中国石油大学（华东）
CHINA UNIVERSITY OF PETROLEUM

软件工程

第一章　软件工程学概述

第一节　软件危机

第二节　软件工程

第三节　软件生命周期

第四节　软件过程

第四节　软件过程

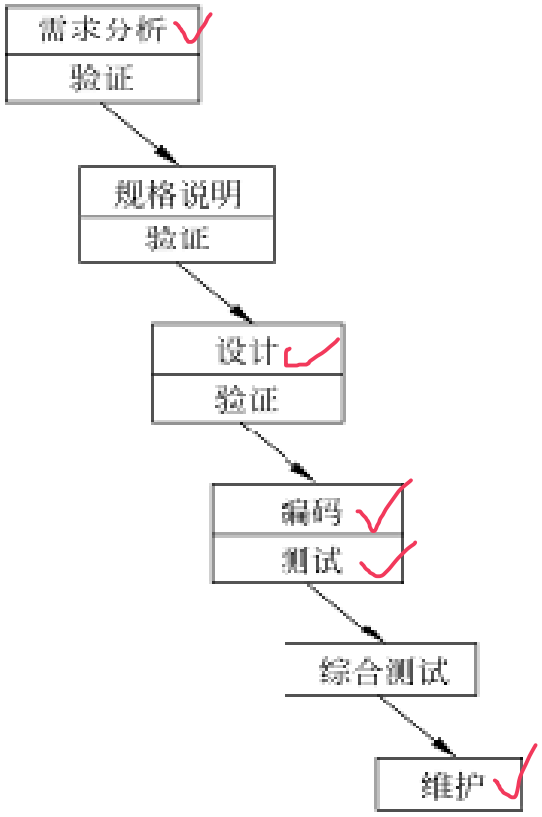
软件过程是为了获得高质量软件所需要完成的一系列任务的框架，它规定了完成各项任务的工作步骤。

过程定义了运用方法的顺序、应该交付的文档资料、为保证软件质量和协调变化所需要采取的管理措施，以及标志软件开发各个阶段任务完成的里程碑。

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，也成为过程模型。

1. 瀑布模型(Waterfall Model)

- 传统的瀑布模型



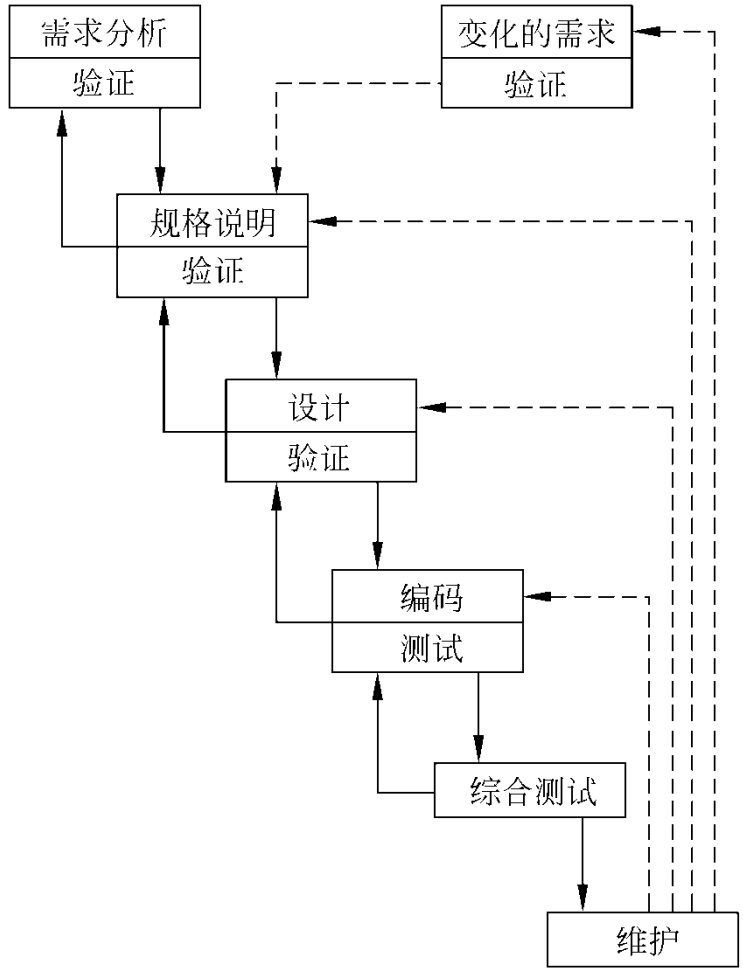
特点：

1. 阶段间具有顺序性和依赖性
2. 推迟实现的观点
3. 质量保证的观点：
   - 每个阶段都必须完成规定的文档。
   - 每个阶段结束之前，都必须对完成的文档进行严格的评审。

