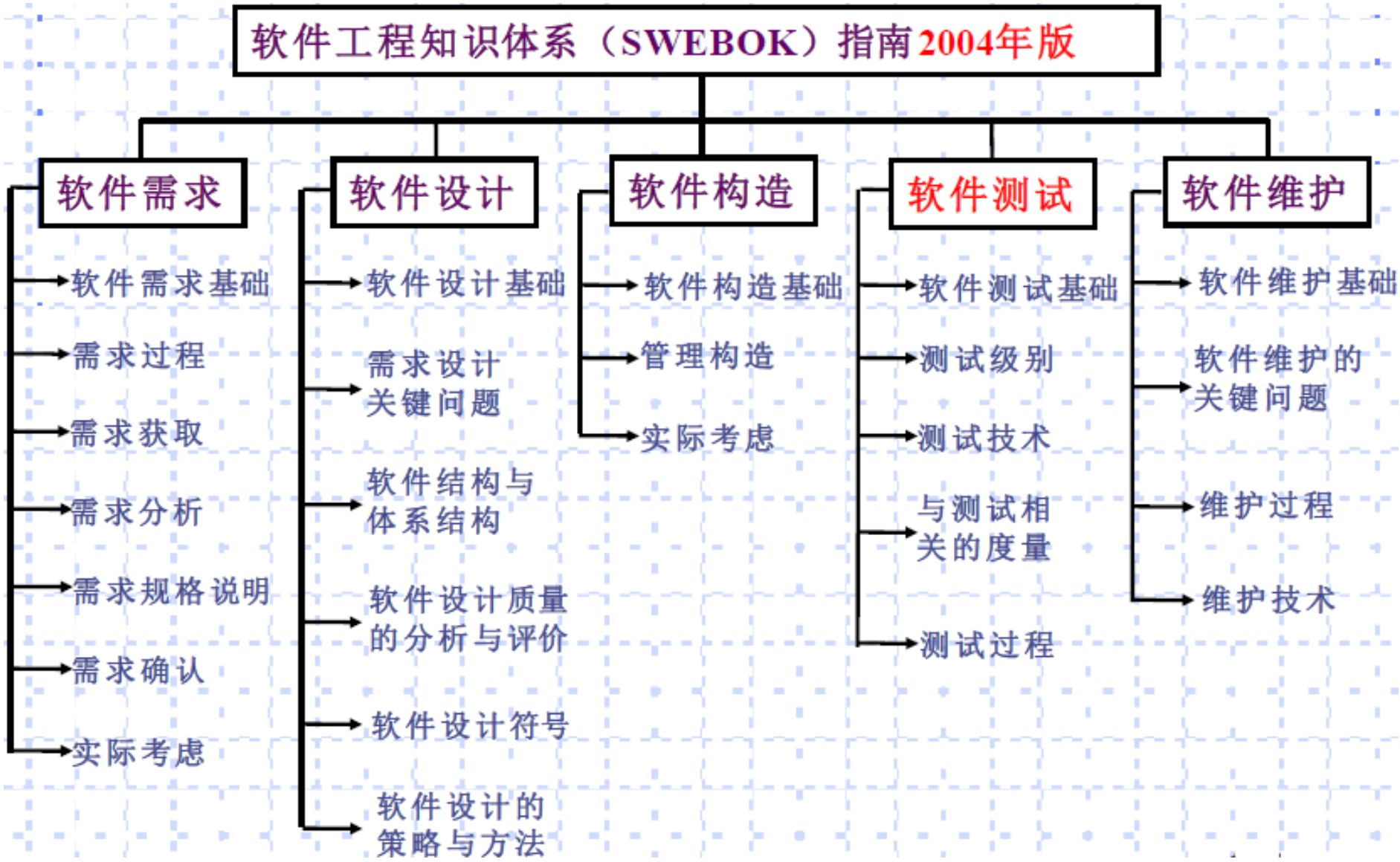
软件质量保证与软件测试的关系



软件测试在软件工程中的地位



软件测试在软件工程中的地位

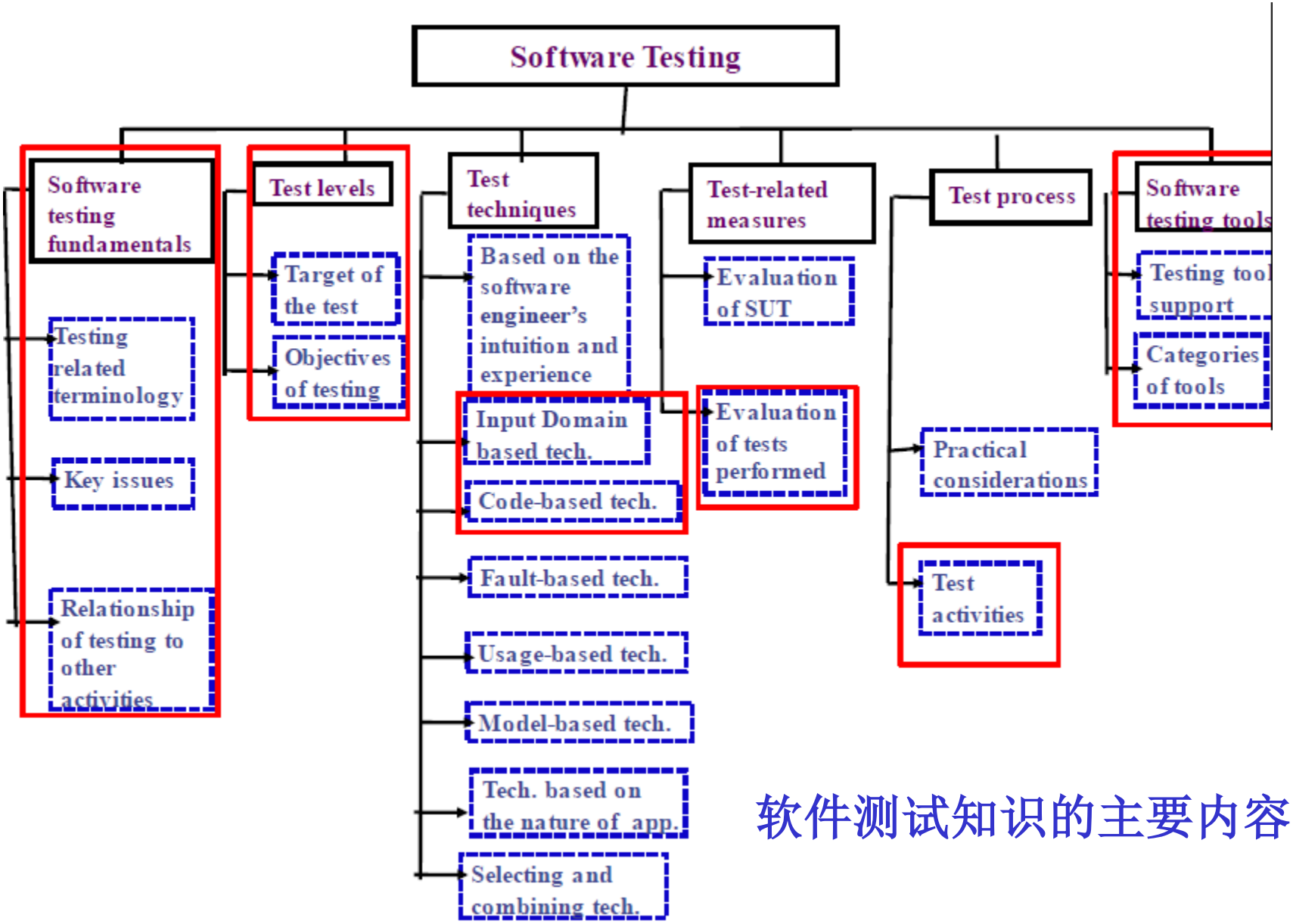


软件测试在软件工程中的地位

软件工程知识体系15个知识点（SWEBOK指南2014年版）

Software Requirements
Software Design
Software Construction
Software Testing
Software Maintenance
Software Configuration Management
Software Engineering Management
Software Engineering Process
Software Engineering Models and Methods
Software Quality
Software Engineering Professional Practice
Software Engineering Economics
Computing Foundations
Mathematical Foundations
Engineering Foundations

软件测试在软件工程中的地位



软件测试知识的主要内容

课程主要内容

- 软件质量保证
  - 软件质量管理
  - 软件质量标准和度量
  - 软件评审
  - 高质量编程*
- 软件测试基础
  - 黑盒测试与白盒测试*
  - 代码评审和静态分析
  - 各级别测试
  - 自动测试和测试工具*
  - 测试管理

课程目标

1.了解软件质量保证和测试的基本概念
2.了解软件测试的完整知识体系，包括测试的概念、原则，各种测试技术、测试级别、测试应用、测试工具、测试计划、测试过程和测试结果分析等
3.掌握软件测试的各种技术，静态、动态测试，白盒、黑盒测试等
4.了解（使用）常用的测试工具。

