

Lecture 2: 软件过程

软件生命周期与过程模型

概念： 软件生命周期

- 概念：三个时期
 - 软件定义时期、软件开发时期、软件维护时期；
- 概念：八个阶段
 - (1) 问题定义；
 - (2) 可行性研究；
 - (3) 需求分析；
 - (4) 概要设计；
 - (5) 详细设计；
 - (6) 编程；
 - (7) 测试；
 - (8) 运行维护；

软件过程模型

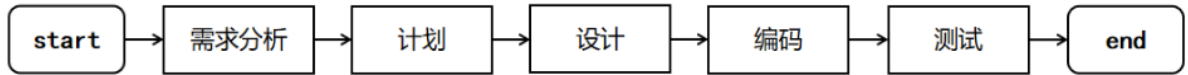
瀑布模型

概念： 瀑布模型

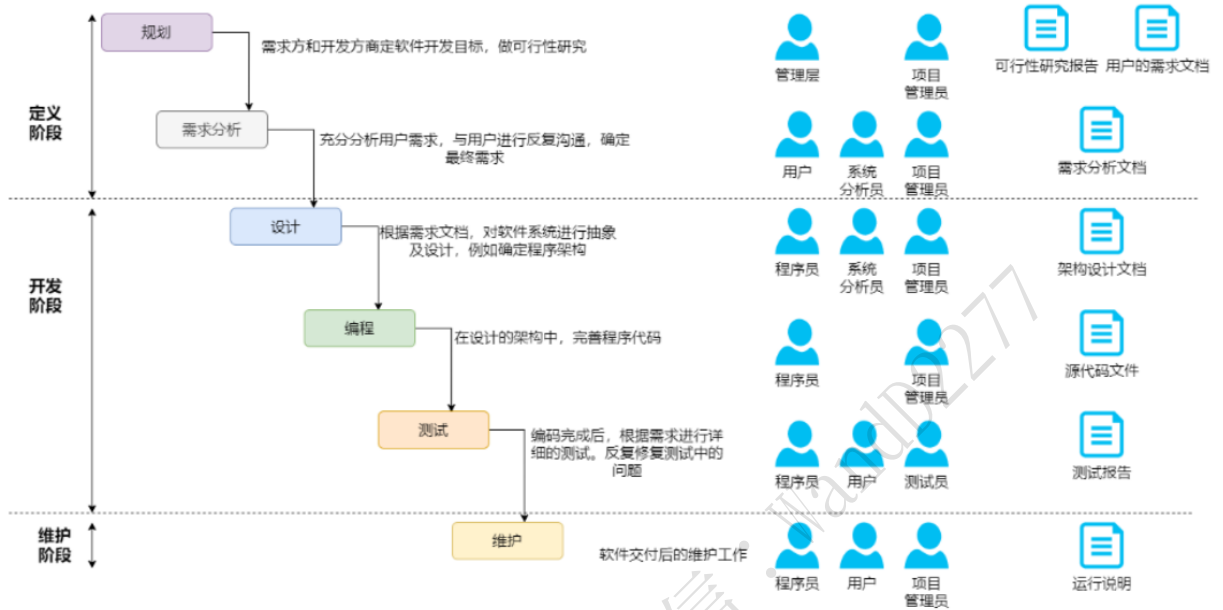
- 概念：
 - 瀑布模型是最早出现的软件开发模型，是所有其他软件开发模型的基础框架。与软件的生命周期不同的是，它缺少了软件运行维护阶段；
 - 每个阶段都只执行一次，因此是线性顺序的软件开发模型，阶段间具有顺序性和依赖性，必须等前一阶段的工作完成之后、才能开始下一阶段的工作；
 - 前一阶段的输出文档就是后一阶段的输入文档；
 - 每个阶段都必须完成规定的文档；每个阶段结束前都要对所完成的文档进行评审；
- 应用：
 - 需求比较明确，且变更很少的项目；
- 特点：
 - 优点：
 - 1. 为项目提供了按阶段划分的检查点，强调开发的阶段性。
 - 2. 强调早期的计划及需求调查。
 - 3. 强调产品测试。
 - 缺点：
 - 1. 只有在项目周期的后期才能看到结果，所以风险往往至后期的测试阶段才显露，因此失去了及早的纠正过程；
 - 2. 单一流程，开发中的经验教训不能反馈应用于本产品的过程；