传统瀑布模型存在什么问题？

1. 瀑布模型(Waterfall Model)

- 实际的瀑布模型



——→ 开发

- - - - - → 维护

实际的瀑布模型是带反馈环的。

- 瀑布模型的优点：

  可强迫开发人员采用规范的方法（例如，结构化技术）；

  严格地规定了每个阶段必须提交的文档；

  要求每个阶段交出的所有产品都必须经过质量保证小组的仔细验证。

  瀑布模型的成功在很大程度上是由于它基本上是一种文档驱动的模型。

- 瀑布模型的缺点：

  用户常常难以清楚地给出所有需求；

  用户必须有耐心，等到系统开发完成；

  开发者常常被不必要地耽搁。

  可能最终开发出的软件产品不能真正满足用户的需要。

2. 快速原型模型（Rapid Prototype Model）

就是快速建立起来的可以实际运行的程序，它所能完成的功能只是最终产品的一个子集（展示了目标系统的关键功能）。

快速原型模型的第一步是快速建立一个能反映用户主要需求的原型系统，让用户在计算机上试用它，通过实践来了解目标系统的概貌。

快速原型的本质是“快速”。开发人员应该尽可能快地建造出原型系统，以加速软件开发过程，节约软件开发成本。

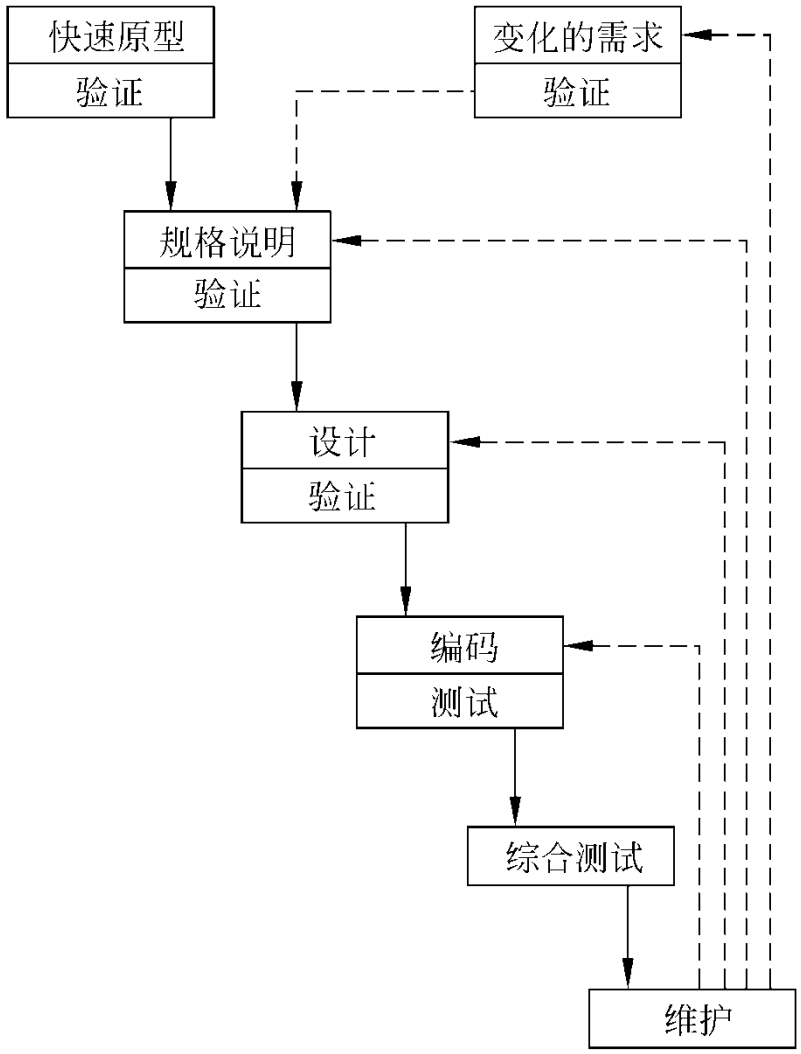
2. 快速原型模型

标准的快速原型模型是不带反馈环的，软件产品的开发基本上是线性顺序进行的。

原因：

- 原型系统已经通过验证；
- 开发人员通过开发原型系统积累了一定经验。

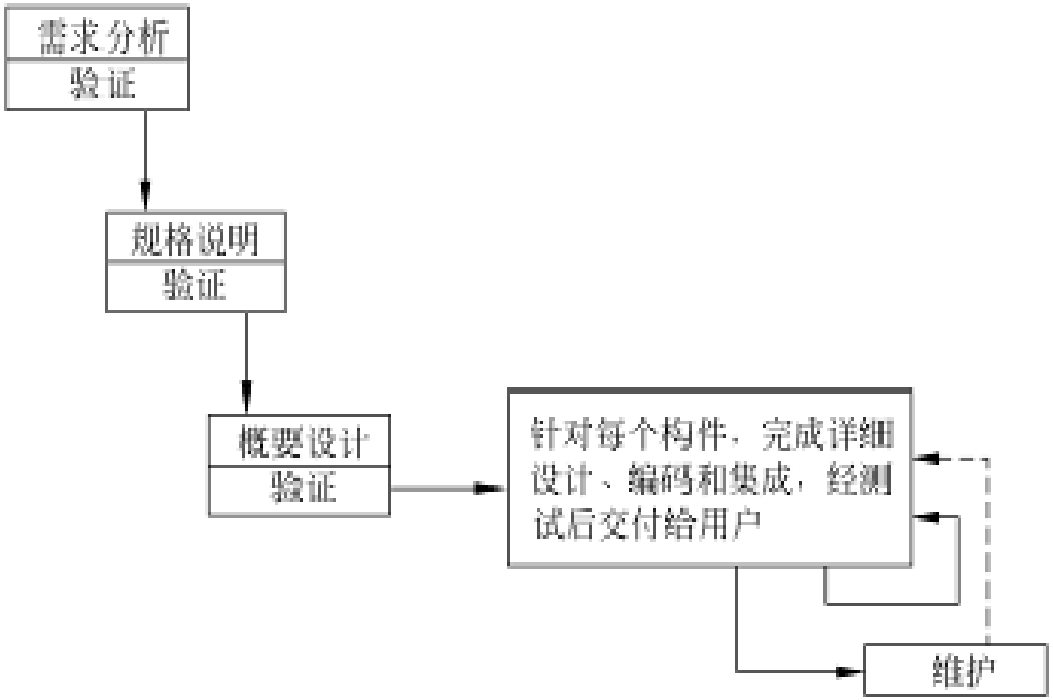
优点：保证用户的真实需要得到满足
不带反馈环的



3. 增量模型(Incremental Model)