用“测试意识”知道编程和软件开发！

教学与考核方式

- 时间：周二 （5-6上课）

  周学时：2

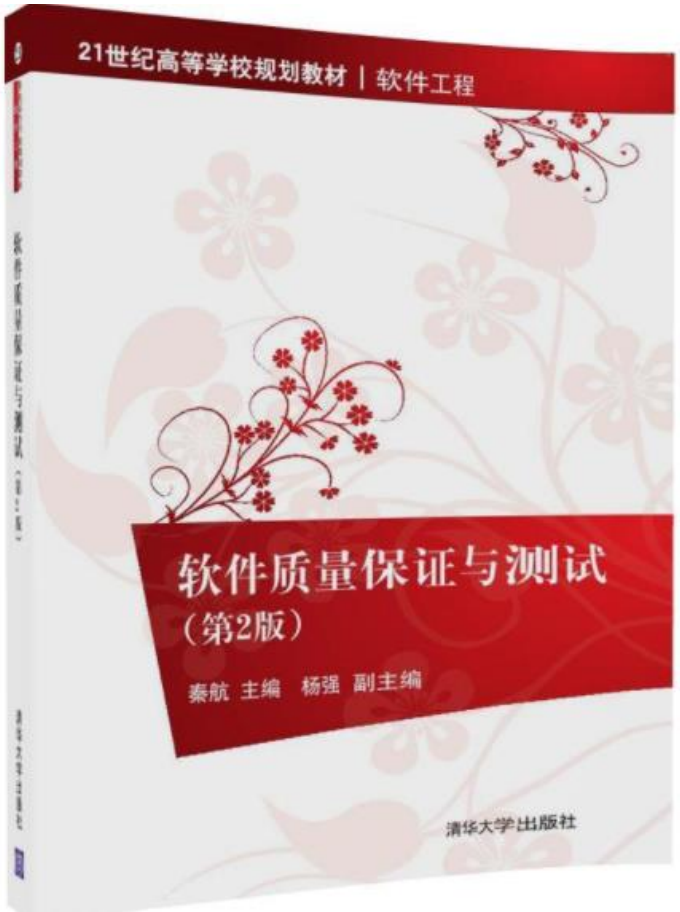
  总学时：<= 36

  方式：　课堂讲授，随堂练习

- 考核：平时成绩（作业、出勤、课程设计）占40%，期末考试占60%

  说明：作业+出勤20分，课程设计20分。

参考书

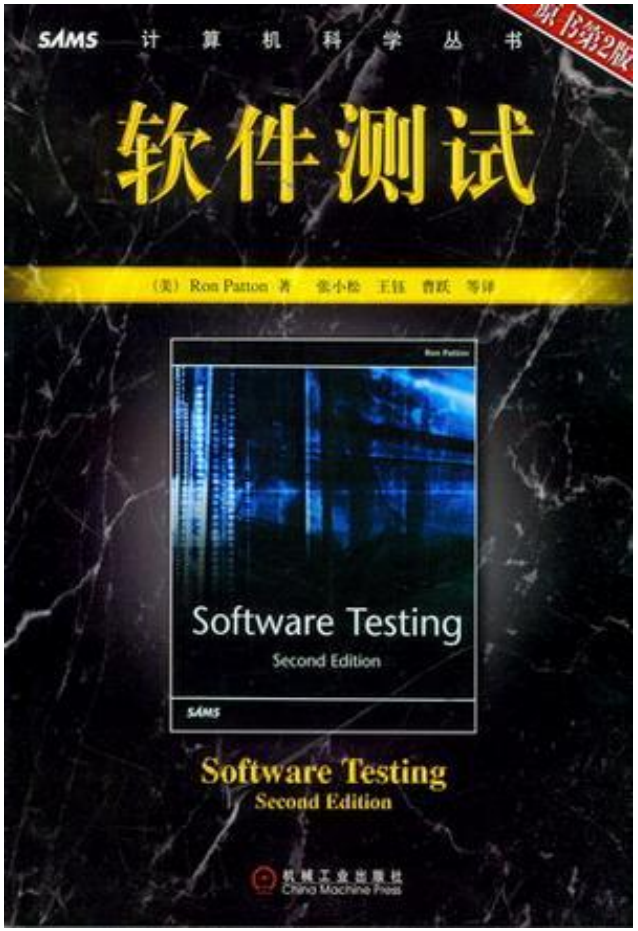


书名： **软件质量保证与测试(2nd)**

作者： 秦航

出版社：清华大学出版社

参考书



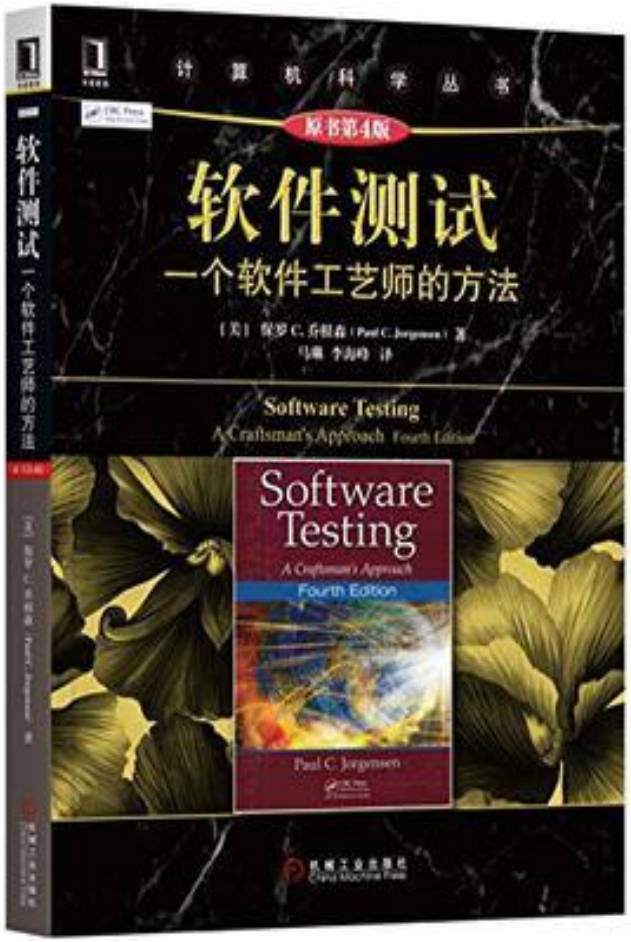
原书名： **Software Testing (2nd)**

原出版社： Sams

作者： （美）Ron Patton

出版社：机械工业出版社

参考书



软件测试-第4版

原书名： Software Testing: A Craftsman's Approach, Third Edition

作者： (美)Paul C. Jorgensen

出版社：机械工业出版社

本书曾是由ACM和IEEE计算机学会（www.swebok.org）联合编制的“软件工程知识体系”软件测试标准的主要参考文献之一。

参考书

软件测试的艺术-第3版

原书名： Software Testing: A Craftsman's Approach, Third Edition

作者： (美)Paul C. Jorgensen

出版社：机械工业出版社

本书曾是由ACM和IEEE计算机学会（www.swebok.org）联合编制的“软件工程知识体系”软件测试标准的主要参考文献之一。

参考书

其他参考书籍：

- **《软件测试与质量保证-理论与实践》，Kshirasagar Naik，电子工业出版社**
- **《软件质量和测试》，傅兵，清华大学出版社**
- **《软件测试实用教程——方法与实践(第2版)》，武剑洁，电子工业出版社**
- **《质量·软件·管理》，Gerald M. Weinberg，清华大学出版社**

引子

你是怎么测试你的软件的？
你发现过自己设计软件中的问题吗？

大家都应该编过程序，但是都测试过程序吗？
也许你在编程序的过程中遇到过问题，然后又找到了问题——这是测试吗？

一个有瑕疵的二分查找

```
public static int buggyBinarySearch(int[] a, int target) {
    int low = 0;
    int high = a.length - 1;
    while (low <= high) {
        int mid = (low + high) / 2;
        int midVal = a[mid];
        if (midVal < target)    low = mid + 1;
        else if (midVal > target)   high = mid - 1;
              else    return mid;
    }
    return -1;
}
```

一个有瑕疵的二分查找：修正

```
public static int buggyBinarySearch(int[] a, int target) {
    int low = 0;
    int high = a.length - 1;
    while (low <= high) {
        int mid = (low + high) >>>2;
        int midVal = a[mid];
        if (midVal < target)    low = mid + 1;
        else if (midVal > target)   high = mid - 1;
              else    return mid;
    }
    return -1;
}
```

```
int mid = low + ((high - low) / 2);
```

SQL注入（Web）

.Net平台下有如下代码（查询字符串）

String queryStr =
“**select *form Accounts where username =‘ ”** +
Request.QueryString[“username”] + “ **’and password= ‘ ”** +
Request.QueryString[“password”] + “ **’ ”** ;

目的是想让用户输入他们的信息并登陆：

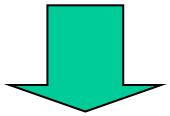
select *form Accounts where username=‘HouKun**’ and password=‘**Ilovethisgame**’**

但是…….

如果用户输入：
用户名：**haha**
密码：**‘; drop table Accounts**

第一条进行无用查找，第二条破坏表

另外，密码输入“ **’ ; delete from Accounts** ”



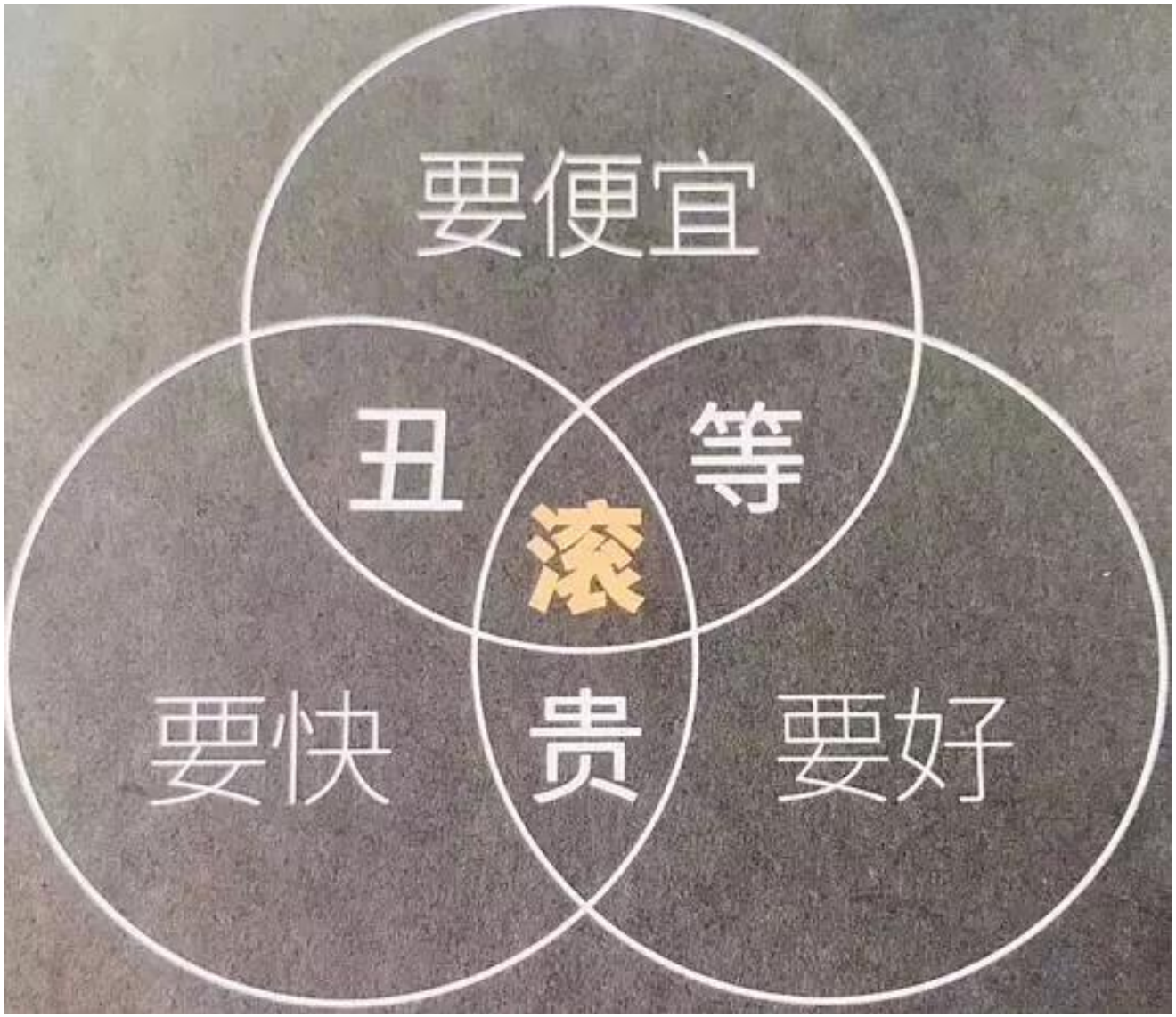
select *form Accounts where username=‘haha’ and password=‘ ’; drop table Accounts