- 3. 要求在项目开始前就研究清楚需求，但实际的项目需求可能是复杂多变的；

• 图示：



• 总结：



螺旋模型

概念：螺旋模型

• 概念：

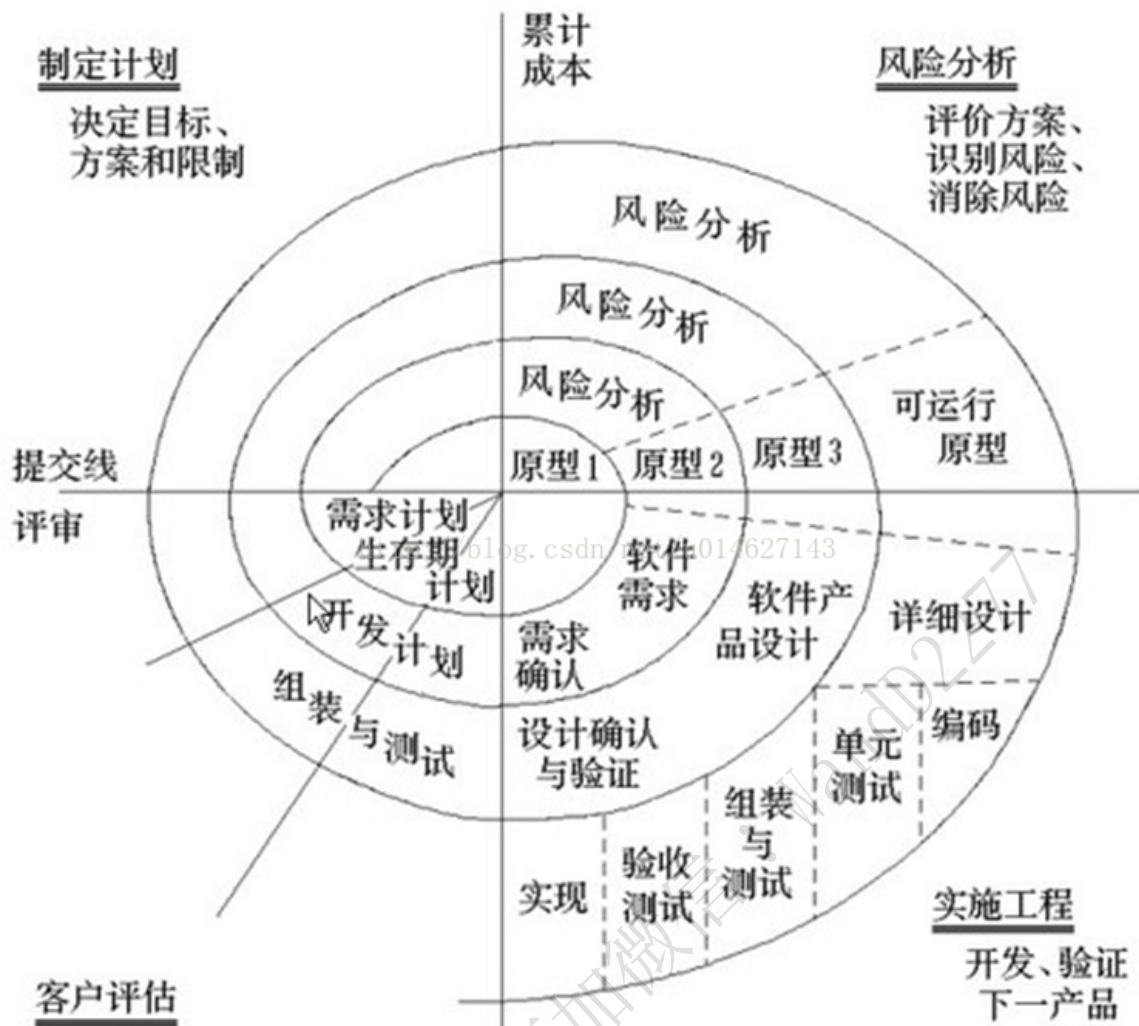
- 针对风险问题演化而来；
- 属于演进模型，融入了循环往复、迭代演进的思想，以原型为基础沿螺旋线旋转、每转一圈都经过计划/风险分析/实施/评估等过程且得到相应新版本、经过若干次螺旋上升得到最终版本；
- 螺旋模型是渐进式开发模型的代表之一；
- 迭代开发的模式给软件测试带来了新的要求，它不允许有一段独立的测试时间和阶段，测试必须跟着开发的迭代而迭代，所以回归测试的很重要；