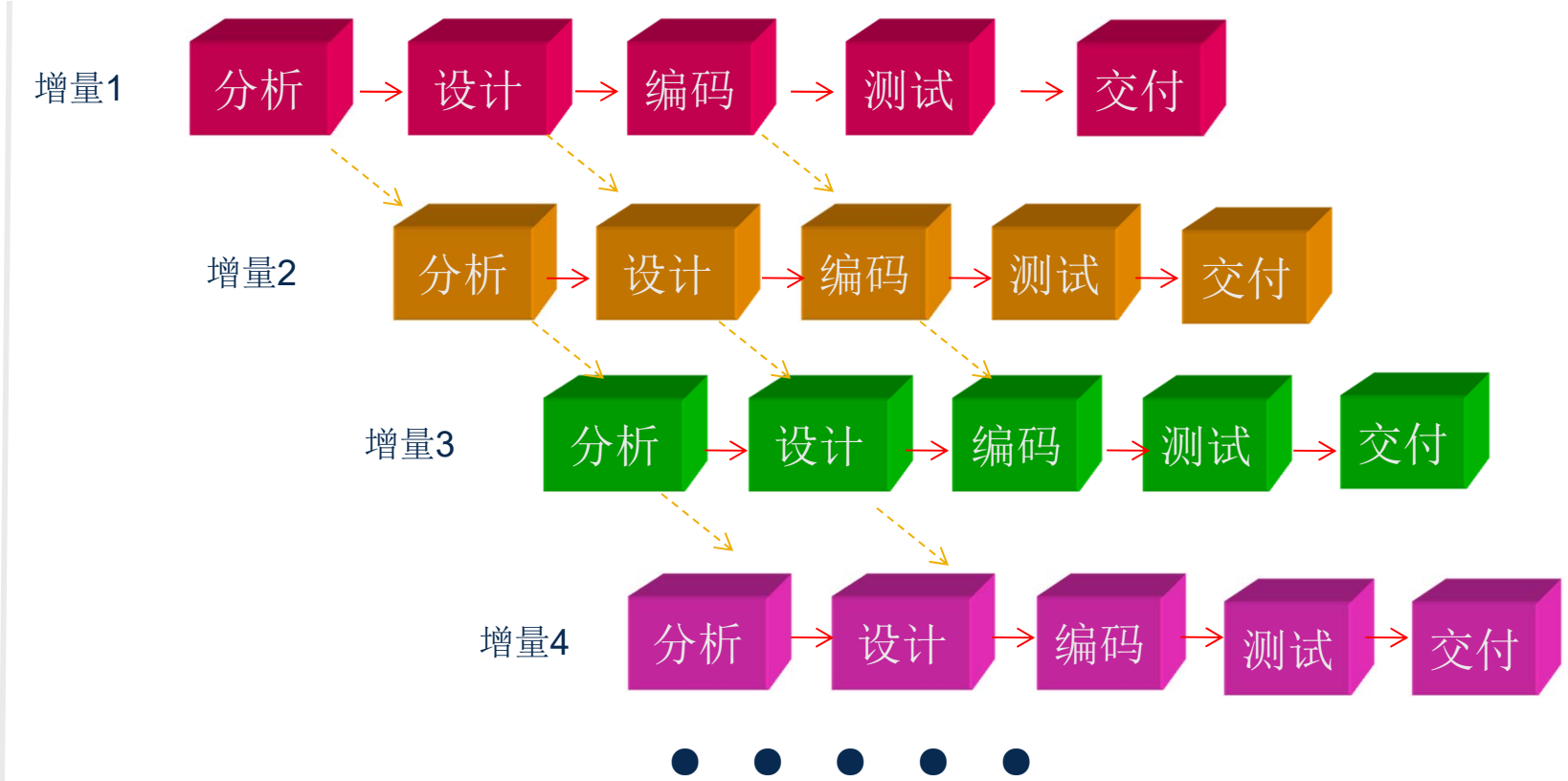
也叫渐增模型,把软件产品分解成一系列的增量构件，在增量开发迭代中逐步加入。

- 每个构件由多个相互作用的模块构成，并且能够完成特定的功能。
- 早先完成的增量可以为后期的增量提供服务。



3. 增量模型

风险更大的增量模型：



3. 增量模型

特点：

- 增量模型是一个构件接一个构件地提交，因此能在较短的时间内向用户提交可完成部分工作的产品；