注：
“ ‘ ” ：打开和关闭数据库字符串-字符串的开始和结束
“ ; ” ：SQL语句结束符
“ -- ”：SQL注释，忽略之后内容

软件测试的背景
——软件工程和软件过程

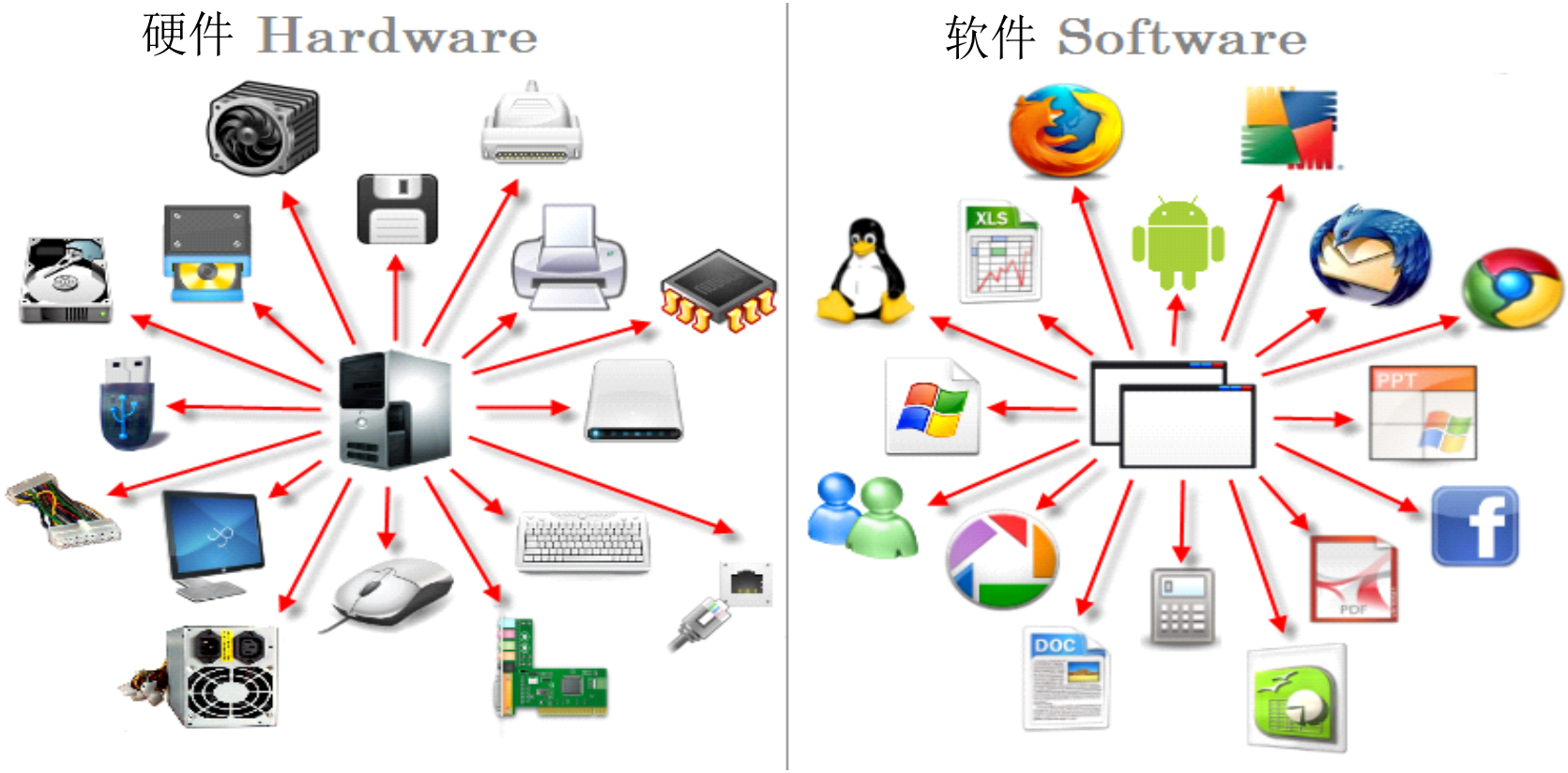
软件特征与软件工程

- 要理解软件的含义并全面地理解软件工程，要明确软件的特征，并据此知道软件与人类建造的其它事物之间的区别。
- IEEE定义对软件的定义如下：
  - 软件是计算机程序、规程以及可能的相关文档和运行计算机系统需要的数据。
  - 软件包含计算机程序、规程、文档和软件系统运行所必需的数据四个部分。


要便宜
丑
等
滚
要快
贵
要好

计算机硬件vs计算机软件

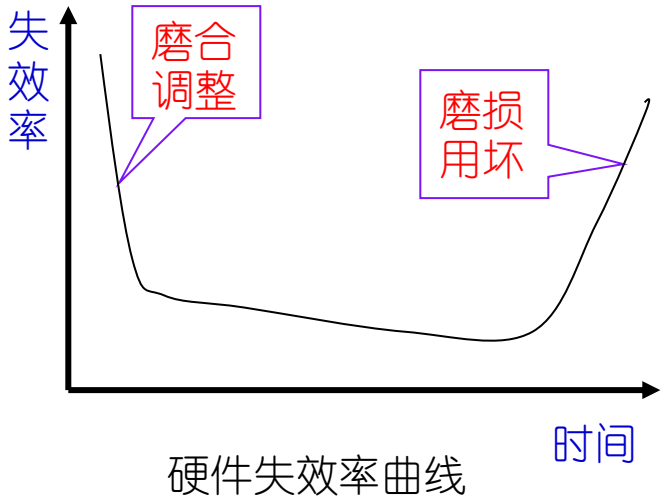
- 软件是逻辑产品，而不是物理产品，所以，软件具有和硬件完全不同的特征



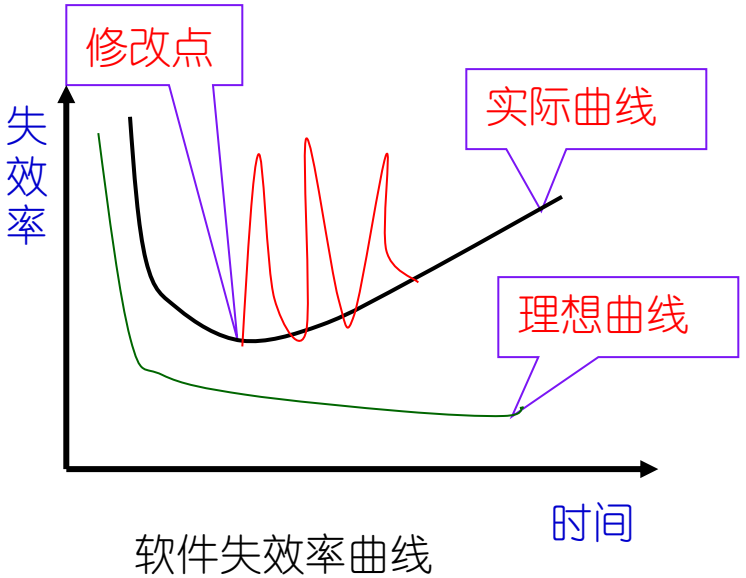
软件的概念与特点

- 软件是一种逻辑实体，而不是具体的物理实体
- 软件的生产与硬件不同
  - 在软件的运行和使用期间，没有硬件那样的机械磨损，老化问题

“浴缸曲线”

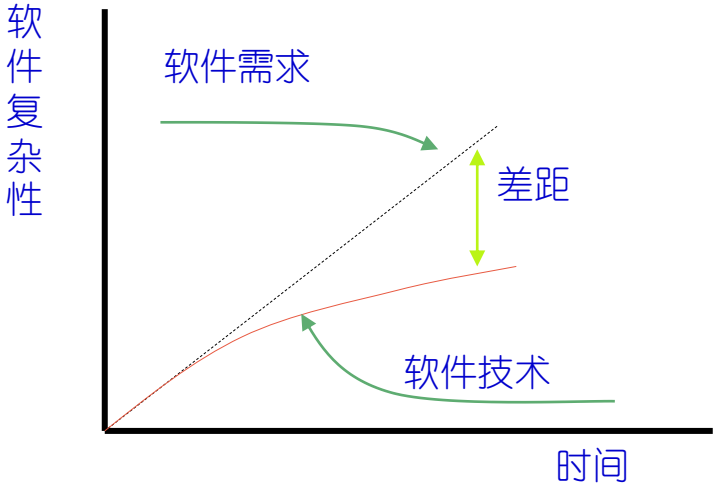


硬件失效率曲线

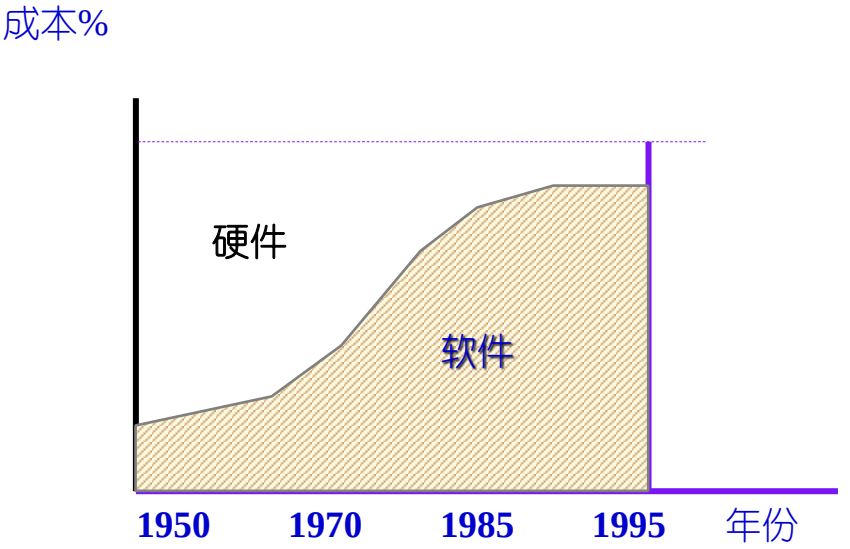


软件失效率曲线

软件的概念与特点



软件技术的发展落后于需求



硬、软件成本比例的变化

软件的成本相当昂贵

软件危机

软件规模

类别	参加人数	研制期限	产品规模(源代码行数)
微型	**1**	**1-4周**	**约500行**
小型	**1**	**1-6周**	**约2000行**
中型	**2-5**	**1-2年**	**5000-50000行**
大型	**5-20**	**2-3年**	**5万-10万行**
甚大型	**100-1000**	**4-5年**	**100万行**
极大型	**2000-5000**	**5-10年**	**1000万行**