• 应用：

- 一般在软件开发初期阶段需求不是很明确时，采用渐进式的开发模型；
- 对新近开发，需求不明确的情况下，适合用螺旋模型进行开发，便于风险控制和需求变更；

• 特点：

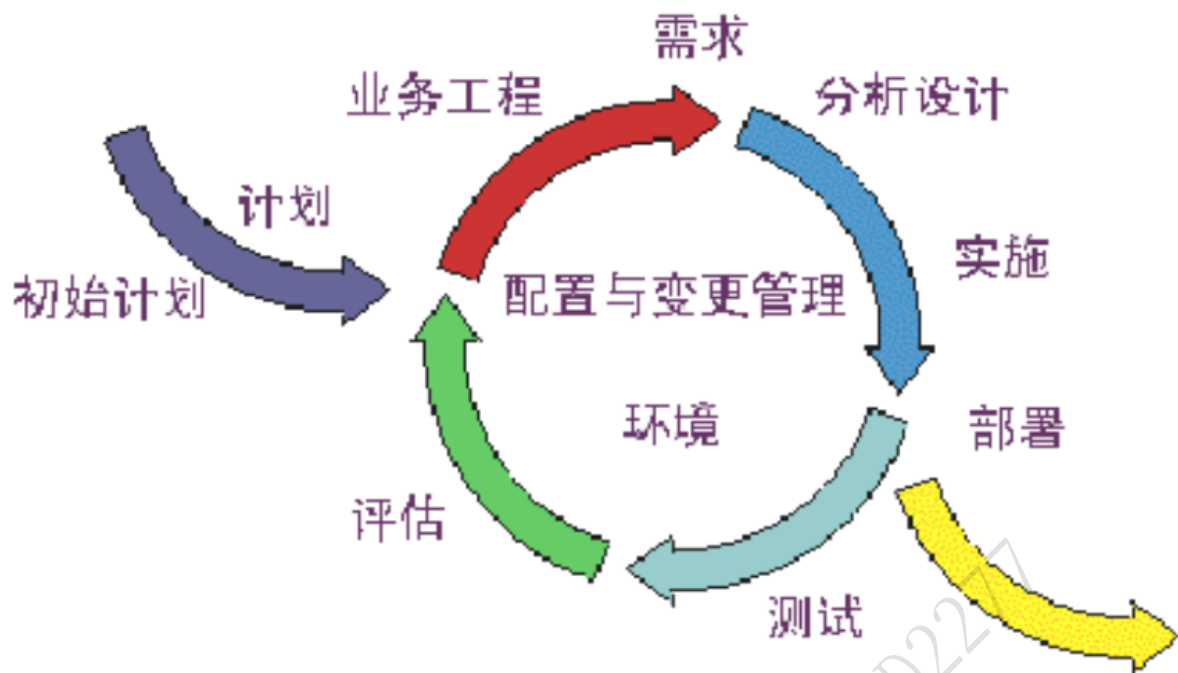
• 图示：



迭代模型

概念： 迭代模型

- 概念：
 - 开发迭代是一次完整地经过所有工作流程的过程：（至少包括）需求工作流程、分析设计工作流程、实施工作流程和测试工作流程；
 - 实质上，迭代模型是类似小型的瀑布式项目；
 - 每一个迭代都会产生一个可以发布的产品，这个产品是最终产品的一个子集；
- 图示：