- 逐步增加产品功能可以使用户有较充裕的时间学习和适应新产品；
- 在把每个新的增量构件集成到现有软件体系结构中时，必须不破坏原来已经开发出的产品。因此，软件体系结构必须是开放的。
- 开发人员既要把软件看作一个整体，又要把它分解成合适的构件序列，每个构件本质上都独立于另一个构件，这一点较难做到。

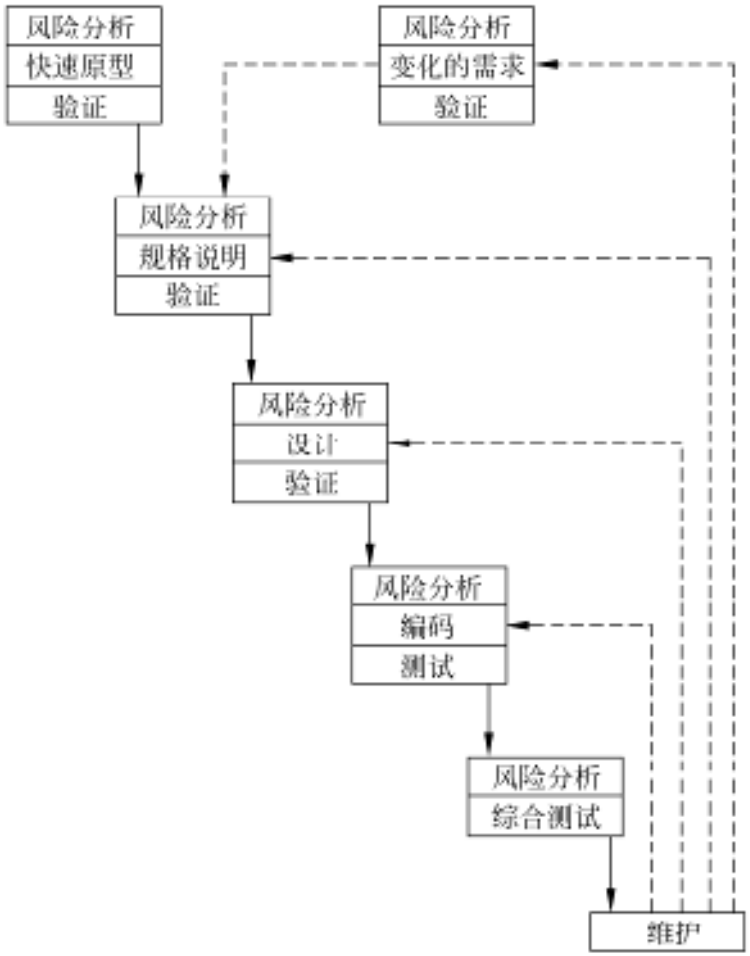
4. 螺旋模型(Spiral Model)