Windows95有1000万行代码
Windows2000有5000万行代码
项目经理约250人
开发人员约1700人
测试人员约3200人

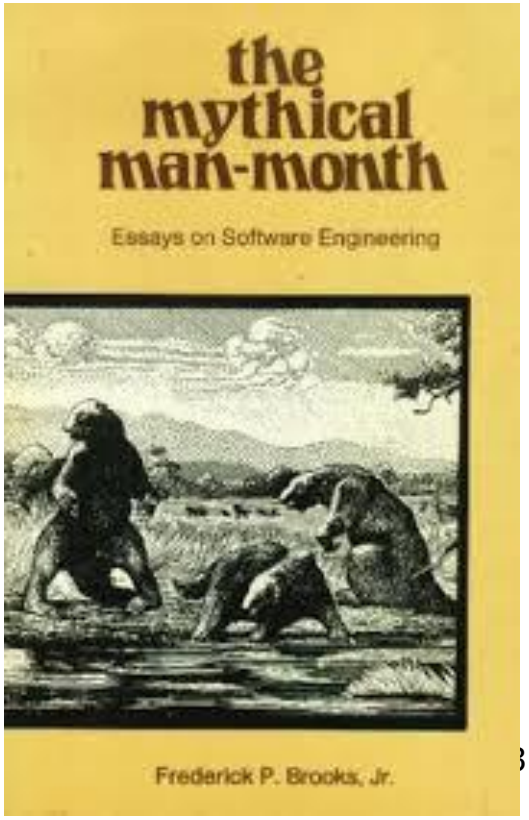
大型软件的命运-IBM System/360

- 1964年4月7日，IBM推出了System/360系列大型主机，这个划时代的创新，改变了商业界、科学界、政府、IT界本身

因为IBM System/360的工作，佛瑞德·布鲁克斯（Frederick P. Brooks）于1999年获得图灵奖

大型软件的命运-IBM System/360

- Frederick P. Brooks把在IBM公司任System计算机系列以及其庞大的软件系统OS项目经理时的实践经验，总结成《人月神话》一书，



失败-大型软件项目的宿命？！

没有别的场景比巨兽们在焦油坑中垂死挣扎的场面更令人震撼

软件危机

软件危机的原因

- 客观：软件本身特点
- 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

软件危机

克服软件危机的途径

- 消除错误的概念和做法
- 推广使用成功的开发技术和方法
- 使用软件工具和***软件工程***支持环境
- 加强软件管理

软件工程的概念及范畴

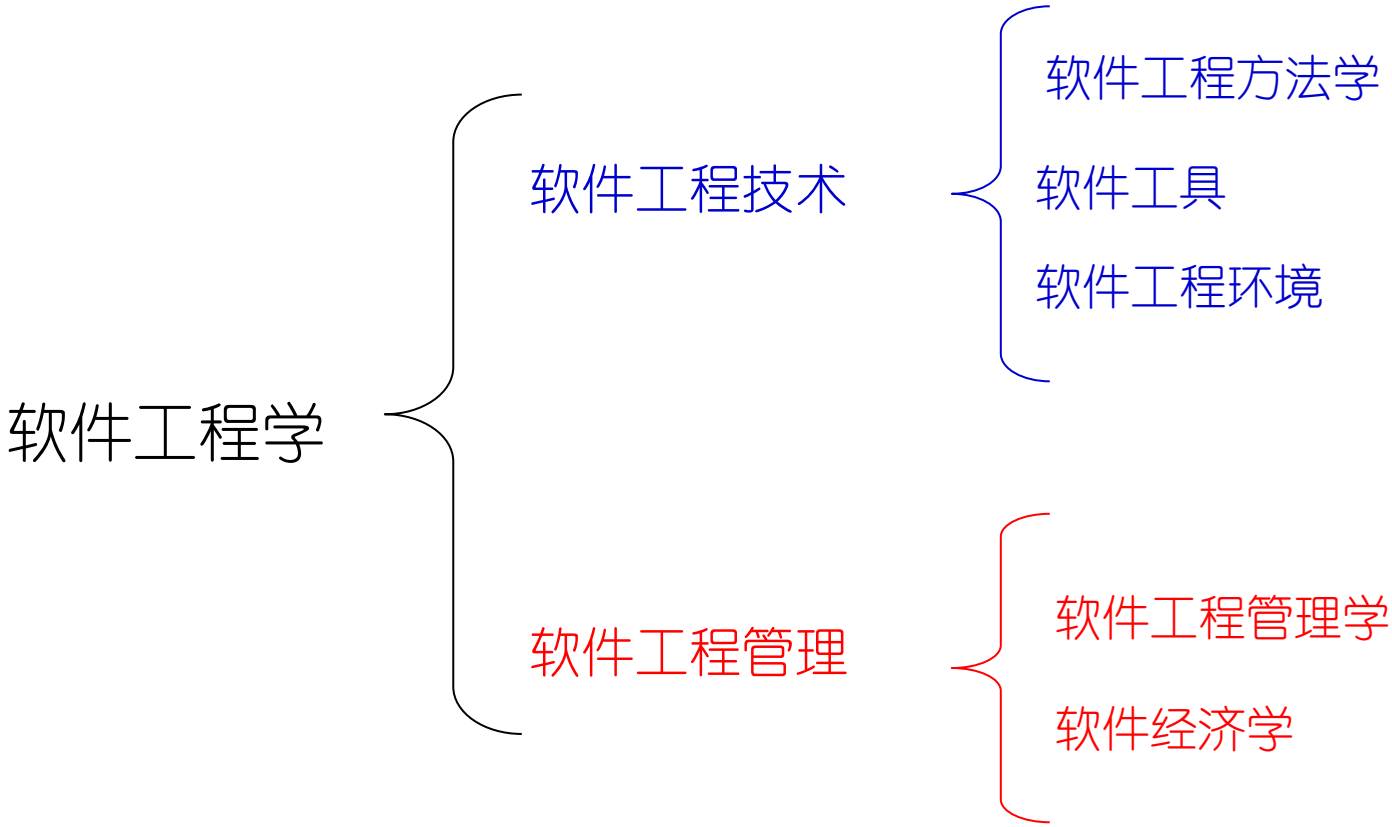
“软件工程”　----Software Engineering

于1968年 NATO 组织在德国召开的一次会议上提出

是把软件当作一种工业产品，要求 “采用工程化的　原理与方法对软件进行计划、开发和维护”。

软件工程的概念及范畴

软件工程学的范畴



软件工程的概念及范畴

软件工程方法学包含3个要素

- 方法

  完成软件开发的各项任务的技术方法。

- 工具

  工具是为运用方法而提供的自动的或半自动的软件工程支撑环境。

- 过程（软件过程）

  为了获得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。

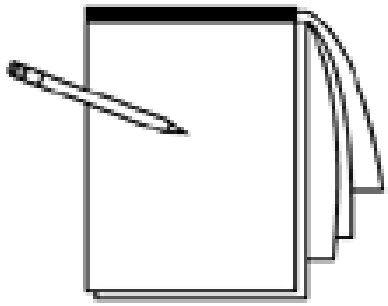
软件过程（软件生命周期）

- **软件过程**：是为了获得高质量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。
- 通常使用**生命周期模型**简洁地描述软件过程。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**。
- 软件过程本身就是软件
  软件过程是一种被由人构成的虚拟机执行的软件。

“大爆炸”和“编写边改”

Big Bang

OR



Typically informal
Product Specification

Code, Fix,
Repeat Until?

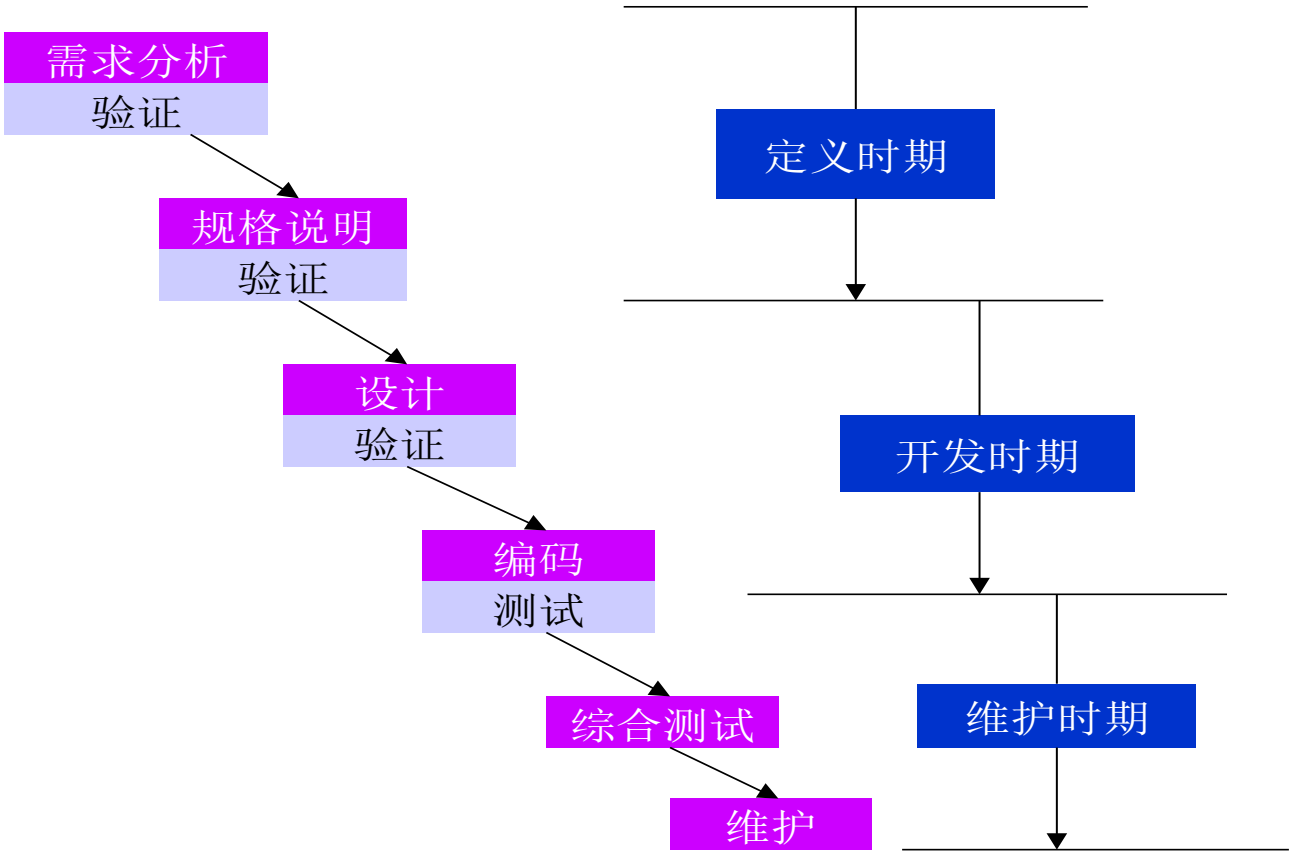
软件开发过程

基本的计划性软件过程（软件生命周期）

- 分析：（需求获取、分析/软件分析）
- 设计：
- 编码：（实现）
- 测试：
- 部署：（发布）
- 维护：

瀑布模型

瀑布模型 — 传统的瀑布模型



基于活动分解项目

瀑布模型

瀑布模型基本思想

- 将软件开发过程划分为分析、设计、编码、测试等阶段。
- 软件开发要遵循过程规律，按次序进行。
- 每个阶段均有里程碑和提交物。
- 工作以线性方式进行，上一阶段的输出是下一阶段的输入。

瀑布模型

瀑布模型的优点

- 简单、易懂、易用。
- 为项目提供了按阶段划分的检查点，项目管理比较规范。
- 每个阶段必须提供文档，而且要求每个阶段的所有产品必须进行正式、严格的技术审查。

