课时1 软件工程学概述

1.1 软件与软件危机

1.2 软件工程

1.3 软件生命周期

1.4 软件过程

课时1 软件工程学概述

考点	重要程度	占分	题型
1.1软件与软件危机	★★★	4~8	选择、填空、简答
1.2软件工程	★★★	4~8	填空、简答
1.3软件生命周期	★★★★	6~8	填空、简答
1.4软件过程	★★★	4~8	选择、填空、简答

1.1软件与软件危机

一：软件的概念、特点与发展

(1) 软件发展经历三个阶段：

程序设计阶段	50至60年代
程序系统阶段	60至70年代
软件工程阶段	70年代以后

(2) 软件的概念：

软件是计算机系统中与硬件相互依存的另一部分，它包括**程序**、**数据**及其相关**文档**的完整集合。(软件=程序+数据+文档)

数据:是使程序能够适当处理信息的数据结构。

程序:是能够完成预定功能和性能的可执行指令序列。

文档:是开发、使用和维护过程程序所需要的图文资料。

1.1软件与软件危机

一：软件的概念、特点与发展

(3) 软件的特点：

1.软件本身的复杂性

2.软件的成本高昂

3.软件开发未摆脱手工开发方式

4.软件维护与硬件有本质差，维护难度高

5.软件开发不是传统硬件制造过程

6.软件是一种逻辑实体，无磨损性

二：软件危机

(1) 软件危机的概念：

在计算机软件开发和维护过程中所遇到的一系列严重问题。

软件危机包含两方面内容：

1、如何开发软件，以满足对软件日益增长的需求

2、如何维护数量不断膨胀的已有软件

二：软件危机

(2) 软件危机的表现:

- 对软件开发成本和进度估算不准确
- 用户对已完成软件不满意
- 软件质量不可靠
- 软件不可维护
- 没有适当文档资料
- 软件成本在计算机系统中所占比例逐年上升
- 软件开发生产率低

视频讲解更清晰
仅3小时

1.1软件与软件危机

(3) 软件危机的原因:

1、主观原因:

忽视需求分析

轻视软件维护

没有认识到程序只是软件的一部分

没有认识到软件开发只是漫长的软件生命周期中一个比较次要阶段

越到后期引入变动付出代价越高昂

2、客观原因:

软件是逻辑实体、缺乏可见性，管理和控制困难

软件不会磨损，维护意味着修改原来设计，维护困难

软件规模庞大，程序复杂性随规模增加指数上升

二：软件危机

(4) 消除软件危机的途径:

1、对计算机软件应该有正确认识

2、吸取借鉴人类长期从事各种工程项目积累的原理、概念、技术和方法

3、积极开发和使用计算机辅助开发工具

4、探索更好更有效的管理措施和手段对开发过程进行控制和管理

管理+技术　　------软件工程出现来源于软件危机

1.2 软件工程

(1) 软件工程定义：

采用**工程**的概念、原理、技术和方法来开发与维护软件，把经过时间考验而证明正确的**管理技术**和当前能够得到的最好的**技术方法**结合起来，**经济**的开发出**高质量**的软件并**维护**它。

1.2 软件工程

(2) 软件工程的本质特性：

- 关注大型程序的构造
- 中心课题是控制复杂性
- 软件经常变化
- 开发效率非常重要
- 开发人员和谐合作是关键
- 软件需有效支持用户
- 软件开发者替代其他领域人员创造产品

视频讲解更清晰
仅3小时

1.2 软件工程

(3) 软件工程基本原理：

- 按软件生存期分阶段制定计划并认真实施
- 坚持进行阶段评审
- 坚持严格的产品控制
- 使用现代程序设计技术
- 结果能够得到清楚的审查
- 用人少而精
- 承认不断改进软件工程实践的必要性

1.2 软件工程

(4) 软件工程方法学：

把在软件生命周期全过程中使用的**一整套技术方法的集合**称为方法学，也称为泛型。

软件工程方法学包括三个要素：**方法**、**工具**和**过程。**

方法:完成软件开发各项任务的技术方法，回答“怎么做”

工具:为运用方法提供的自动或半自动软件工程支撑环境

过程:是为了获得高质量软件所需要完成的一系列任务框架，回答“何时做”

1.2 软件工程

(5) 软件工程方法学分类：

传统方法学(生命周期方法学)：

- 采用结构化技术完成软件开发各项任务
- 把软件生命周期的全过程依次划分为若干阶段
- 每个阶段开始和结束有严格标准
- 每个阶段结束后进行严格审查

视频讲解更清晰
仅3小时

面向对象方法学:

- 把对象作为融合了数据及在数据上的操作行为的统一的软件构件
- 把所有对象划分为类
- 按照父类与子类关系，把若干类组成层次结构的系统
- 对象彼此间仅能通过发送消息互相联系

1.3 软件生命周期

软件生命周期

- **软件定义**
  - **问题定义：** 弄清用户要解决什么问题
  - **可行性研究：**上阶段确定问题是否可行
  - **需求分析：** 为解决这个问题，系统需要具备什么功能
- **软件开发**
  - **总体设计：** 设计软件结构，确定程序由哪些模块组成以及模块间的关系
  - **详细设计：** 争对每个模块，设计详细规格说明，确定算法和数据结构
  - **编码和单元测试：**将详细设计内容用语言实现，并测试每个模块
  - **综合测试：** 通过各种类型测试使软件达到预定要求
- **软件维护**
  - **运行维护：** 使软件在整个生命周期内保证满足用户需求