（1）基本思想：

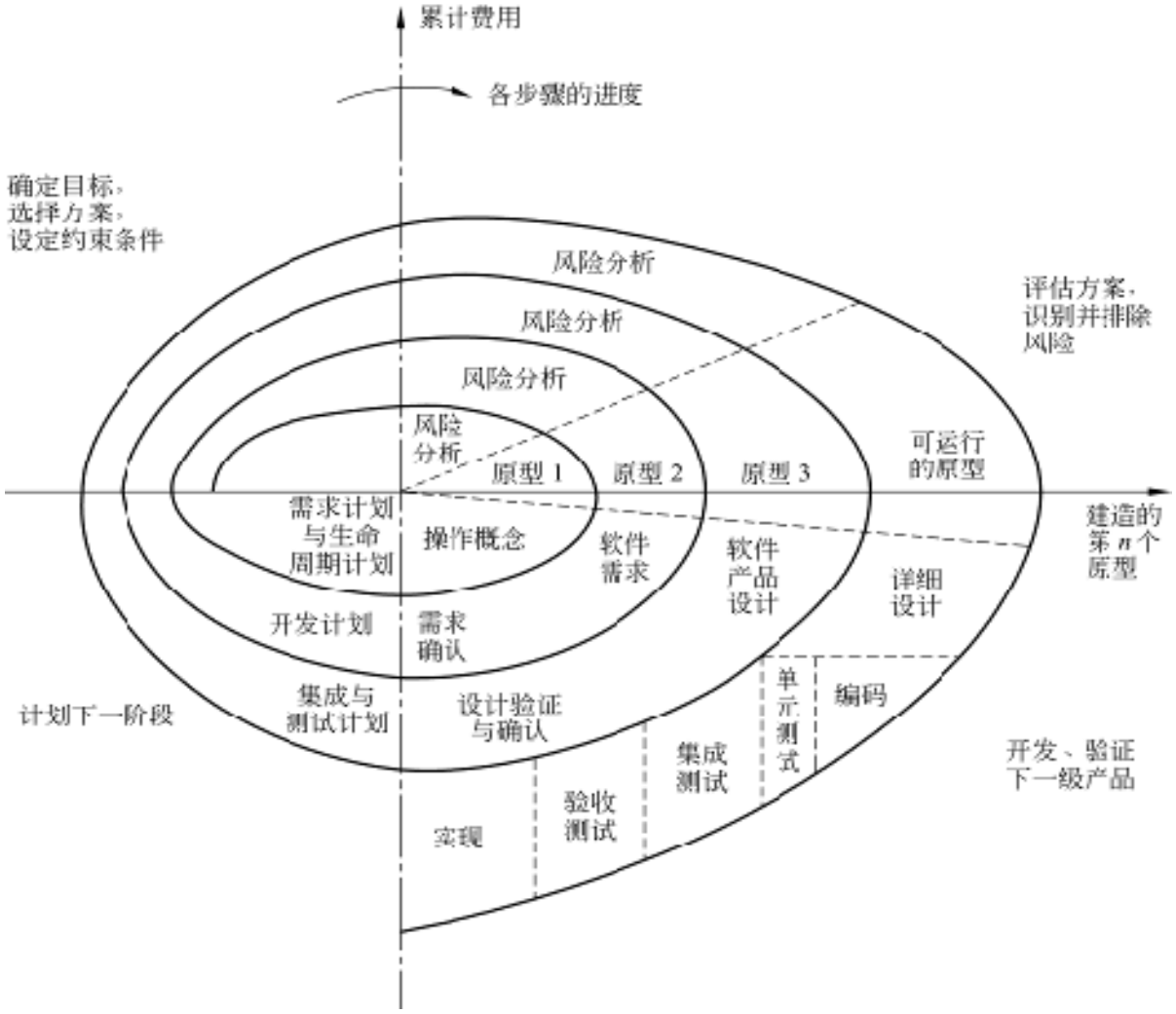
螺旋模型是一种风险驱动的模型，通过在开发过程的每个阶段之前增加风险分析过程来尽量降低风险。