瀑布模型

运用瀑布模型遇到的问题

- 实际的项目很少遵守瀑布模型提出的顺序。
- 客户通常难以清楚地描述所有的需求。
- 客户必须要有耐心，只有在项目接近尾声的时候，他们才能得到可执行的程序。

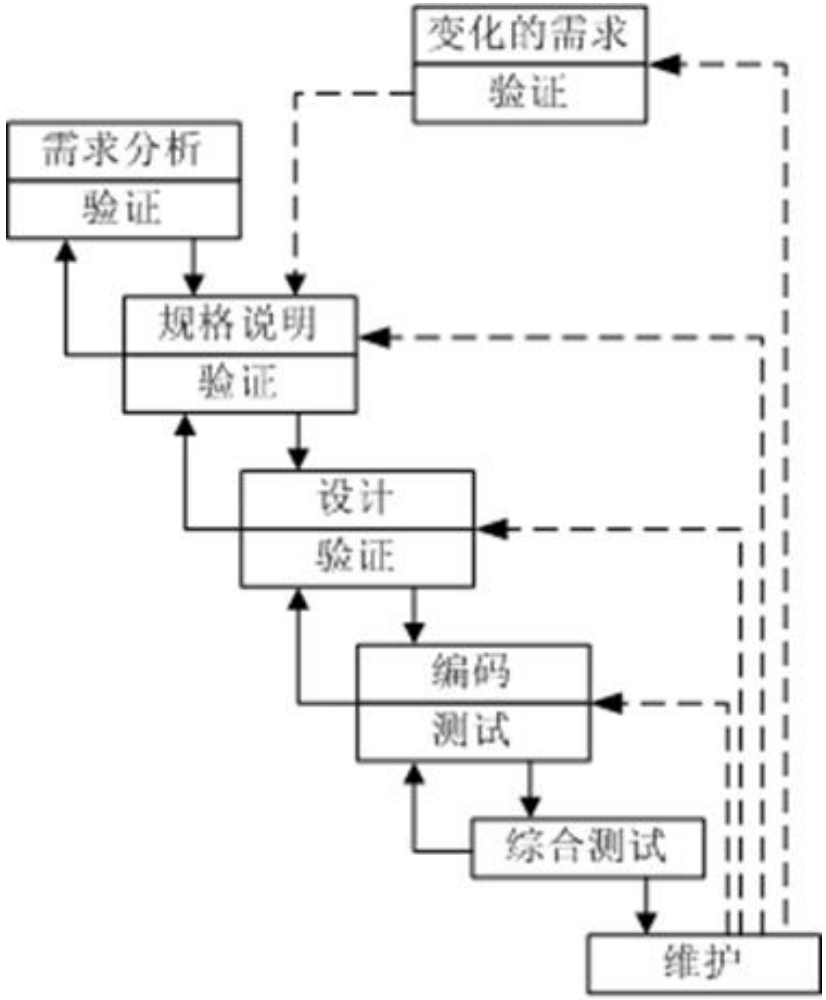
瀑布模型

瀑布模型的适用场合

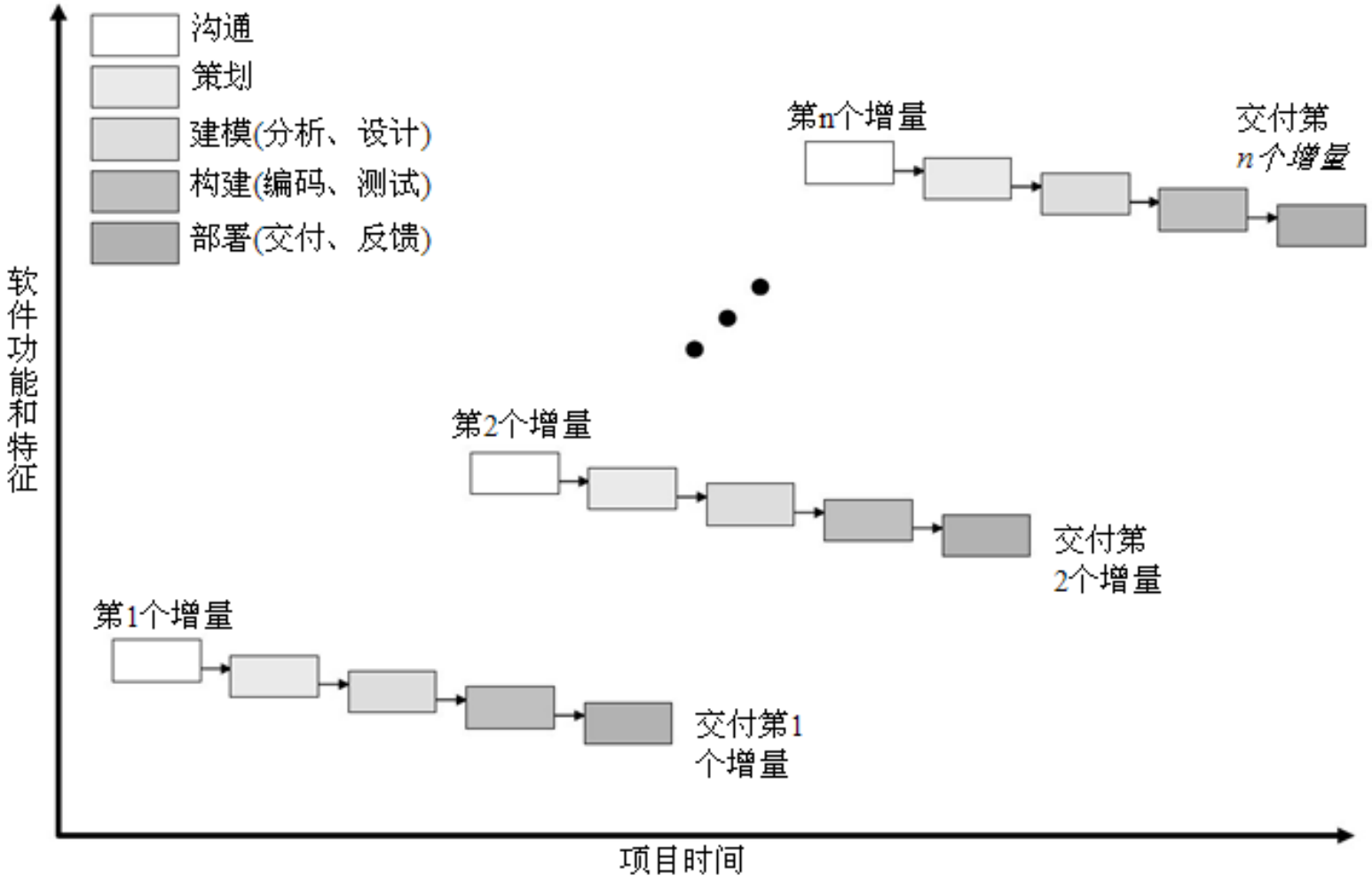
- 需求相当稳定，客户需求被全面的了解风险管理。
- 开发团队对于这一应用领域非常熟悉。
- 外部环境的不可控因素很少。
- 小型清晰的项目或长周期的项目。

瀑布模型

瀑布模型——实际的瀑布模型：



增量模型


沟通
策划
建模(分析、设计)
构建(编码、测试)
部署(交付、反馈)
软件功能和特征
项目时间
第1个增量
交付第1个增量
第2个增量
交付第2个增量
第n个增量
交付第n个增量

增量模型

增量模型的使用方法

- 软件被作为一系列的增量来进行开发，每一个增量都提交一个可以操作的产品，可供用户评估。
- 第一个增量往往是核心产品：满足了基本的需求，但是缺少附加的特性。
- 客户使用上一个增量的提交物并进行自己评价，制定下一个增量计划，说明需要增加的特性和功能。
- 重复上述过程，直到最终产品产生为止。

增量模型

增量模型应用举例

- 应用举例：开发一个类似于Word的字处理软件。
  - 增量1：提供基本的文件管理、编辑和文档生成功能。
  - 增量2：提供高级的文档编辑功能。
  - 增量3：实现拼写和语法检查功能。
  - 增量4：完成高级的页面排版功能。

增量模型

增量模型的优点

- **提高对用户需求的响应**：用户看到可操作的早期版本后会提出一些建议和需求，可以在后续增量中调整。
- **人员分配灵活**：如果找不到足够的开发人员，可采用增量模型，早期的增量由少量人员实现，如果客户反响较好，则在下一个增量中投入更多的人力。
- **可规避技术风险**：不确定的功能放在后面开发。

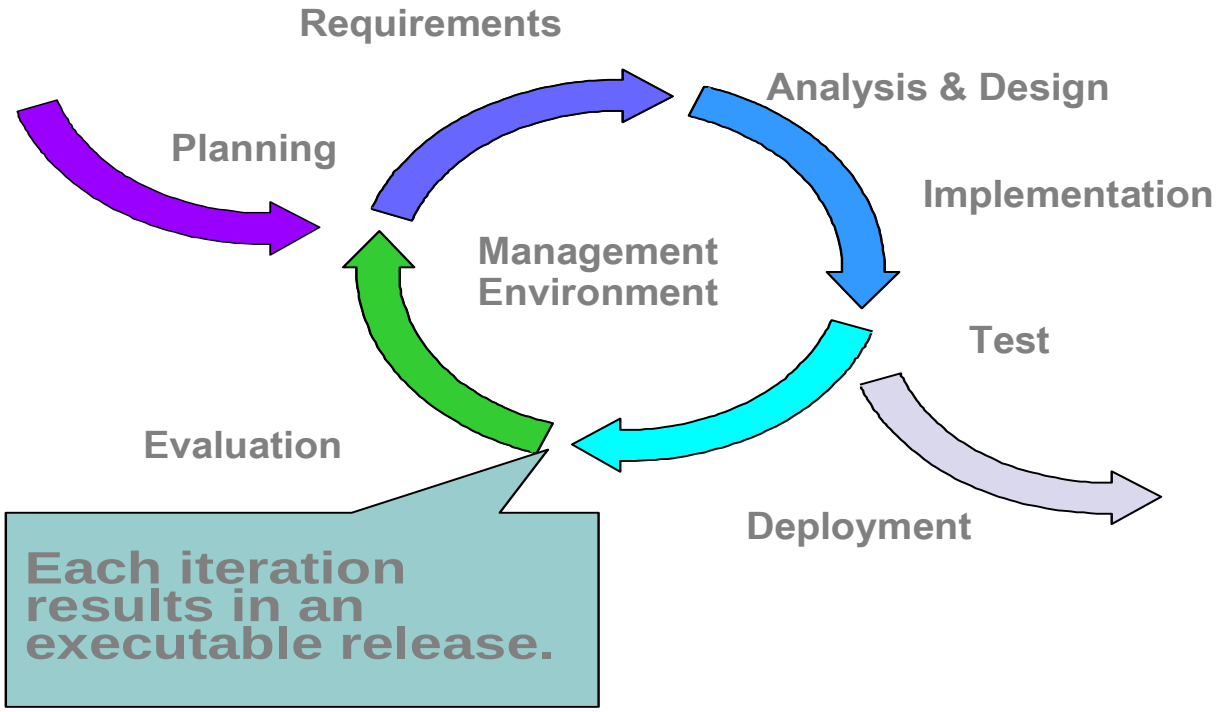
增量模型

增量模型存在的问题

- **每个附加的增量并入现有的软件时，必须不破坏原来已构造好的东西。**
- **加入新增量时应简单、方便 ——该类软件的体系结构应当是开放的。**
- **仍然无法处理需求发生变更的情况。**
- **管理人员须有足够的技术能力来协调好各增量之间的关系。**

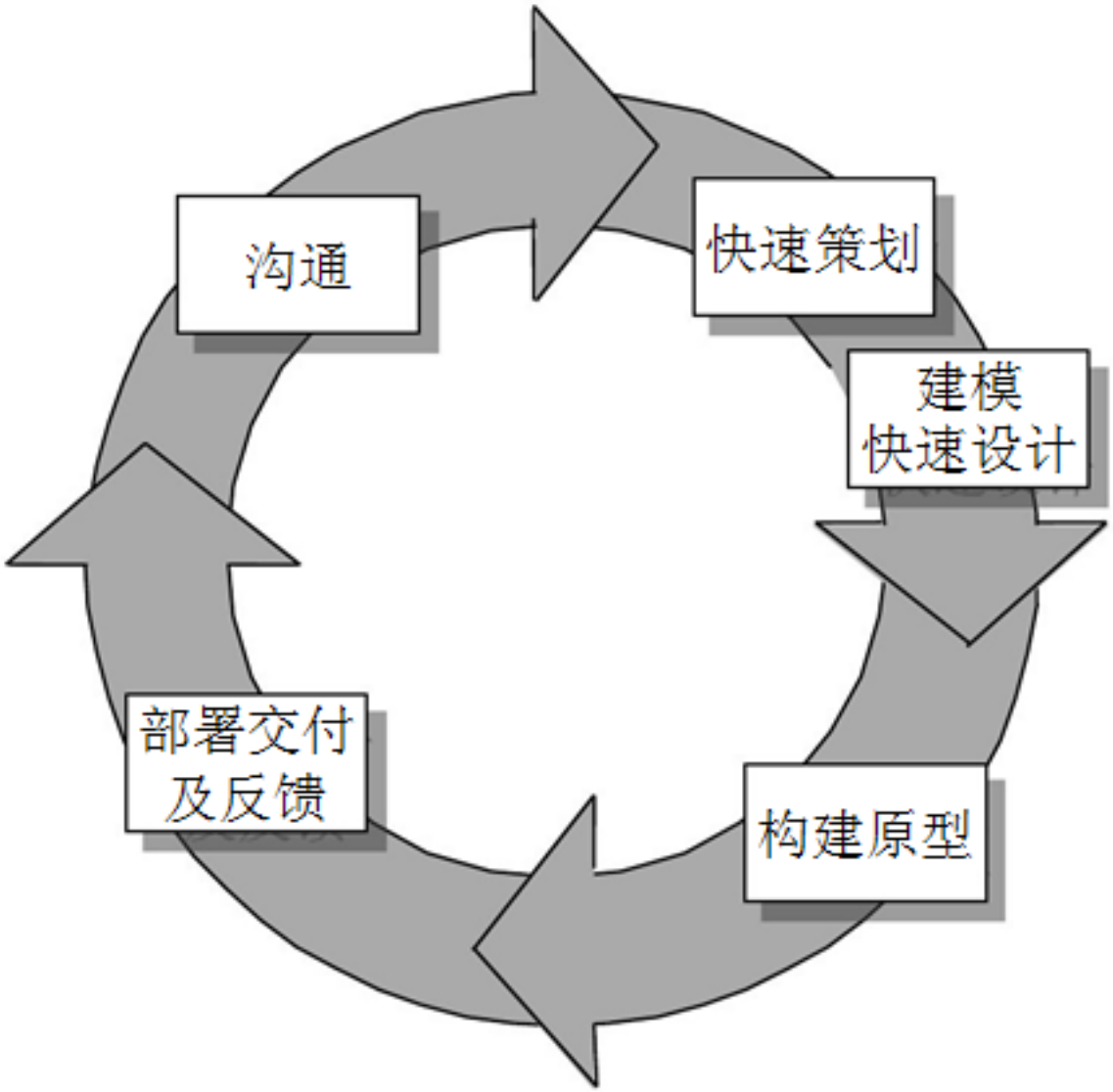
迭代模型

迭代模型(Cont.) ——一般形式



（快速）原型模型

一种迭代模型



（快速）原型模型

原型模型的使用方法

- 使用方法一：
  - 步骤1：开发人员与客户双方通过沟通，明确已知的需求，并大致勾画出以后再进一步定义的东西。
  - 步骤 2：迅速策划一个原型开发迭代并进行建模，主要集中于那些最终用户
  - 所能够看到的方面，如人机接口布局或者输出显示格式等；
  - 步骤3：快速设计产生原型，对原型进行部署，由客户和用户进行评价；
  - 步骤4：根据反馈，抛弃掉不合适的部分，进一步细化需求并调整原型；
  - 步骤5：原型系统不断调整以逼近用户需求。
- 使用方法二：把原型系统作为需求分析的工具，明确需求后，原型系统被抛弃。