• 总结:



增量模型

概念：增量模型

• 概念:

- 增量模型综合了线性过程流和并行过程流的特征，随着时间的推移，在每个阶段都运用线性序列；
- 整体上，按照瀑布模型的流程实施项目开发；
- 软件的实际创建中，将软件系统按功能分解为许多增量构件，并以构件为单位逐个地创建与交付，直到全部增量构件创建完毕，集成到系统之中交付用户使用；
- “增量”的理解：从一组给定的需求开始，通过构造一系列可执行中间版本来实施开发活动。第一个版本纳入一部分需求，下一个版本纳入更多的需求，依此类推，直到系统完成；
- 每个中间版本都要执行必需的过程、活动和任务；

• 特点:

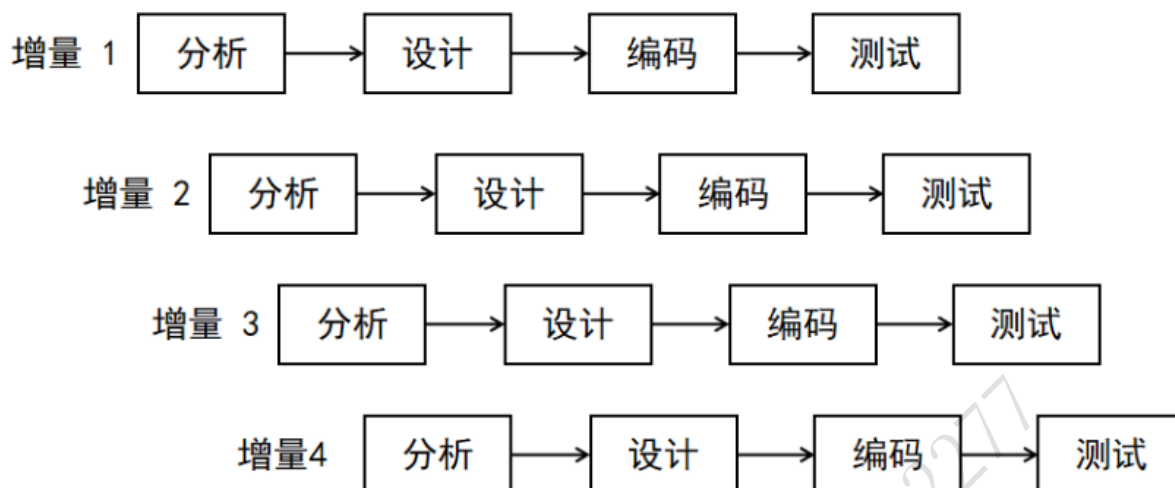
• 优点:

- 开始时不用投入大量人力资源，可以事先推出核心产品以稳定用户，可以有计划的管理技术风险；

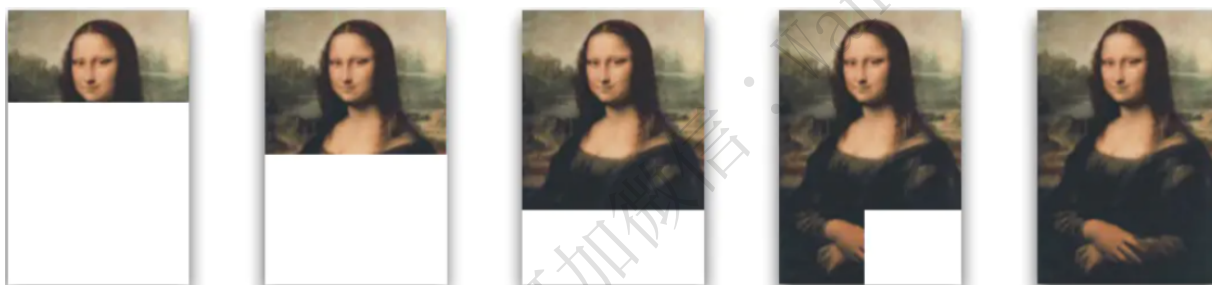
• 缺点:

- 需要开放式体系结构，可能会产生设计效果差、开发效率低的情况；

- 应用：
 - 需求经常发生改变的软件开发过程；
- 图示：



- 总结：



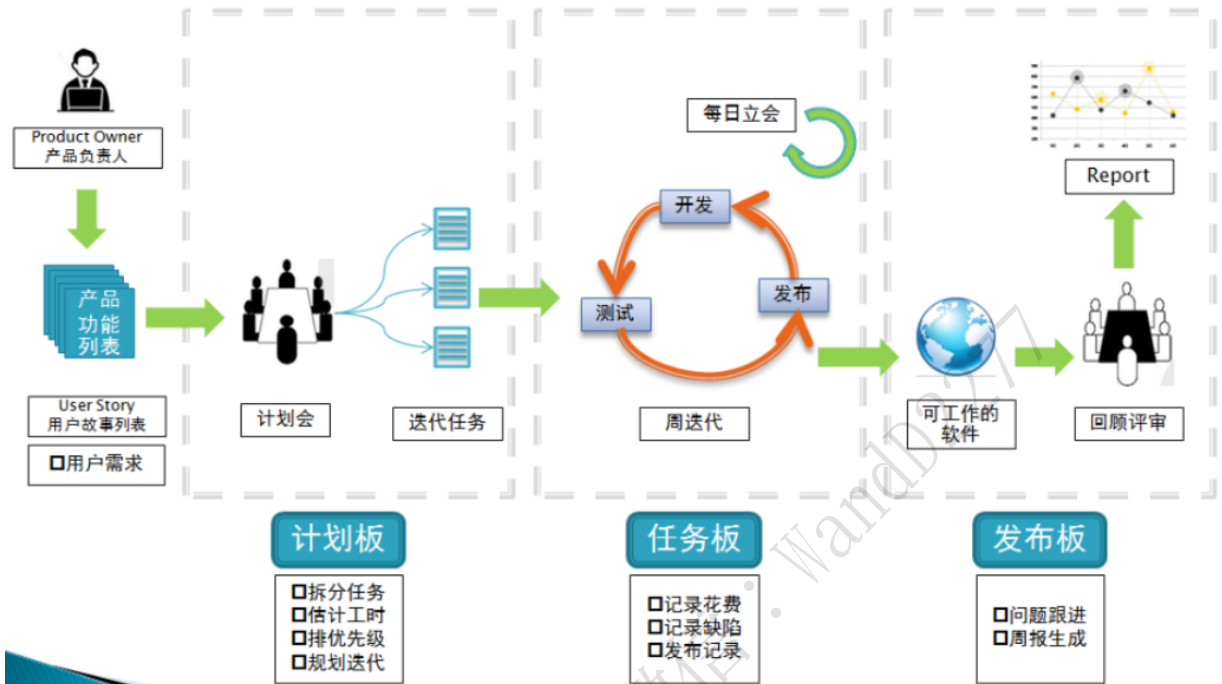
敏捷模型

概念：敏捷模型

- 概念：
 - 极限编程是一个轻量级的、灵巧的软件开发方法，同时它也是一个非常严谨和周密的方法；
 - 它的基础和价值观是交流、朴素、反馈和勇气（任何一个软件项目都可以从四个方面入手进行改善：加强交流；从简单做起；寻求反馈；勇于实事求是。）
 - 敏捷开发以用户的需求进化为核心，采用迭代、循序渐进的方法进行软件开发。是一种近螺旋式的开发方法，它将复杂的开发过程分解为一个个相对比较简单的小周期，通过积极的交流、反馈以及其它一系列的方法，开发人员和客户可以非常清楚开发进度、变化、待解决的问题和潜在的困难等，并根据实际情况及时地调整开发过程；
- 四句话：
 - 「个体与交互」胜过「过程与工具」
 - 「可以工作的软件」胜过「面面俱到的文档」
 - 「客户协作」胜过「合同谈判」
 - 「响应变化」胜过「遵循计划」
- 在敏捷开发中，软件项目在构建初期被切分成多个子项目，各个子项目的成果都经过测试，具备可视、可集成和可运行使用的特征；

- 换言之，就是把一个大项目分为多个相互联系，但也可独立运行的小项目，并分别完成，在此过程中软件一直处于可使用状态；
- 敏捷开发借助互联网浪潮开始流行起来，相比瀑布模式，敏捷无疑更加贴近互联网时代背景下快速发展变化的市场环境以及业务需求；

• 图示：



软件过程概念与 CMM

CMM 核心思想

概念： CMM 核心思想

- 概念：
 - CMM，全称为 Capability Maturity Model for Software，即“软件能力成熟度模型”，是对组织软件过程的描述，核心内容是将软件开发视为一个过程，并且根据相应的原则对于软件开发进行相应的监控和研究；
 - CMM 是一个将软件不断从混乱走向成熟的规范化过程的一个框架，CMM 自从 1987 年开始实施以来，已经成为软件业权威的评估认证体系；
 - 帮助软件开发组织建立一个成熟的、有规律的软件过程，使其能开发出质量更好的软件；