（2）简化的螺旋模型：

就是每个阶段之前增加了风险分析过程的快速原形模型。



（3）完整的螺旋模型


累计费用
各步骤的进度
确定目标，
选择方案，
设定约束条件
风险分析
风险分析
风险分析
风险
分析
评估方案，
识别并排除
风险
可运行
的原型
原型 1
原型 2
原型 3
建造的
第 n 个
原型
需求计划
与生命
周期计划
操作概念
软件
需求
软件
产品
设计
详细
设计
开发计划
需求
确认
计划下一阶段
集成与
测试计划
设计验证
与确认
单元测试
编码
集成
测试
验收
测试
实现
开发、验证
下一级产品

（4）螺旋模型的特点

- 优点：

对可选方案和约束条件的强调有利于已有软件的重用，也有助于把软件质量作为软件开发的一个重要目标；减少了过多测试或测试不足带来的风险；维护和开发之间并没有本质区别。

- 缺点：

需要开发人员具有丰富的风险评估经验和这方面的专门知识，否则风险更大。

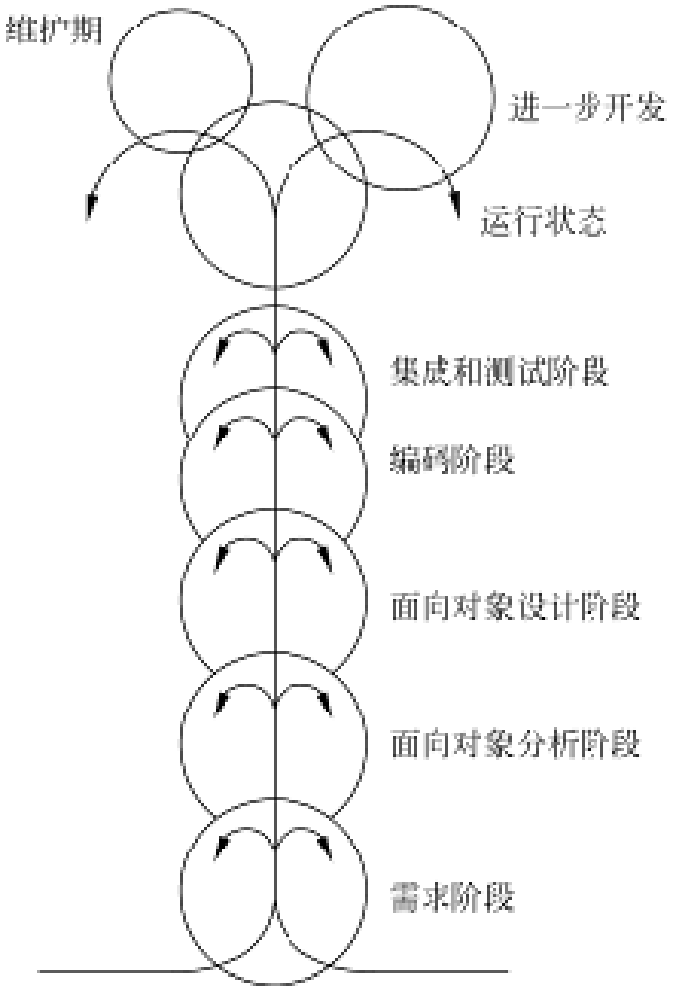
螺旋模型主要适用于内部开发的大规模软件项目。

5. 喷泉模型(Fountain Model)