（快速）原型模型

原型开发应用举例

- 应用举例：开发一个教务管理系统。
  - 第一次迭代：完成基本的学籍管理、选课和成绩管理功能。（6周）
  - 客户反馈：基本满意，但是对大数据量运行速度慢效率，不需要学生自己维护学籍的功能等。
  - 第二次迭代：修改细节，提高成绩统计和报表执行效率（2周）。
  - 客户反馈：需要严格的权限控制，报表打印格式不符合要求。
  - 第三次迭代：完善打印和权限控制功能。（2周）
  - 客户反馈：可以进行正式应用验证。

（快速）原型模型

原型开发的优点

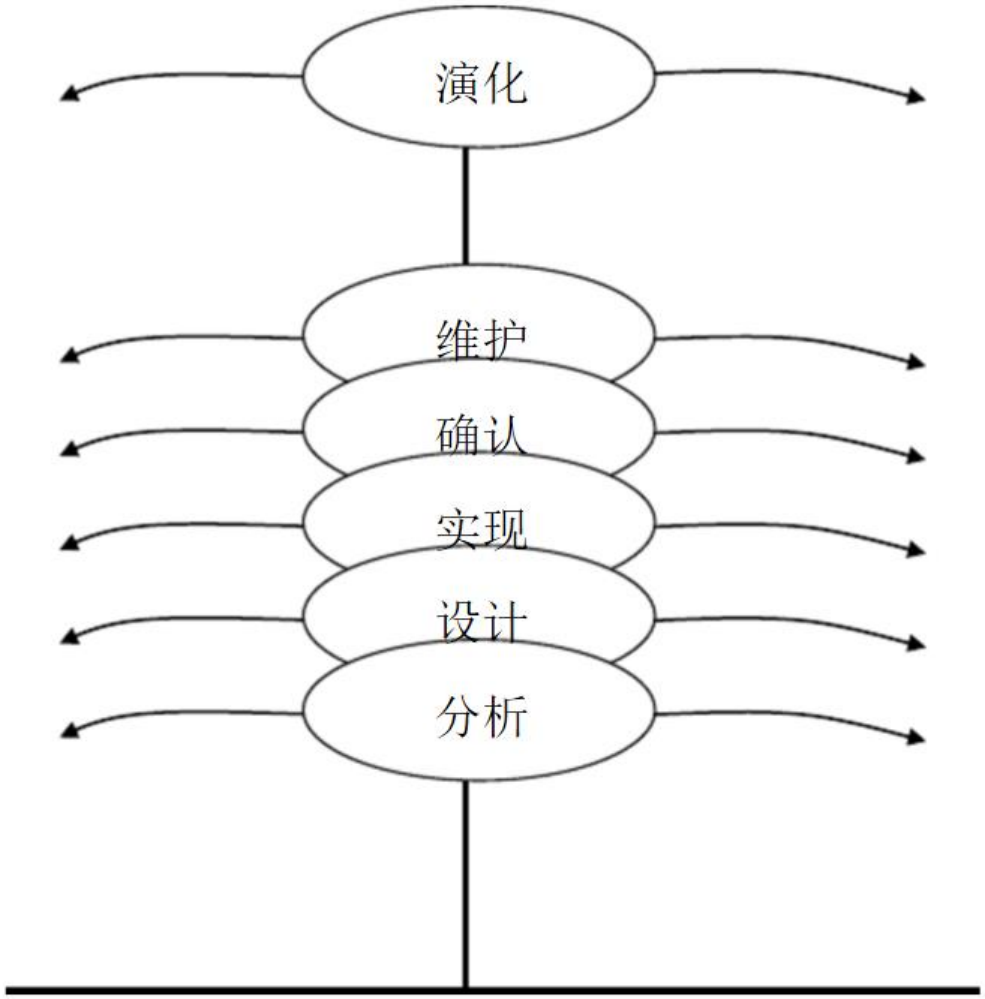
- **快速开发出可以演示的系统，方便了客户沟通。**
- **采用迭代技术能够使开发者逐步弄清客户的需求。**

（快速）原型模型

原型开发存在的问题

- **为了尽快完成原型，开发者没有考虑整体软件的质量和长期的可维护性，系统结构通常较差。**
- **用户可能混淆原型系统和最终系统，原型系统在完全满足用户需求之后可能会被直接交付给客户使用。**

喷泉模型



一种迭代模型

喷泉模型

喷泉模型的优点

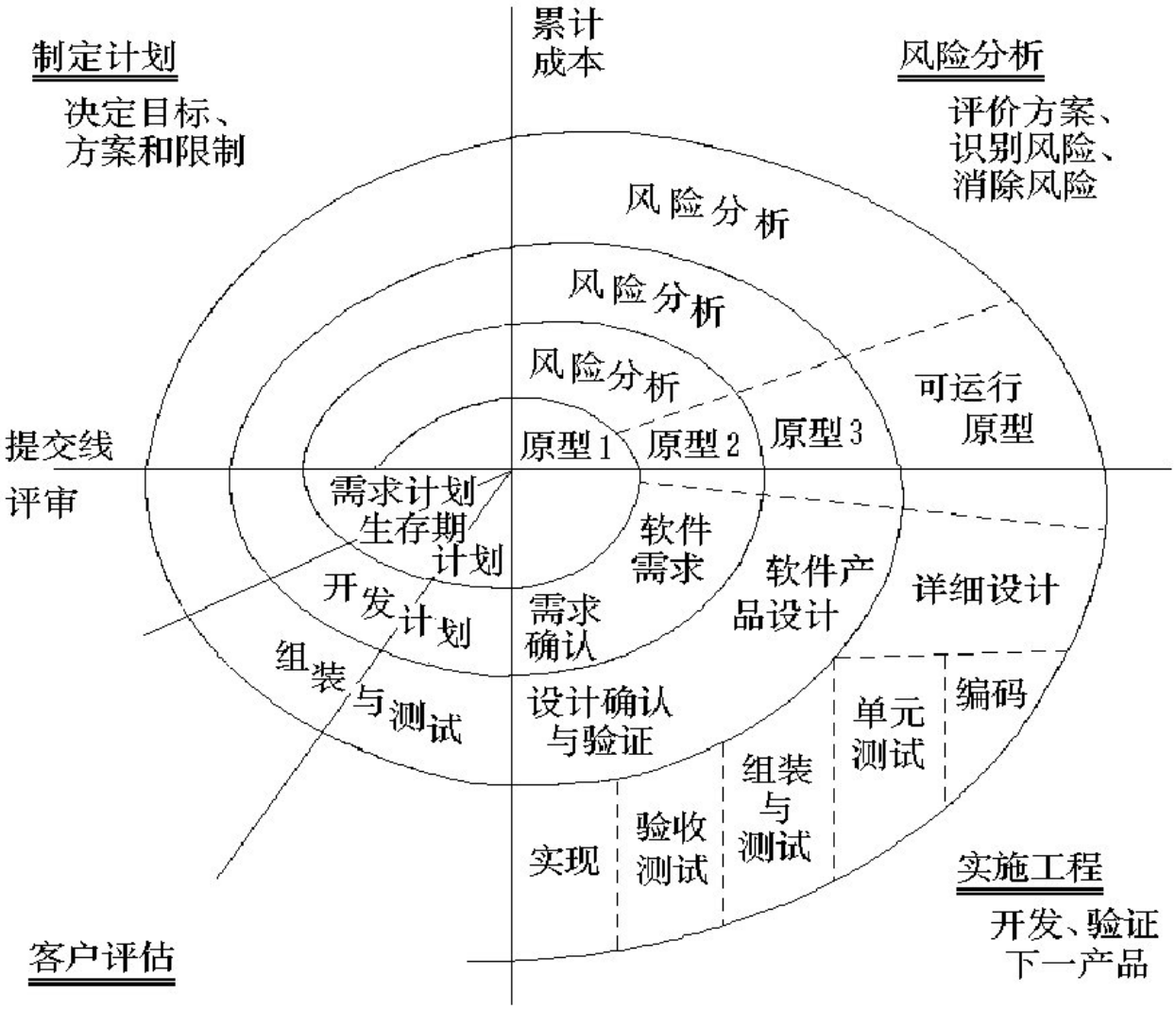
- **该模型的各个阶段没有明显的界限，开发人员可以同步进行开发。可以提高软件项目开发效率，节省开发时间，适应于面向对象的软件开发过程。**

喷泉模型

喷泉模型的缺点

- 由于喷泉模型在各个开发阶段是重叠的，因此在开发过程中需要大量的开发人员，因此不利于项目的管理。此外这种模型要求严格管理文档，使得审核的难度加大，尤其是面对可能随时加入各种信息、需求与资料的情况。

螺旋模型


累计
成本
制定计划
决定目标、
方案和限制
风险分析
评价方案、
识别风险、
消除风险
风险分析
风险分析
风险分析
原型1
原型2
原型3
可运行
原型
提交线
评审
需求计划
生存期
计划
开发计划
组装
与测试
软件
需求
需求
确认
软件产
品设计
详细设计
设计确认
与验证
单元
测试
编码
组装
与
测试
验收
测试
实现
实施工程
开发、验证
下一产品
客户评估

一种迭代模型

螺旋模型

螺旋模型的优点

- 结合了原型的迭代性质与瀑布模型的系统性和可控性，是一种风险驱动型的过程模型。
- 采用循环的方式逐步加深系统定义和实现的深度，同时更好地理解、应对和降低风险。
- 确定一系列里程碑，确保各方都得到可行的系统解决方案。
- 始终保持可操作性，直到软件生命周期的结束。
- 由风险驱动，支持现有软件的复用。

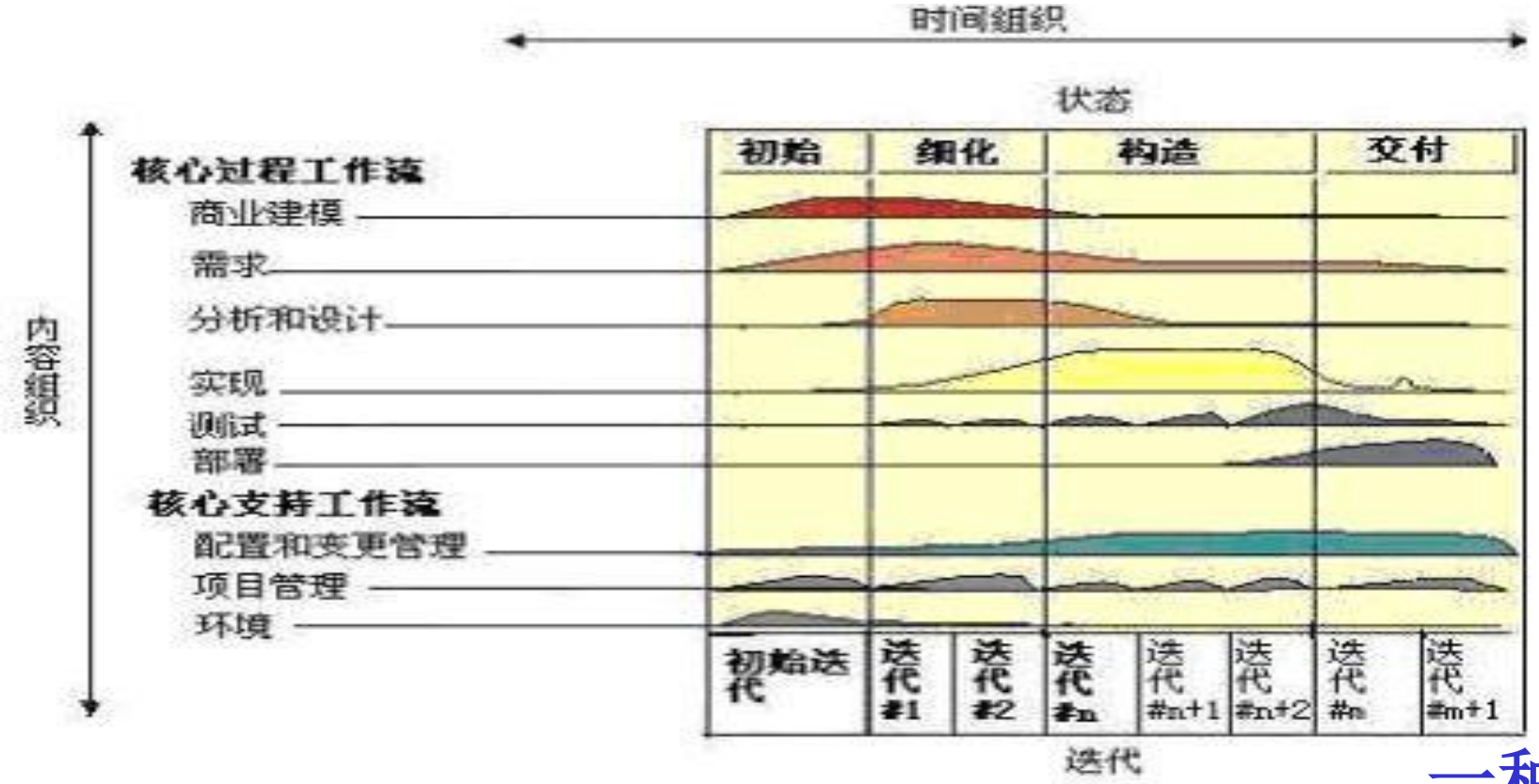
螺旋模型

螺旋模型存在的问题

- 螺旋模型依赖大量的风险评估专家来保证成功。如果有较大的风险没有被发现和管理，肯定会发生问题。
- 软件开发人员应该擅长寻找可能的风险，准确的分析风险，否则将会带来更大的风险。