- 概念：能力成熟度等级
 - CMM 一共有 5 个等级，分别为初始级、可重复级、已定义级、已管理级、优化级；

CMM 等级结构

概念： 初始级

- 概念：
 - 初始级时，对于软件的管理制度较为缺乏，过程缺乏定义；
 - 初始级时成功是依靠出众的个人能力和经验，工作无序，项目进行过程中常常会放弃当初的规划，开发项目的成效也不稳定；

- 团队对于管理无章，缺乏健全的管理制度；
- 特点：
 - 机构没有明确的管理制度，软件没有稳定的工作环境，制订计划没有有效地执行。
 - 成功比较依赖于个人能力。
 - 在紧急情况下，急于编码和测试。规定的过程无法克服因为缺乏管理而带来的不稳定性。

概念：可重复性

- 概念：
 - 管理制度化，建立了基本的管理制度以及规程，管理工作有章可循；
 - 管理人员可以采取一定的措施控制费用和时间，可以及时发现问题并且采取措施，在一定程度上可以重复类似项目的软件开发；
- 特点：
 - 新的项目可以根据以往类似的项目的经验；
 - 软件的需求和产品的都有基线的控制；
 - 初步实现了标准化，开发工作有了较好的实施标准；
 - 形成了基本的管理制度，管理过程有章可循；

概念：已定义级

- 概念：
 - 在开发的过程中，技术工作以及管理工作开始文档化和标准化。采用了评审的制度保证了软件质量；
- 特点：
 - 建立了完善的培训制度以及专家评审制度；
 - 全部技术活动和管理活动均可以稳定实施；
 - 项目的质量和费用均得到了控制。

概念：已管理级

- 概念：
 - 已管理级能够制订效率目标并且收集和测试，可以利用统计数据来进行相应的改进，对于软件进程以及产品质量有定量的理解和控制；
- 特点：
 - 已经建立了过程数据库；
 - 实现项目产品和过程的控制；
 - 可以预测过程和质量；
 - 生产过程中的生产效率和质量是可度量的。

概念：优化级

- 概念：
 - 优化级是持续改进软件的过程，效率以及质量都稳步提升。
- 特点：
 - 采用较新的技术和方法；
 - 拥有防止出现缺陷的手段；

- 可以取得过程有效性的统计数据，并且能够对其进行分析，进而取得更好的方法；

获取笔记持续更新可加微信：WandD2277