喷泉模型是一种以用户需求为动力，以对象为驱动的模型，主要用于描述面向对象的软件开发过程。

软件开发过程自下而上周期的各阶段是相互重叠和多次反复的，就像水喷上去又可以落下来，类似一个喷泉。

各个开发阶段没有特定的次序要求，并且可以交互进行，可以在某个开发阶段中随时补充其他任何开发阶段中的遗漏。



5. 喷泉模型

- 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。

- 缺点：

由于该模型在各个开发阶段是重叠的，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。

该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。

6. Rational统一过程（Rational Unified Process, RUP ）

（1）最佳实践

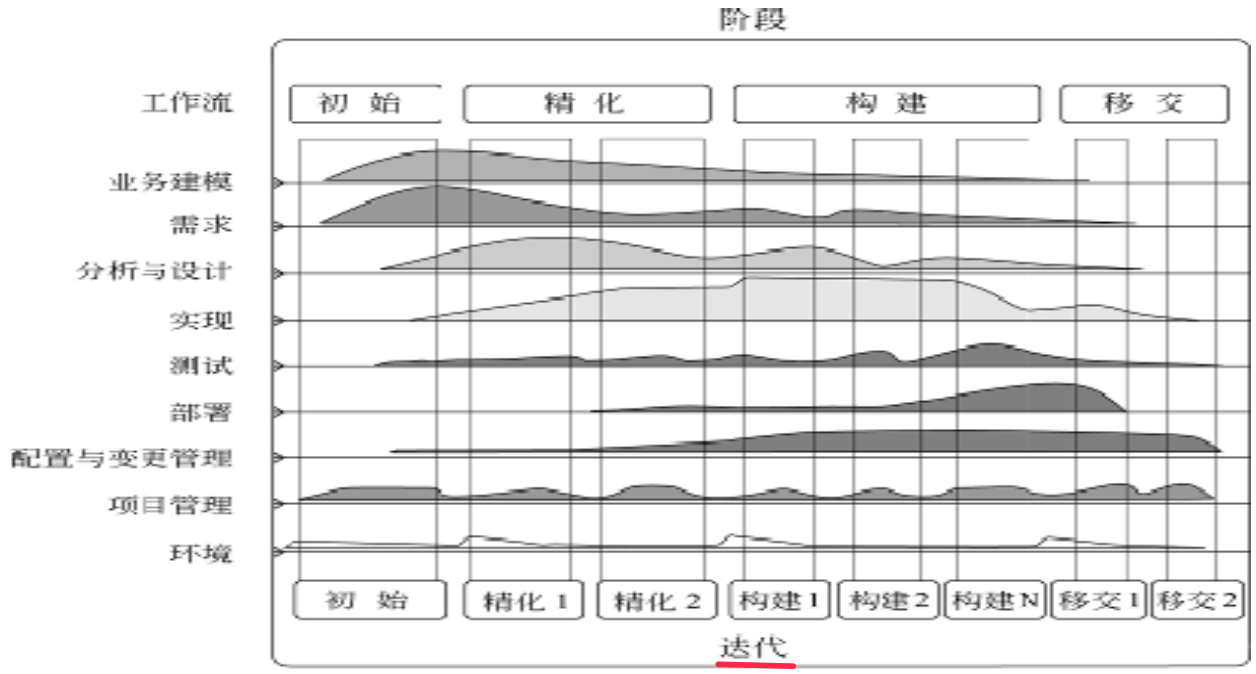
Rational统一过程充分体现了下述6条经过多年实践检验的软件开发经验：

- 采用迭代方式开发软件
- 在软件开发的全过程中有效地管理需求
- 采用基于构件的软件体系结构
- 建立软件产品的可视化模型
- 在软件开发的全过程中严格地验证软件的质量
- 控制软件变更

6. Rational统一过程

（2）RUP软件开发生命周期

是一个二维的软件生命周期模型，纵轴代表核心工作流，横轴代表时间。



（2）RUP软件开发生命周期

- 核心工作流：RUP有9个核心工作流，其中前6个为核心过程工作流(业务建模、需求、分析与设计、实现、测试、部署)，后3个为核心支持工作流（配置与变更管理、项目管理、环境）。
- 工作阶段：分成4个连续的阶段，分别为初始阶段、精化阶段、构建阶段和移交阶段。
- RUP强调采用迭代和渐增的方式进行软件开发，整个项目开发过程由多个迭代过程组成。

7. 敏捷过程与极限编程

（1）敏捷过程

根据下述4个价值观提出的软件过程统称为敏捷过程：

- 开发人员的素质及相互间的交互与协作比过程和工具更重要
- 可以工作的软件比面面俱到的文档更重要
- 与客户的合作比合同谈判更重要
- 及时响应变化比死板地遵守计划更重要

强调软件开发团队具有高效工作和快速响应变化的能力。

7. 敏捷过程与极限编程

（2）极限编程(eXtreme Programming，XP)