统一过程模型(RUP-Rational Unified Process)

RUP软件开发生命周期是一个二维的软件开发模型。横轴通过时间组织，是过程展开的生命周期特征，体现开发过程的动态结构，用来描述它的术语主要包括周期(Cycle)、阶段(Phase)、迭代(Iteration)和里程碑(Milestone)；纵轴以内容来组织为自然的逻辑活动，体现开发过程的静态结构



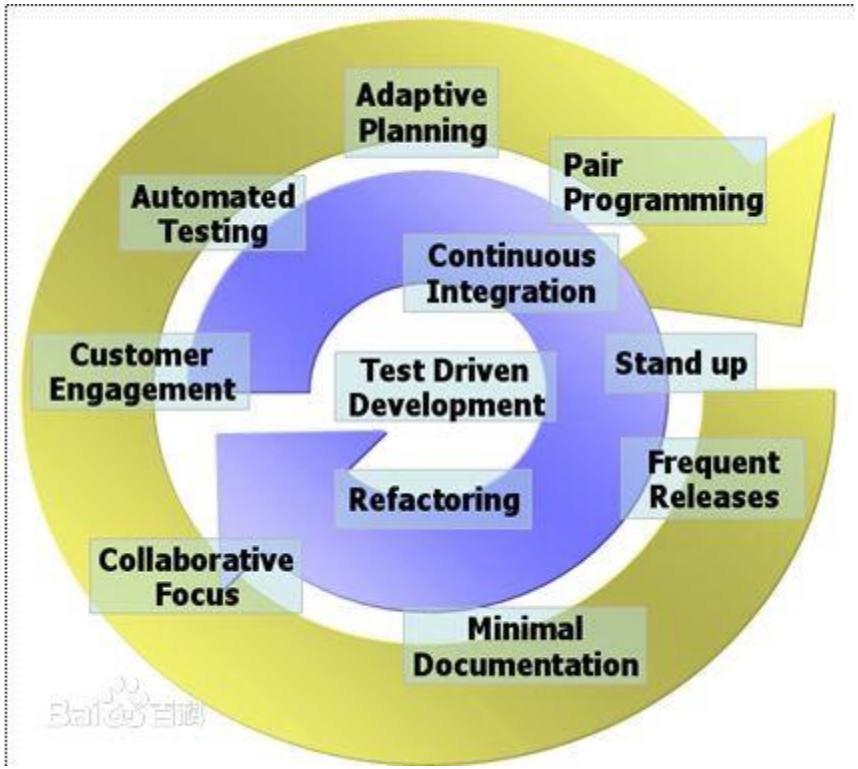
参考：《 The Rational Unified Process : An Introduction》

一种迭代模型

敏捷开发模型（Agile ）

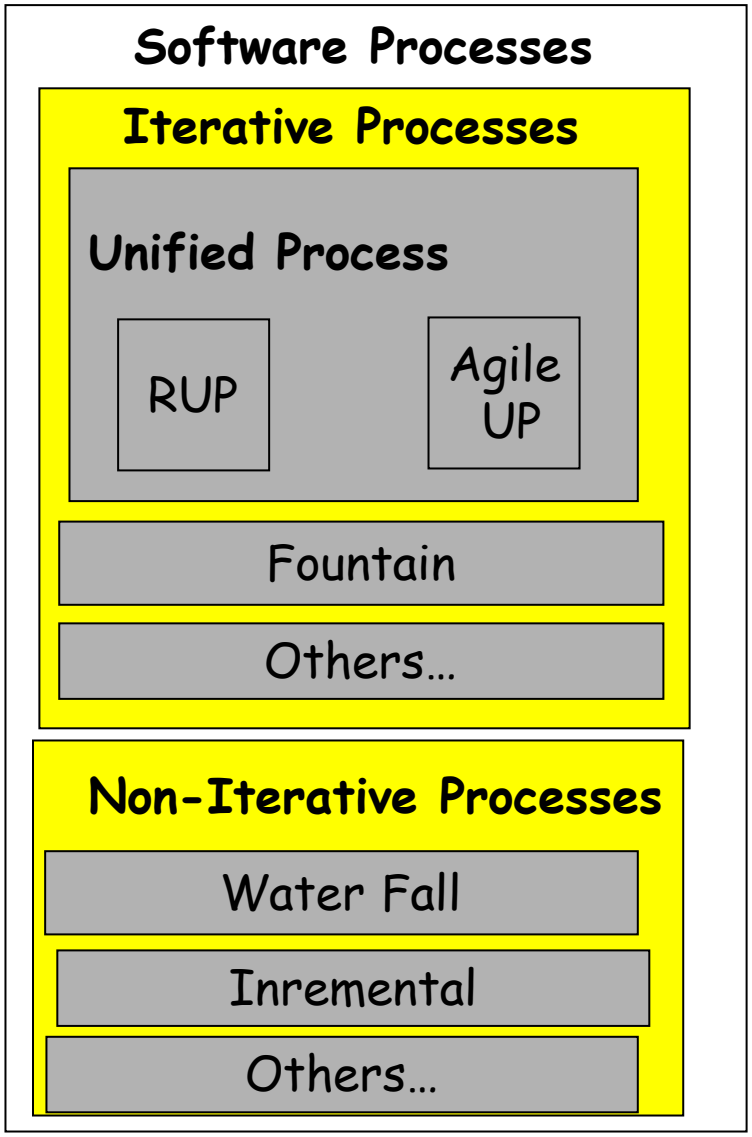
一种应对快速变化的需求的一种软件开发能力。它们的具体名称、理念、过程、术语都不尽相同，相对于“非敏捷”，更强调程序员团队与业务专家之间的紧密协作、面对面的沟通（认为比书面的文档更有效）、频繁交付新的软件版本、紧凑而自我组织型的团队、能够很好地适应需求变化的代码编写和团队组织方法，也更注重软件开发过程中人的作用。

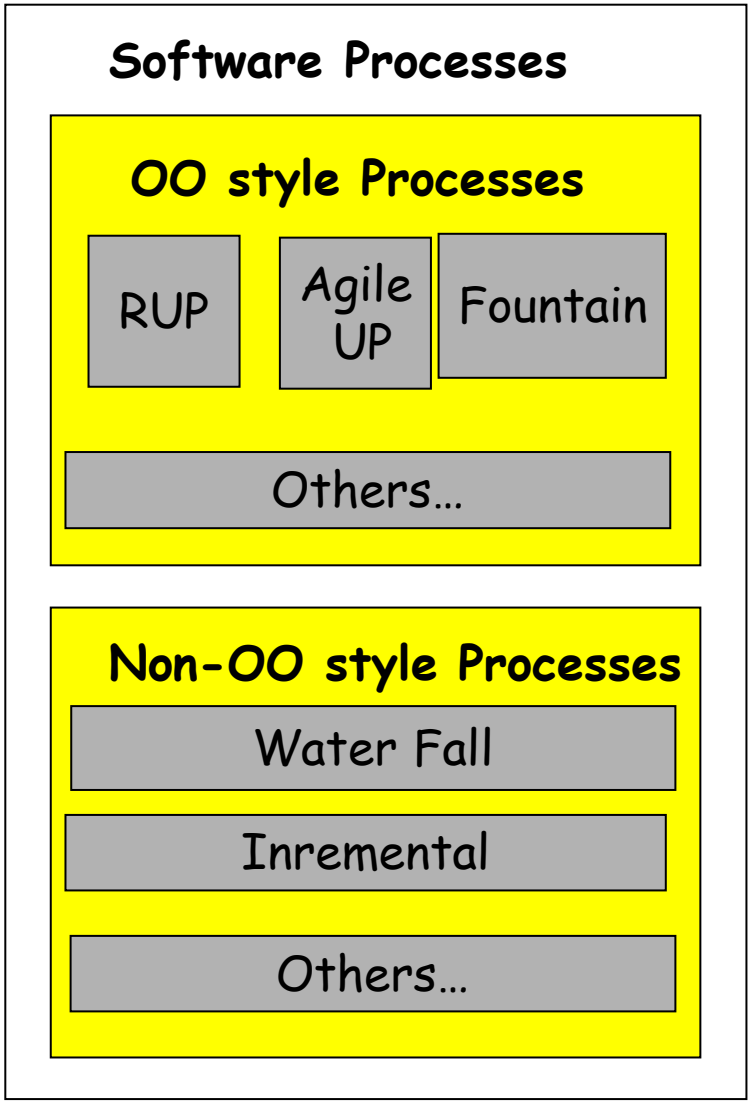
个体和互动：高于 流程和工具。
工作的软件：高于 详尽的文档。
客户合作：高于 合同谈判。
响应变化：高于 遵循计划。