1.4 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框架。

通常用软件生命周期模型描述软件过程。

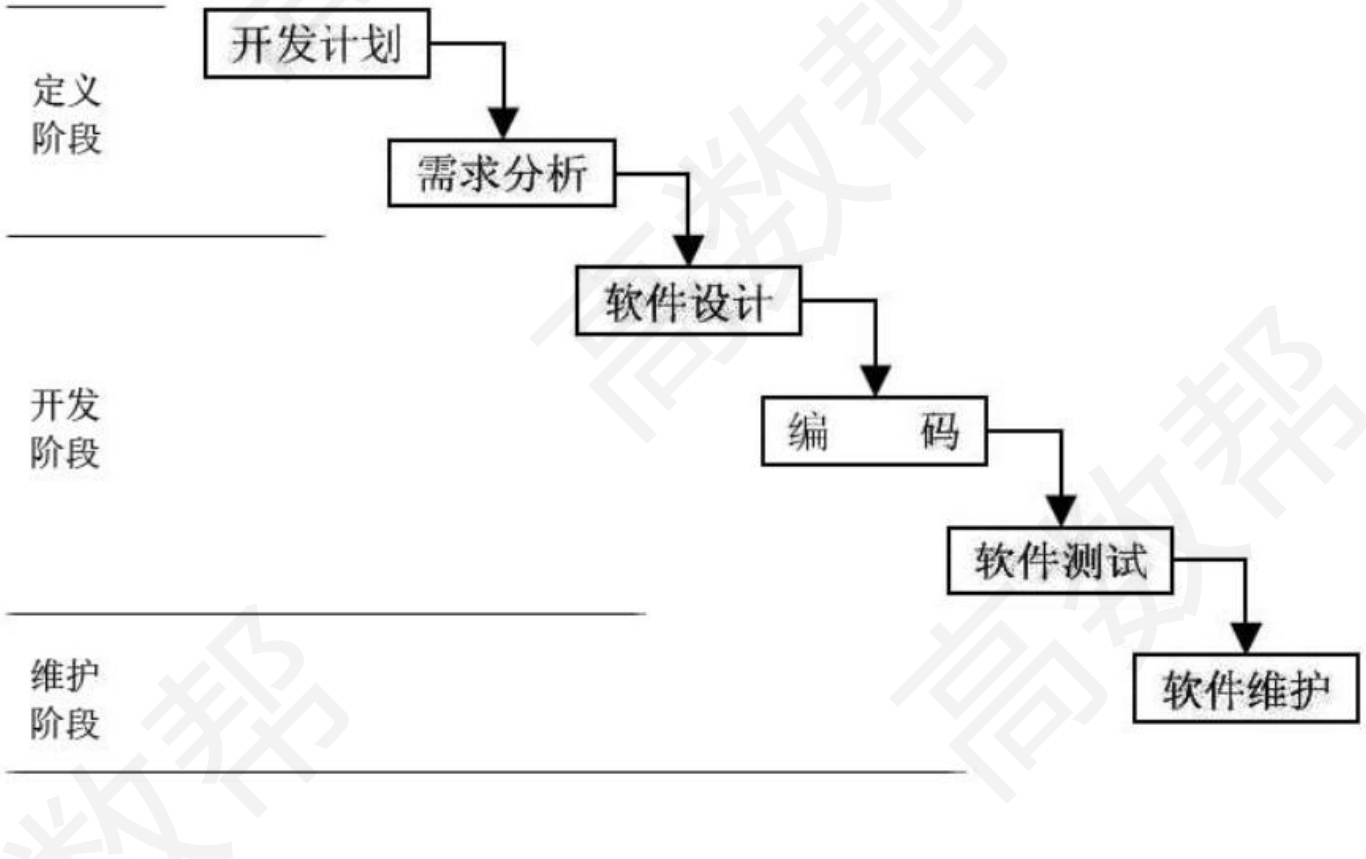
主要包括:

- 瀑布模型
- 快速原型模型
- 增量模型
- 螺旋模型
- 喷泉模型
- 其他模型

1.4软件过程

瀑布模型:

将软件生存周期的各项活动规定为依照固定顺序连接的若干阶段工作，最终得到软件产品。



1.4软件过程

瀑布模型特点:

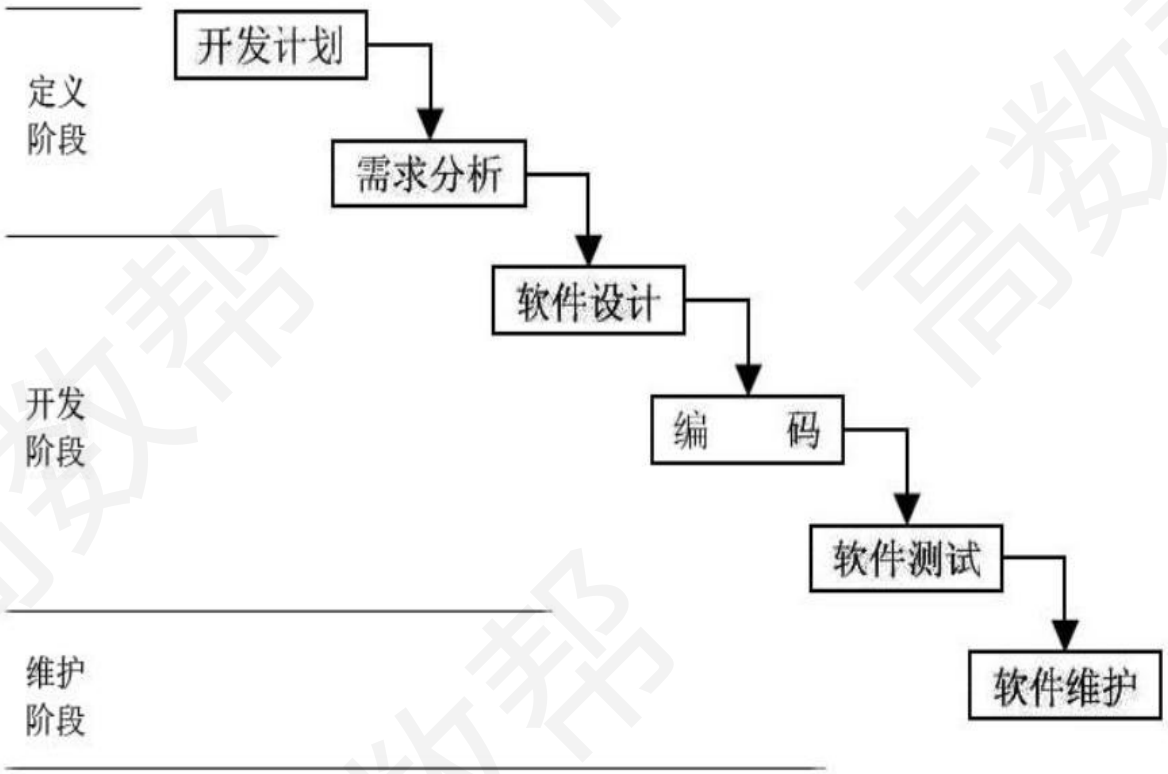
1.阶段间具有顺序性和依赖性。

2.推迟实现的观点。

3.质量保证的观点

每个阶段必须完成规定的文档;

每个阶段结束前完成文档审查,及早改正错误。



1.4软件过程

瀑布模型优缺点:

1、瀑布模型的优点（强迫开发人员使用规范的方法，严格规定了每个阶段必须提交的文档，要求每个阶段）

可以强迫开发人员采用规范的方法;

严格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。

2、瀑布模型的缺点

在软件开发的初期阶段就要求做出正确、 全面、完整的需求分析对许多应用软件来说是极其困难的。

在需求分析阶段，当需求确定后，无法及时验证需求是否正确、完整。

作为整体开发的瀑布模型，由于不支持产品的演化，缺乏灵活性，对开发过程中很难发现的错误，只有在最终产品运行时才能暴露出来， 从而使软件产品难以维护。

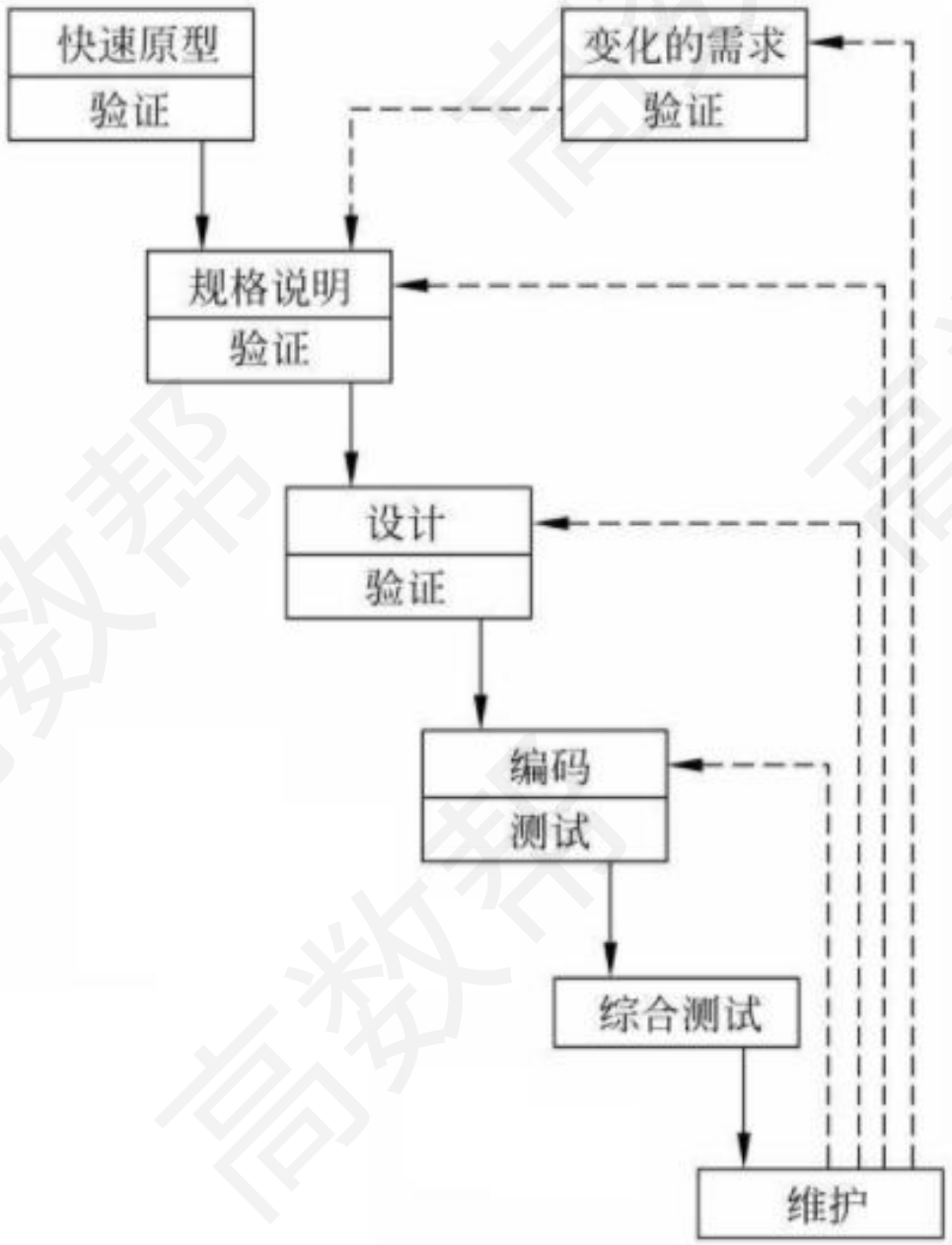
1.4软件过程

改进的瀑布模型:

1.4软件过程

快速原型模型:

快速建立可运行的程序，它完成的功能往往是最终产品功能的一个子集。



1.4软件过程

快速原型模型优缺点:

优点:

1、开发的软件产品通常满足用户需求

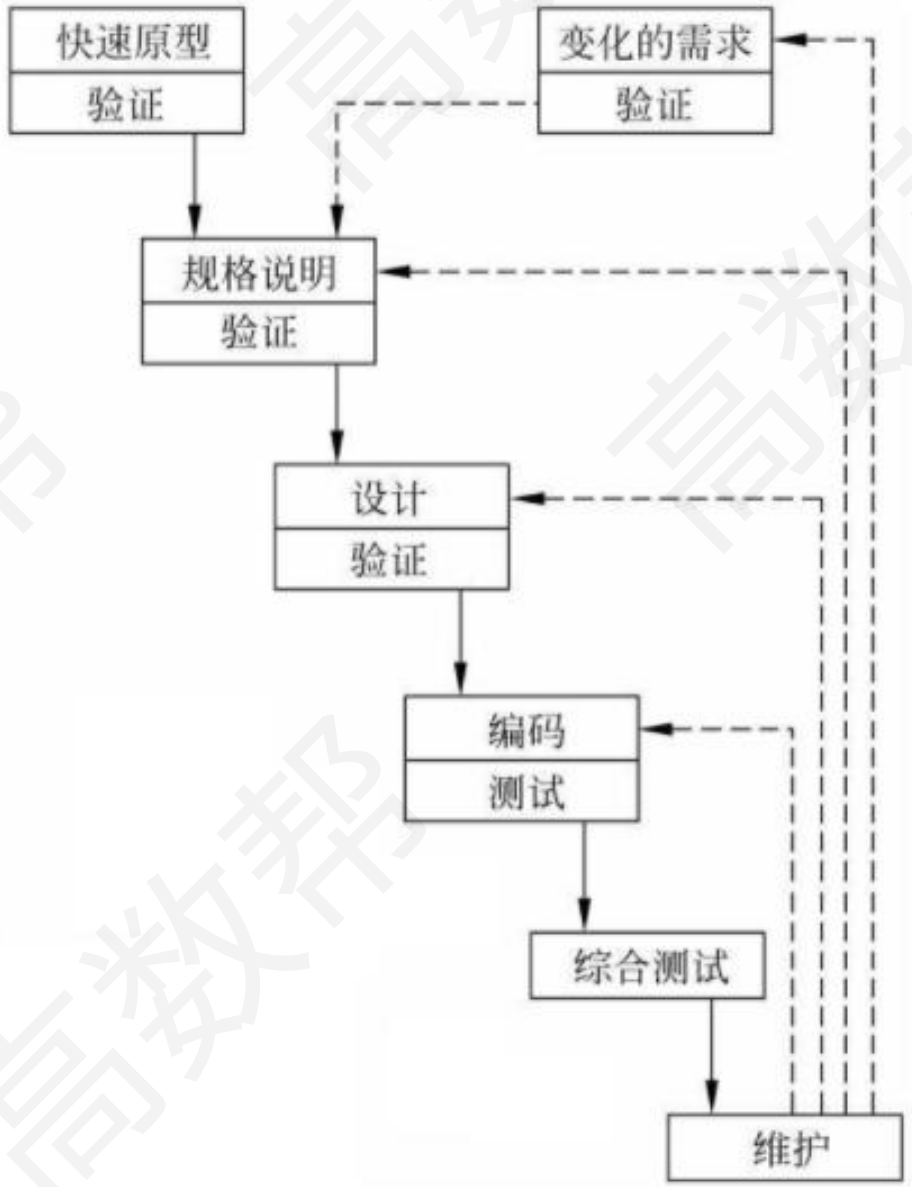
2、软件产品开发基本是线性过程

缺点:

1、准确原型设计困难

2、原型理解可能不同

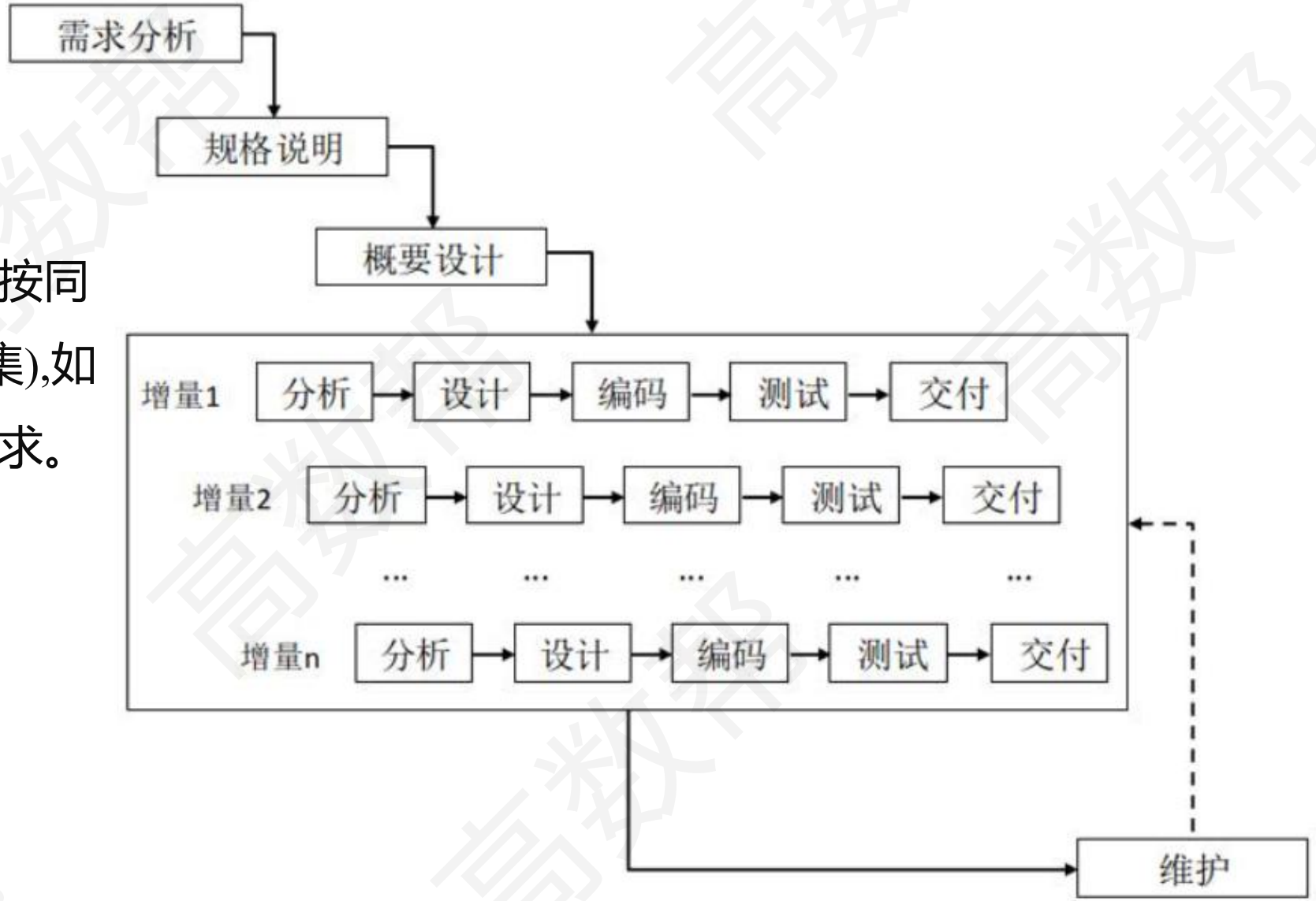
3、不利于开发人员创新



1.4软件过程

增量模型:

先完成一个系统子集的开发，再按同样的开发步骤增加功能 (系统子集),如此递增下去直至满足全部系统需求。



1.4软件过程

增量模型优缺点:

优点:

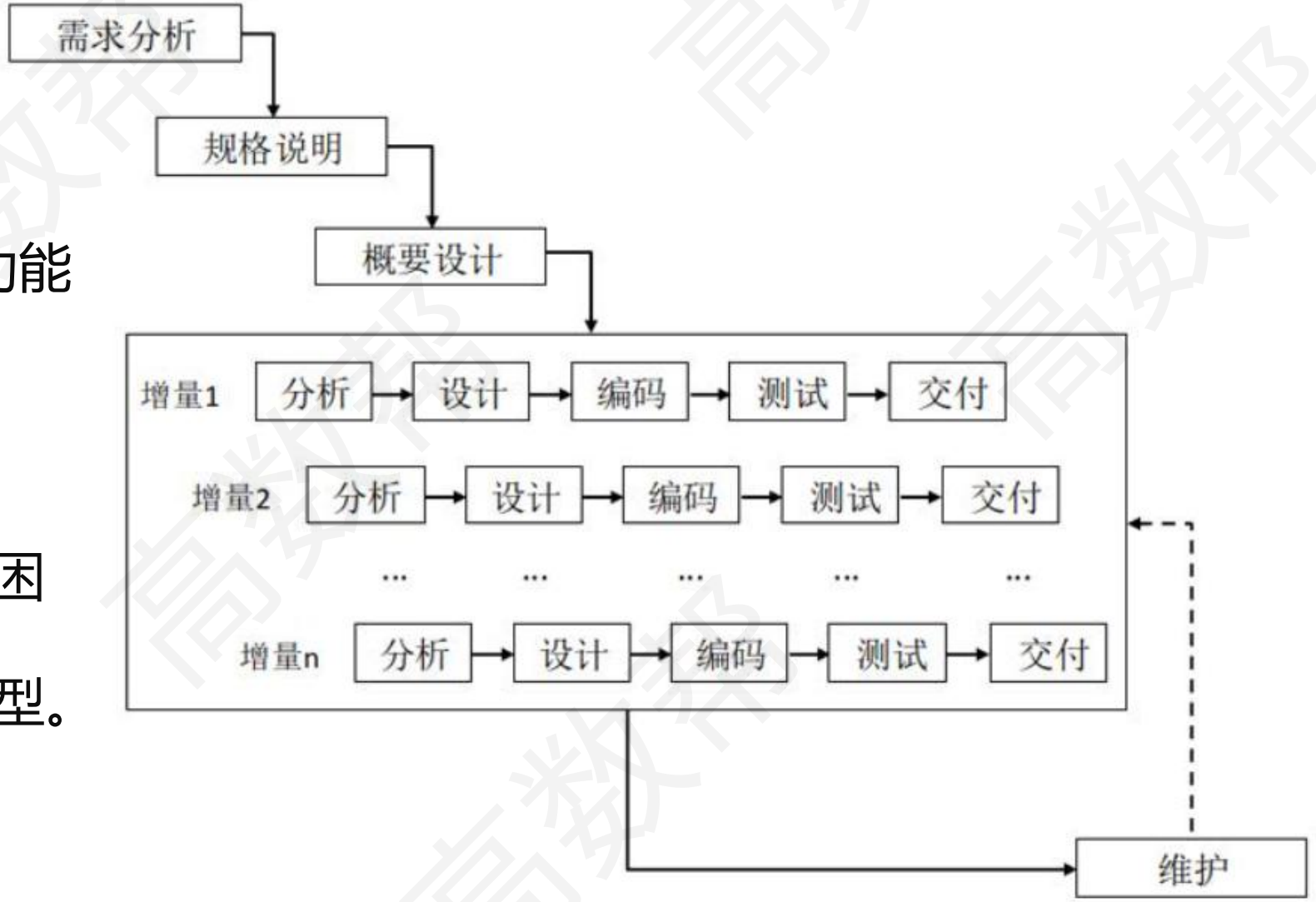
1、短时间内可提交完成部分功能

2、逐渐增加产品功能，
用户适应产品快。

缺点:

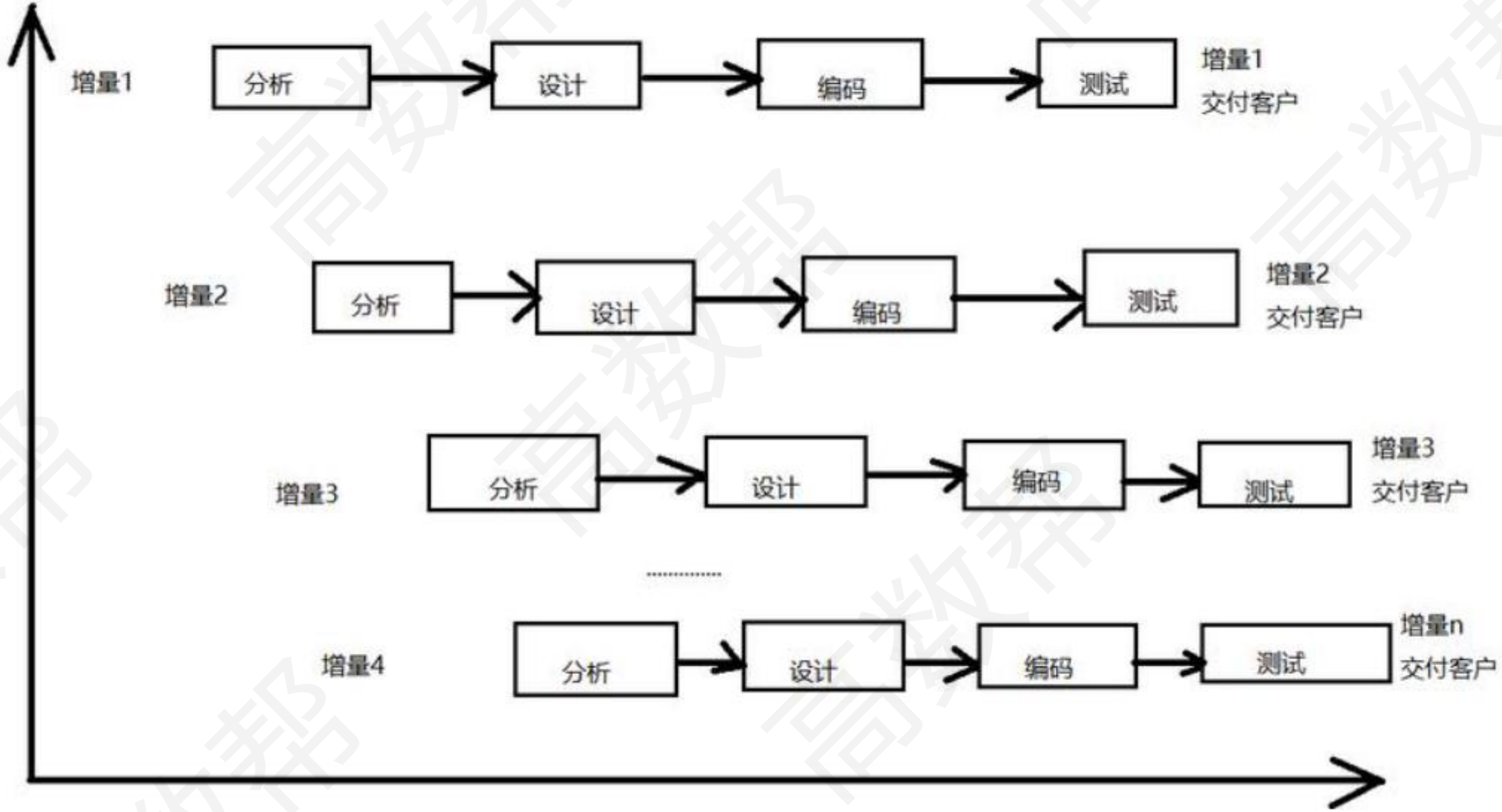
1、增量构件划分以及集成困难。

2、容易退化为边做边改模型。



1.4软件过程

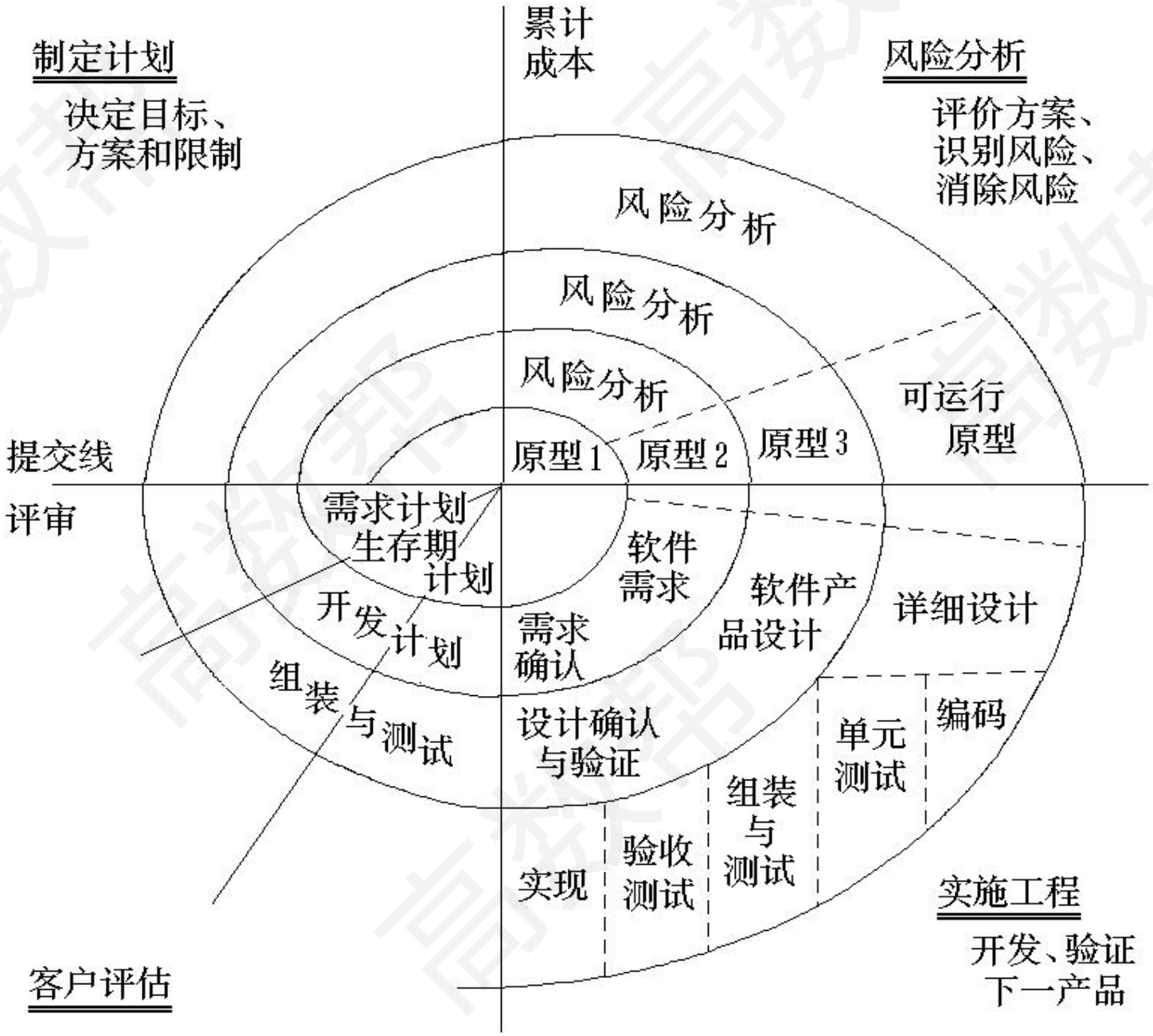
风险更大增量模型:



1.4软件过程

螺旋模型:

在每个阶段之前都增加了风险分析过程的快速原型模型。看作增加了**风险分析**的**快速原型模型。**



1.4软件过程

螺旋模型优缺点:

优点:

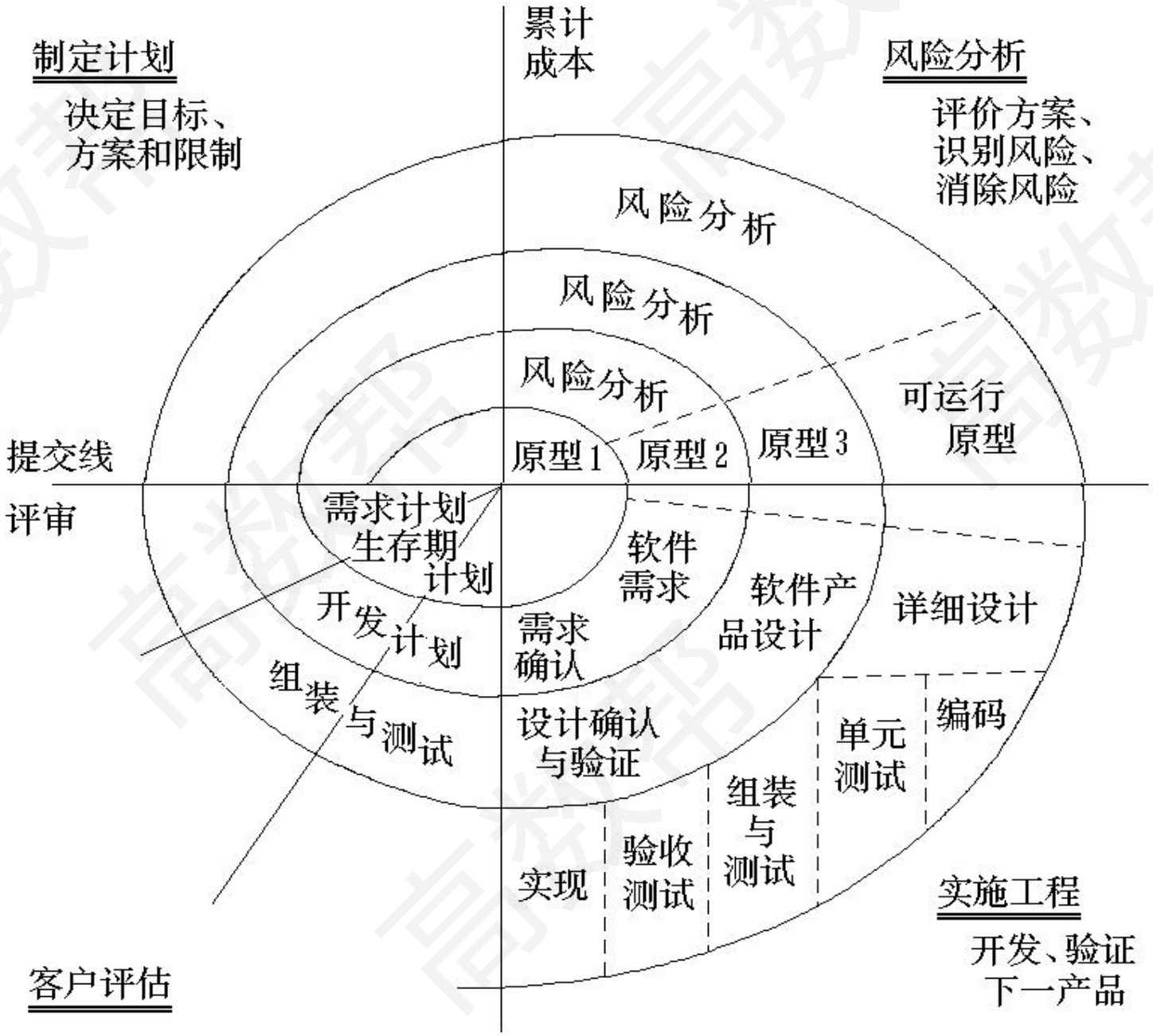
1、利于把软件质量作为软件开发目标。

2、减少测试

3、维护和开发不分开

缺点:

1、风险估计困难

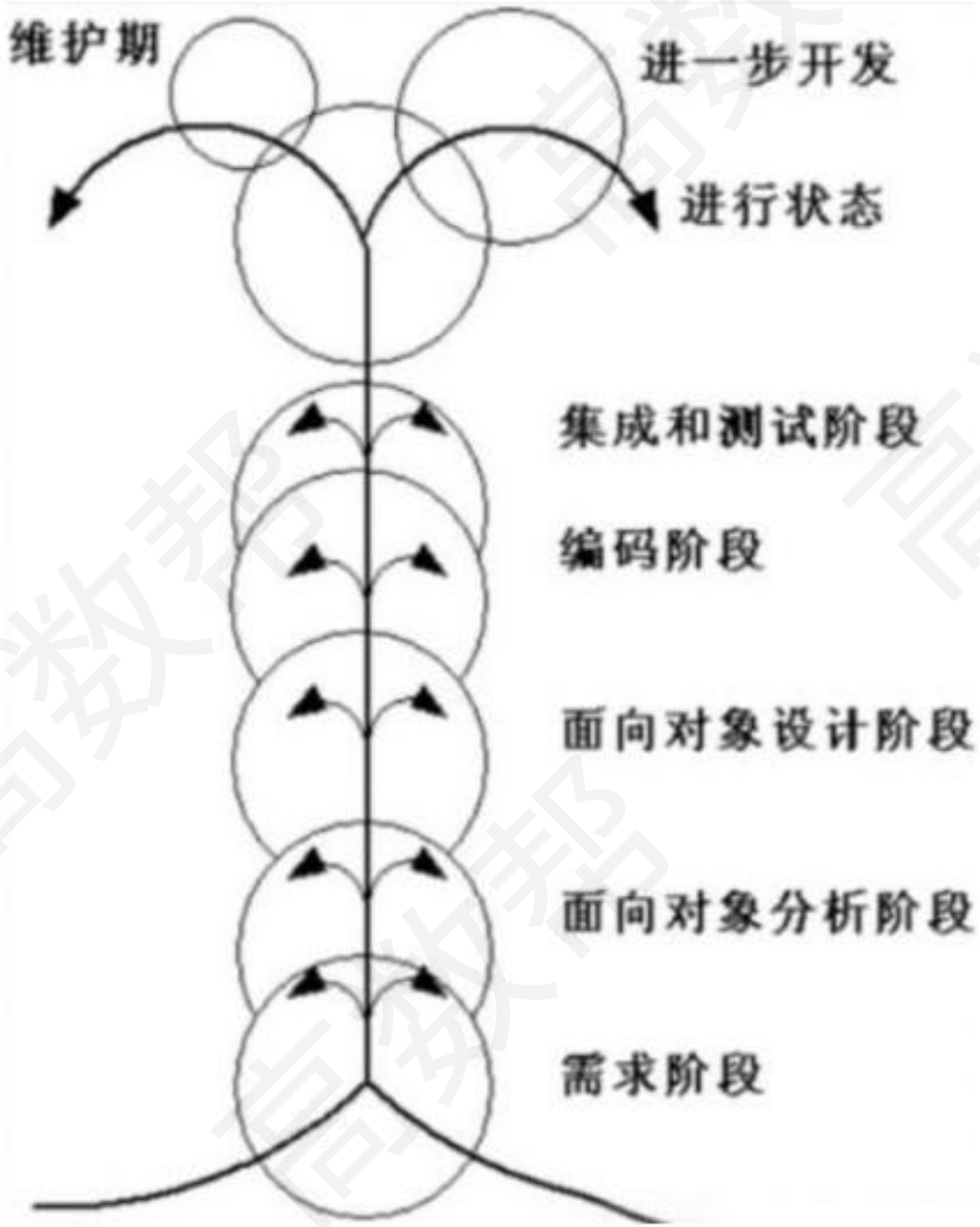


1.4软件过程

喷泉模型:

典型的面向对象软件过程模型。

体现迭代和无缝的特性。



1.4软件过程

其他模型:

- Rational统一过程
- 敏捷过程与极限编程
- 微软过程

视频讲解更清晰
仅3小时

总结

- 掌握软件、软件工程、软件危机相关概念
- 掌握软件生命周期各阶段及其任务
- 掌握软件开发过程：

  瀑布模型

  快速原型

  增量模型

  螺旋模型

  喷泉模型

视频讲解更清晰
仅3小时