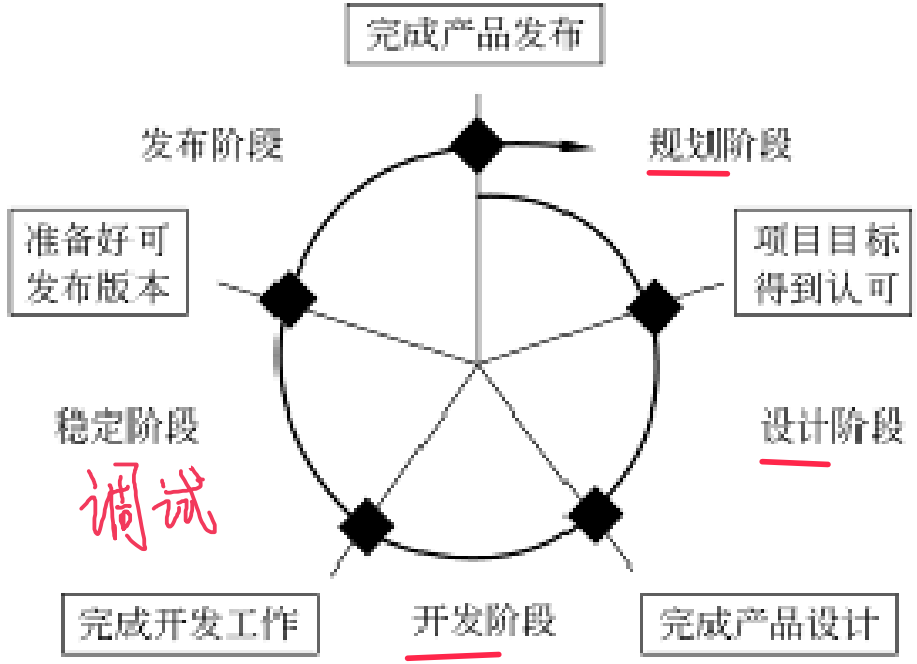
极限编程是敏捷过程中最富盛名的一个软件过程，其名字中的”极限”的含义是指把有效的软件开发实践运用到极致。极限编程适应于需求模糊且经常改变的场合。

极限编程具有对变化和不确定性的更快速、更敏捷的反应特性，能够较好地适应商业竞争环境下对小项目提出的有限资源和有限开发时间的约束。

8. 微软过程

（1）微软软件生命周期

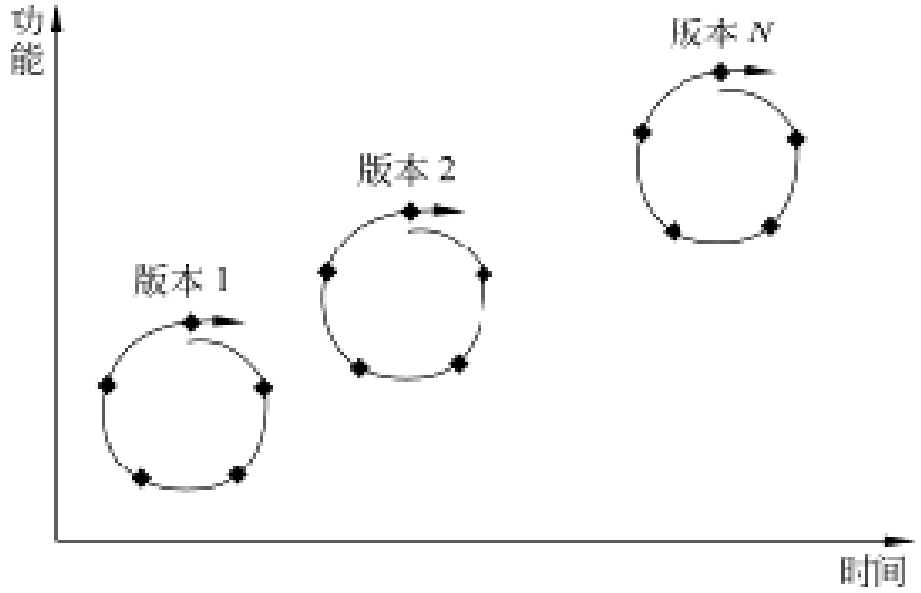
微软软件生命周期阶段划分和主要里程碑



8. 微软过程

（2）微软过程模型

每个生命周期发布一个递进的版本，各生命周期持续快速地迭代循环；综合了Rational统一过程和敏捷过程的许多优点。




Thank
You