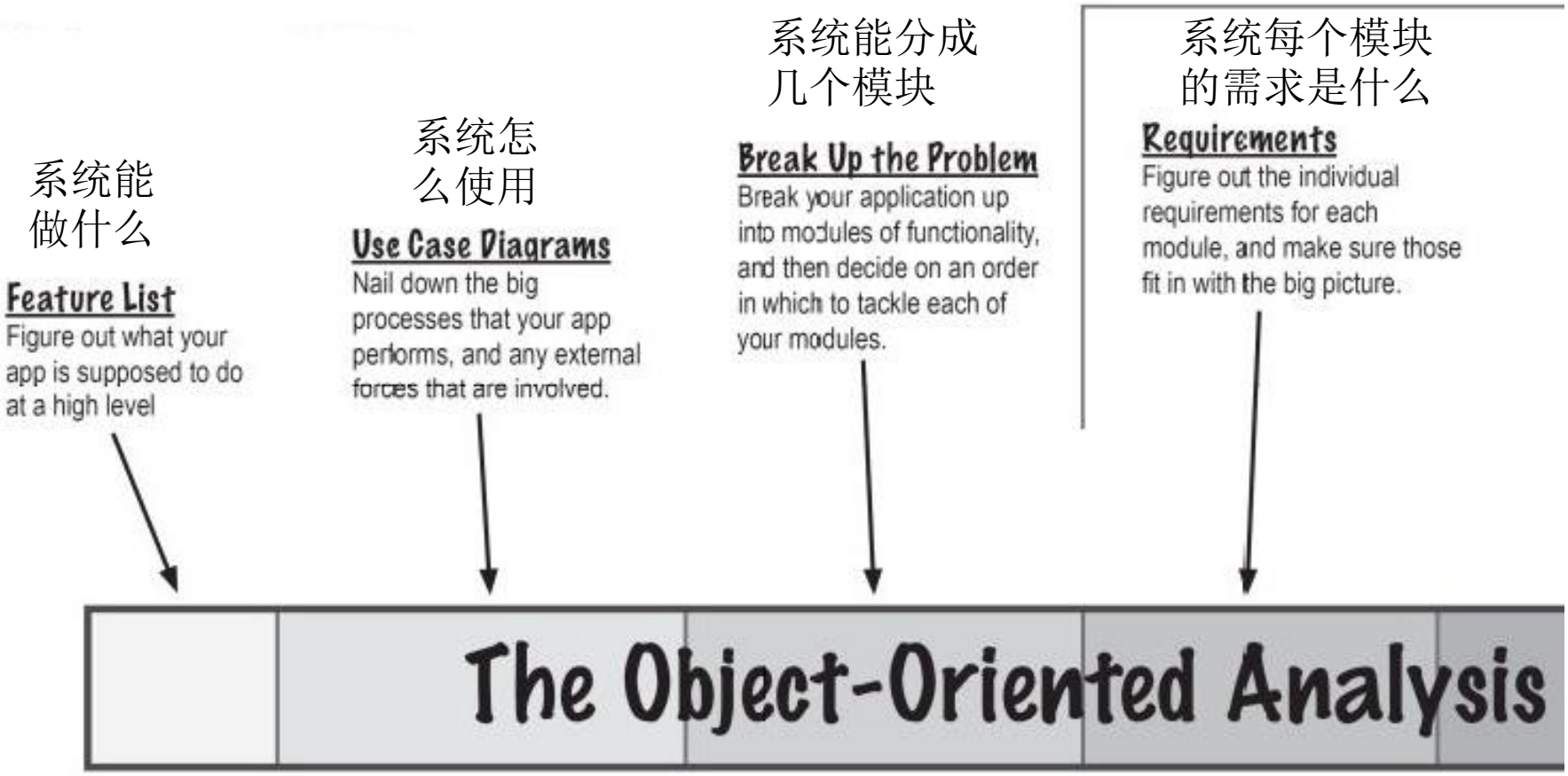
一种迭代模型

软件过程的分类

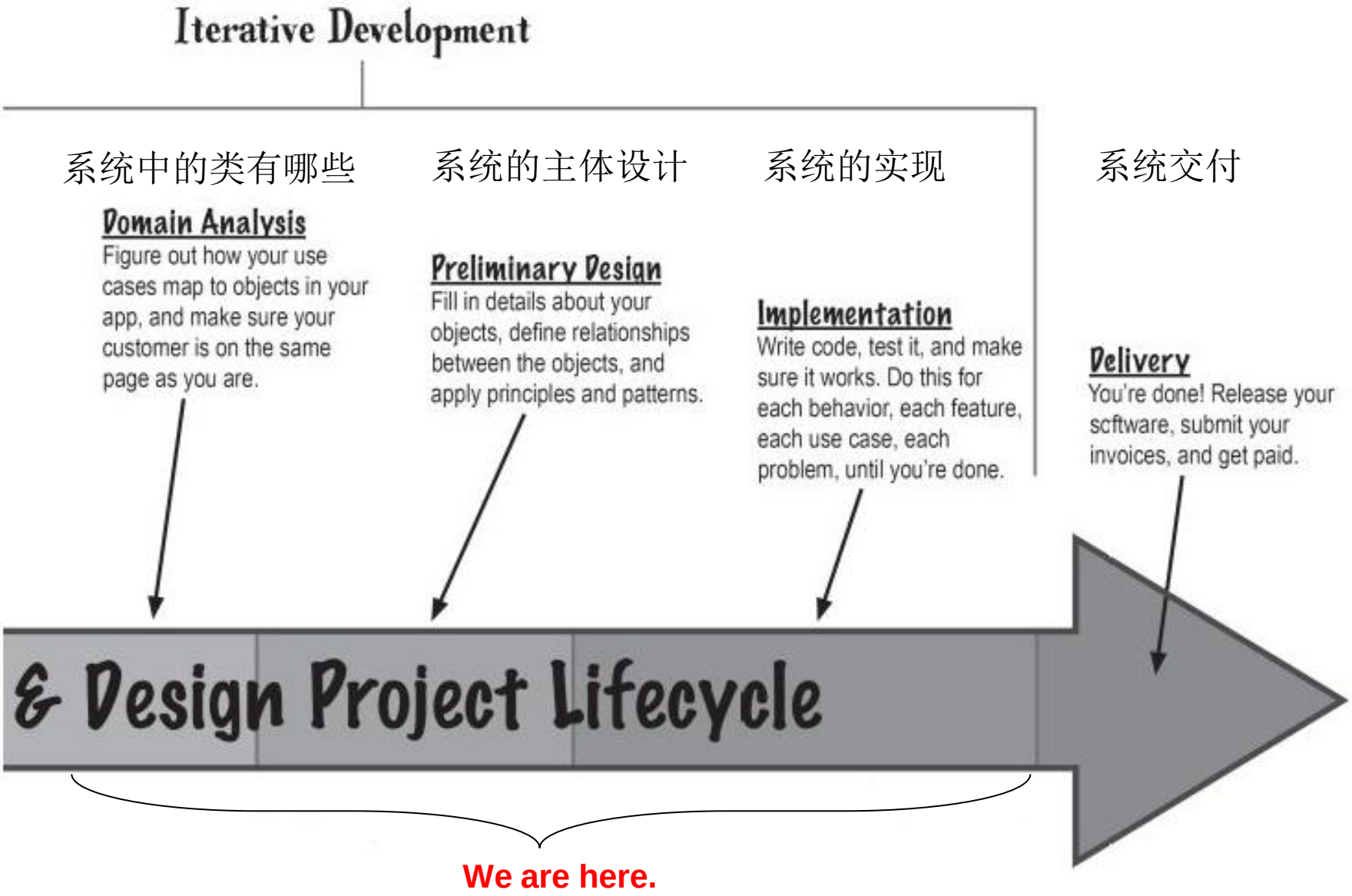

Software Processes
Iterative Processes
Unified Process
RUP
Agile UP
Fountain
Others…
Non-Iterative Processes
Water Fall
Inremental
Others…


Software Processes
OO style Processes
RUP
Agile UP
Fountain
Others…
Non-OO style Processes
Water Fall
Inremental
Others…

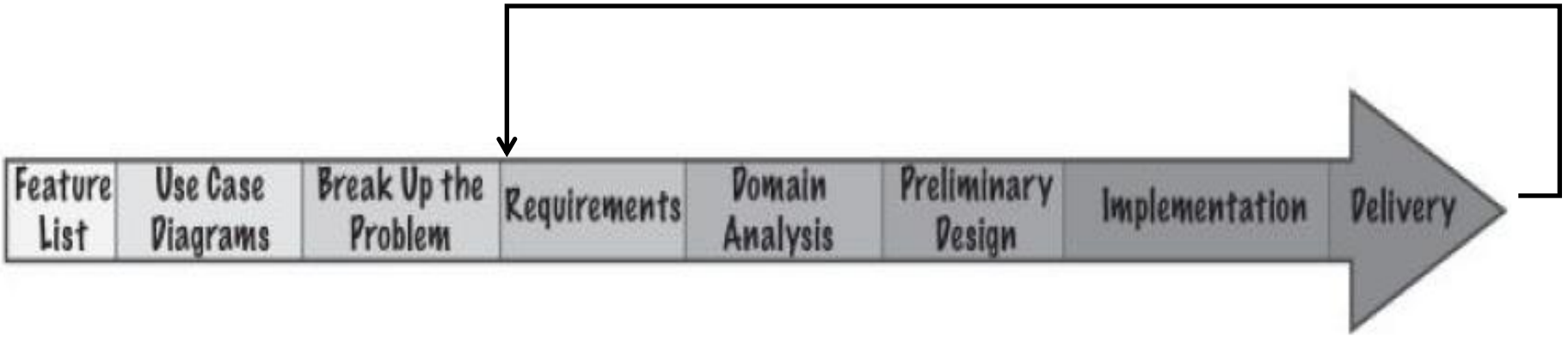
OOA&D Style Process



OOA&D Style Process



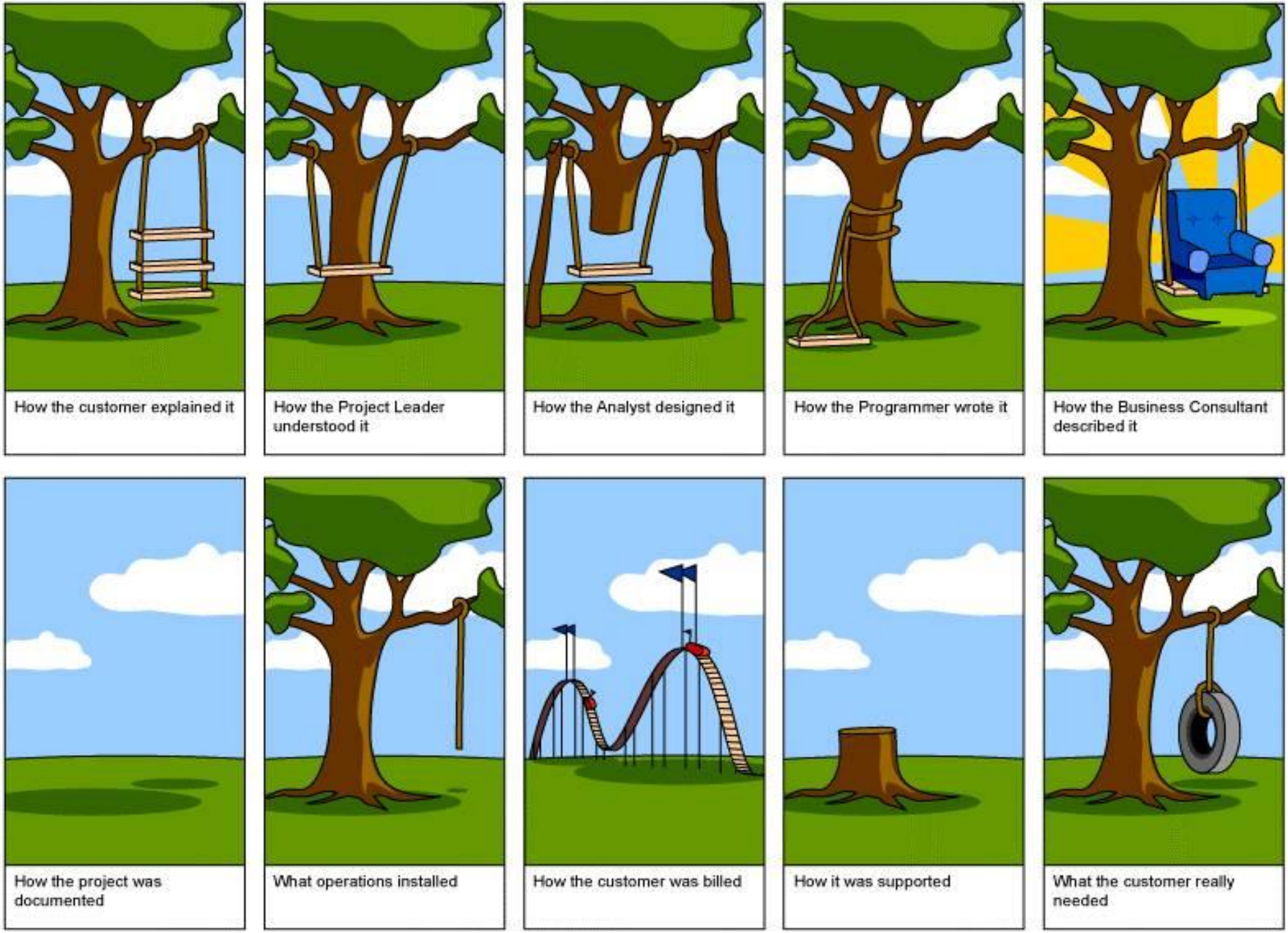
OOA&D Style Process



参考：

《Head First Object-oriented Analysis and Design》

chapter 10


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed