软件工程

Software Engineering

张宏国

哈尔滨理工大学 计算机科学与技术学院

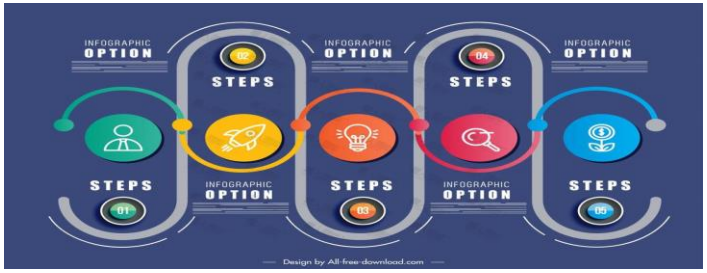
2024年8月

2.1 软件过程概述

软件工程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。

ISO9000的软件工程过程定义

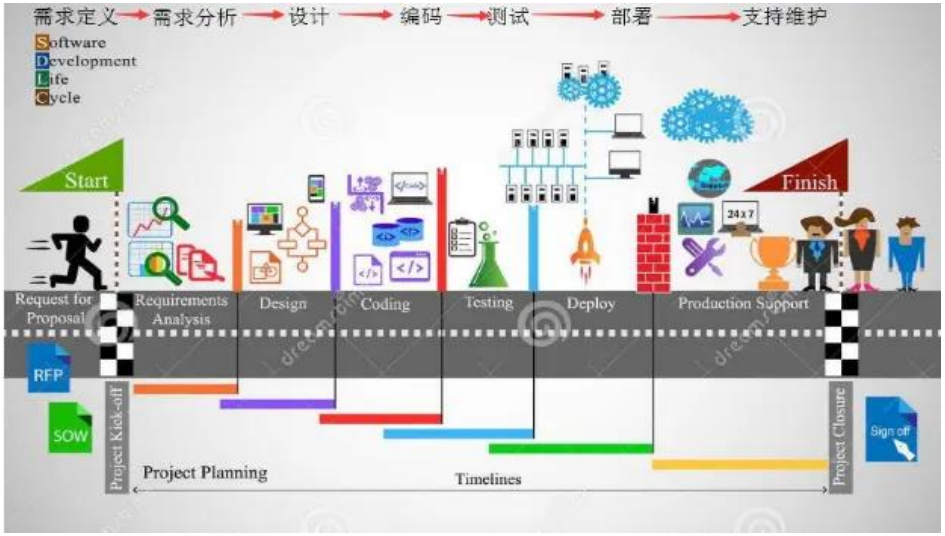
软件工程过程是：把输入转化为输出的一组彼此相关的资源和活动。



2.2 软件生命周期

2.2.1 软件生命周期的概念

软件产品的生命周期是指从设计产品的构想开始，到软件需求的确定、软件设计、软件实现、产品测试与验收、投入使用以及产品的不断维护与更新，到最终该产品被市场淘汰的全过程。



2.2 软件生命周期

2.2.2 传统软件生命周期的各个阶段

软件定义时期

* 确定软件开发工程必须完成的总目标
* 确定工程的可行性
* 导出实现工程目标应采取的策略及系统必须完成的功能
* 估计完成该项工程需要的资源和成本并且制定工程进度表

软件开发时期

* 具体设计和实现在前一时期定义的软件

软件维护时期

* 使软件持久地满足用户的需要

2.2 软件生命周期

2.2.1 软件生命周期的概念

软件定义时期

问题定义	可行性研究	需求分析
* 回答“解决的问题是什么”。	* 回答“上一个阶段所确定的问题是否有行得通的解决办法”	* 回答“目标系统必须做什么”
* 提出关于问题的性质、工程目标和工程规模的书面报告，并且要得到客户的认可。	* 从技术、经济、社会因素等方面分析可行性，最终提供可行性研究工作报告	* 系统分析员与用户共同确定系统必须完成的工作，对目标系统提出完整、准确、清晰、具体的要求，提交规格说明书。

2.2 软件生命周期

2.2.1 软件生命周期的概念

软件开发时期

概要设计	详细设计	编码和单元测试	综合测试
回答“怎样实现目标系统”。	回答“怎样具体地实现目标系统”。	写出正确的易理解、易维护程序模块	通过各类测试(调试)使软件达到预定的要求
设计出实现目标系统的多种方案。 设计软件的体系结构。 提交概要设计报告	详细设计每个模块，确定实现模块功能所需要的算法和数据结构。 提交详细设计报告	根据目标系统的性质和环境选取编程语言。 把详细设计结果翻译成程序，并进行每一模块的测试。	进行集成测试 进行验收测试 提交测试计划、测试方案、实际测试结果文档。

2.2 软件生命周期

2.2.1 软件生命周期的概念

软件定义时期 → 软件开发时期 → 软件维护时期

通过各种必要的维护活动使系统持久地满足用户的需要

四类维护活动：
改正性维护：诊断修改使用中发现的错误
适应性维护：修改软件适应环境变化
完善性维护：改进完或扩充软件使其完善
预防性维护：修改软件为未来维护做准备
每一项维护活动必须准确记录下来，作为正式的文档加以保存。

2.3 软件过程模型

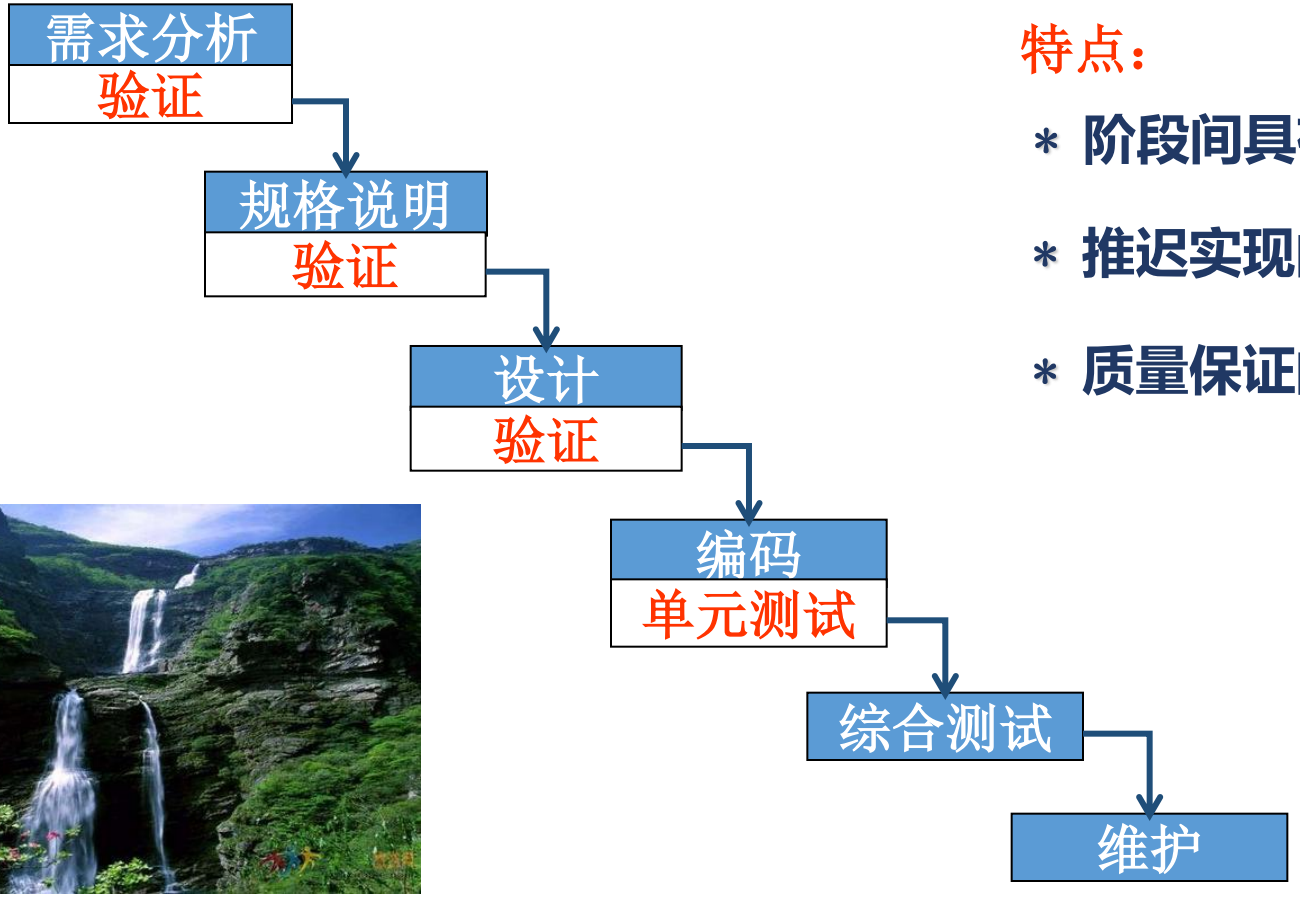
在软件工程中，人们通过建立抽象的软件过程模型，把软件生命周期中的各个活动或步骤安排到一个框架中，将软件开发的全过程清晰且直观地表达出来。（简单地说，对生命周期中的各项活动或步骤进行的不同组织方式，形成不同的过程模型）。

软件过程模型的特点：

1. 描述了主要的开发阶段
2. 定义了每个阶段要完成的主要任务和活动
3. 规范了每个阶段的输入和输出
4. 提供了一个框架，把必要的活动映射到这个框架中

2.3.1 瀑布模型

1970年温斯顿·罗伊斯（Winston Royce）提出瀑布模型（waterfall model），在20世纪80年代之前，一直是唯一被广泛采用的生命周期模型，目前仍是软件工程应用中应用最广泛的过程模型。

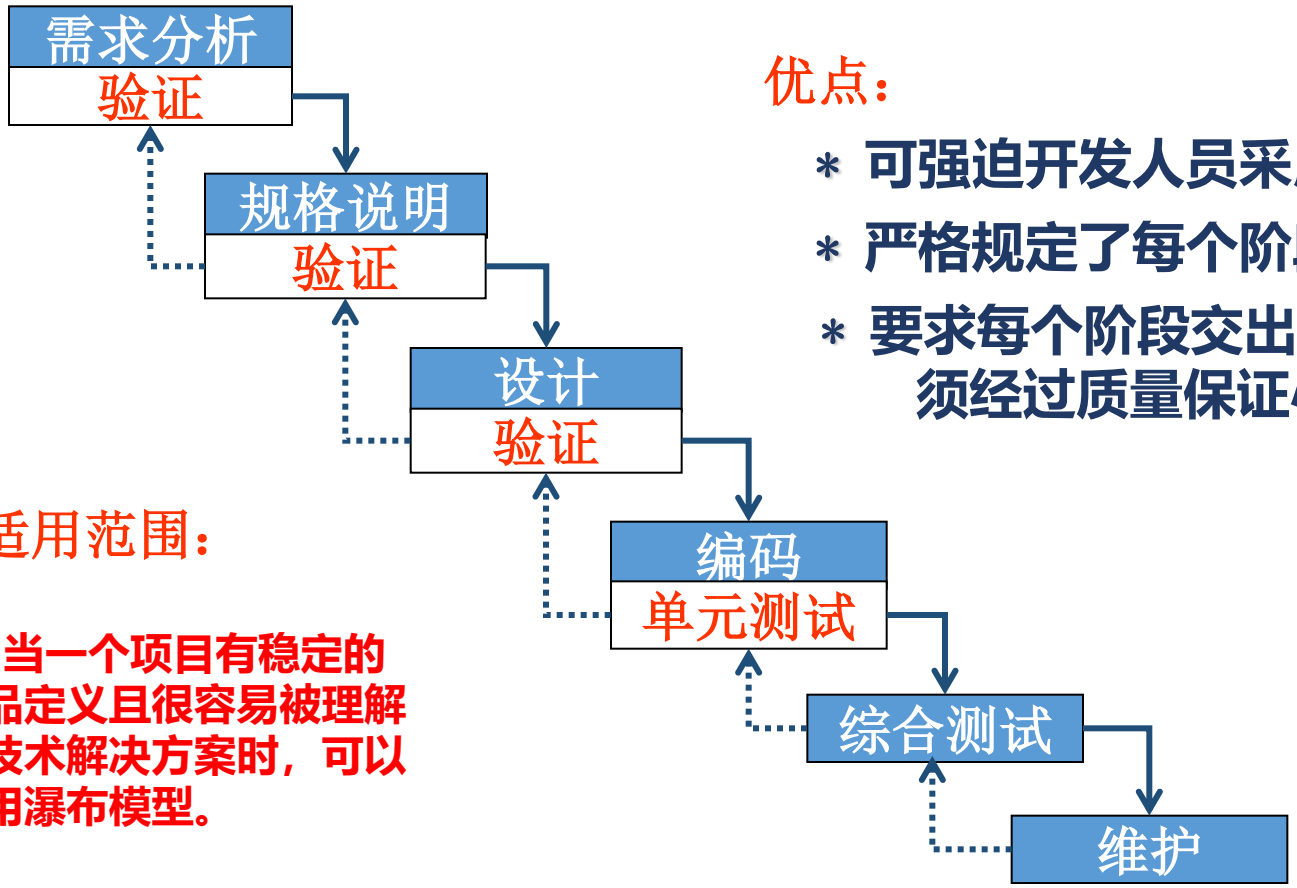


特点：

* 阶段间具有顺序型和依赖性
* 推迟实现的观点
* 质量保证的观点

2.3.1 瀑布模型

传统的瀑布模型过于理想化了。事实上，人在工作过程中不可能不犯错误。为此，在传统瀑布模型的基础上，加入迭代过程。



优点：

* 可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

适用范围：

当一个项目有稳定的产品定义且很容易被理解的技术解决方案时，可以使用瀑布模型。

缺点：

1. “文档驱动”只能通过静态的文档来认识动态的软件产品，很可能导致最终开发出的软件产品不能真正满足用户的需要。
2. 回溯性差。

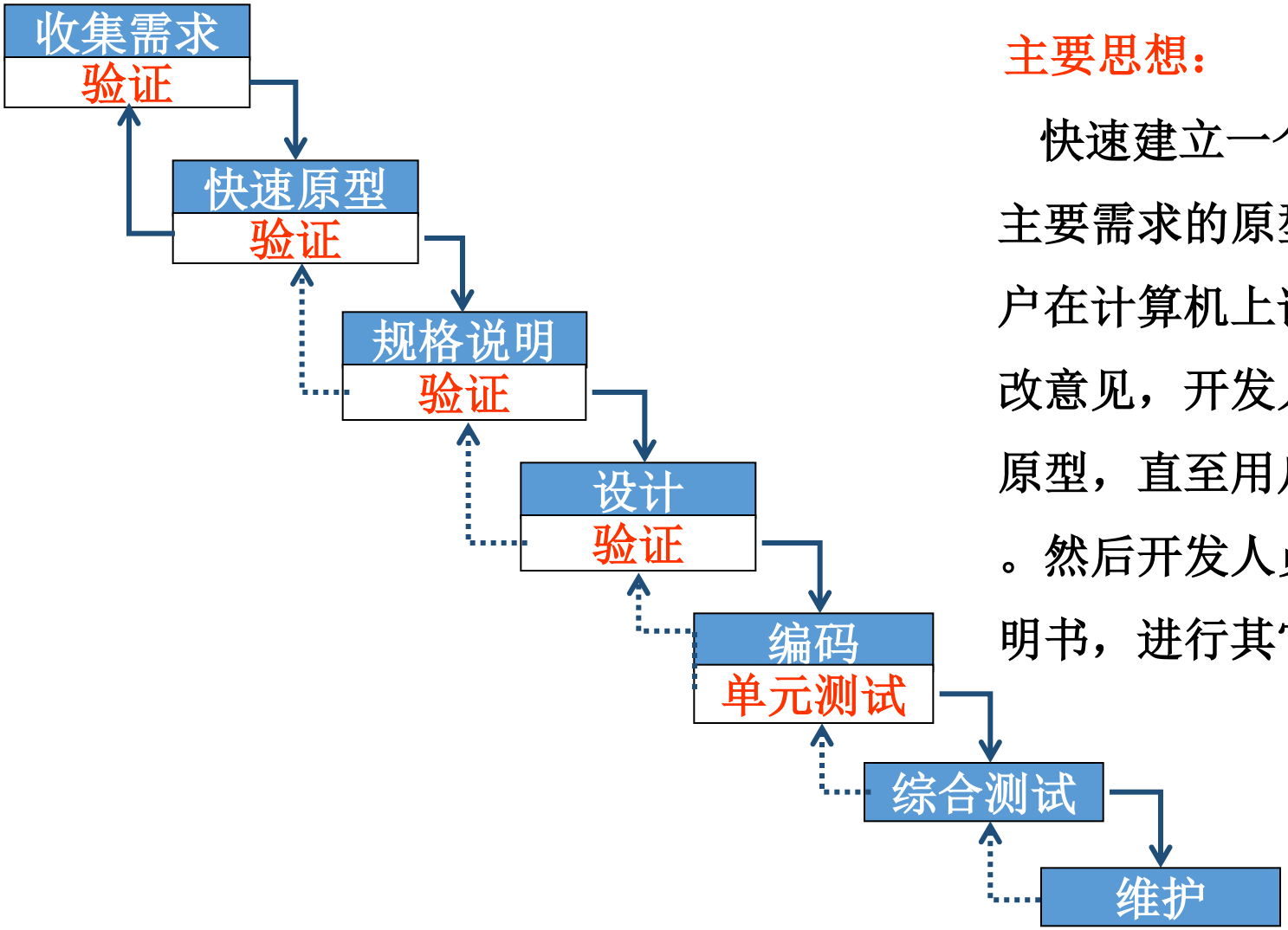
2.3.2 快速原型模型

当一个项目有稳定的产品定义且很容易被理解的技术解决方案时，可以使用瀑布模型。

当一个项目不能给出完整准确定义，对模型进行怎样的改进呢？

人们为了解决这个问题，引入了快速原型（rapid prototype）模型，通过快速建立反映用户主要需求的原型系统，让用户在计算机上试用，不断提出修改意见，不断完善原型系统，从而得到真实需求。

2.3.2 快速原型模型



主要思想：

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，提出修改意见，开发人员快速修改原型，直至用户无意见为止。然后开发人员书写规格说明书，进行其它阶段工作。

2.3.2 快速原型模型

优点：

软件产品的开发基本上是线性顺序进行的。

* 原型系统已经通过与用户交互而得到验证，使产生的规格说明书正确地描述了用户需求。

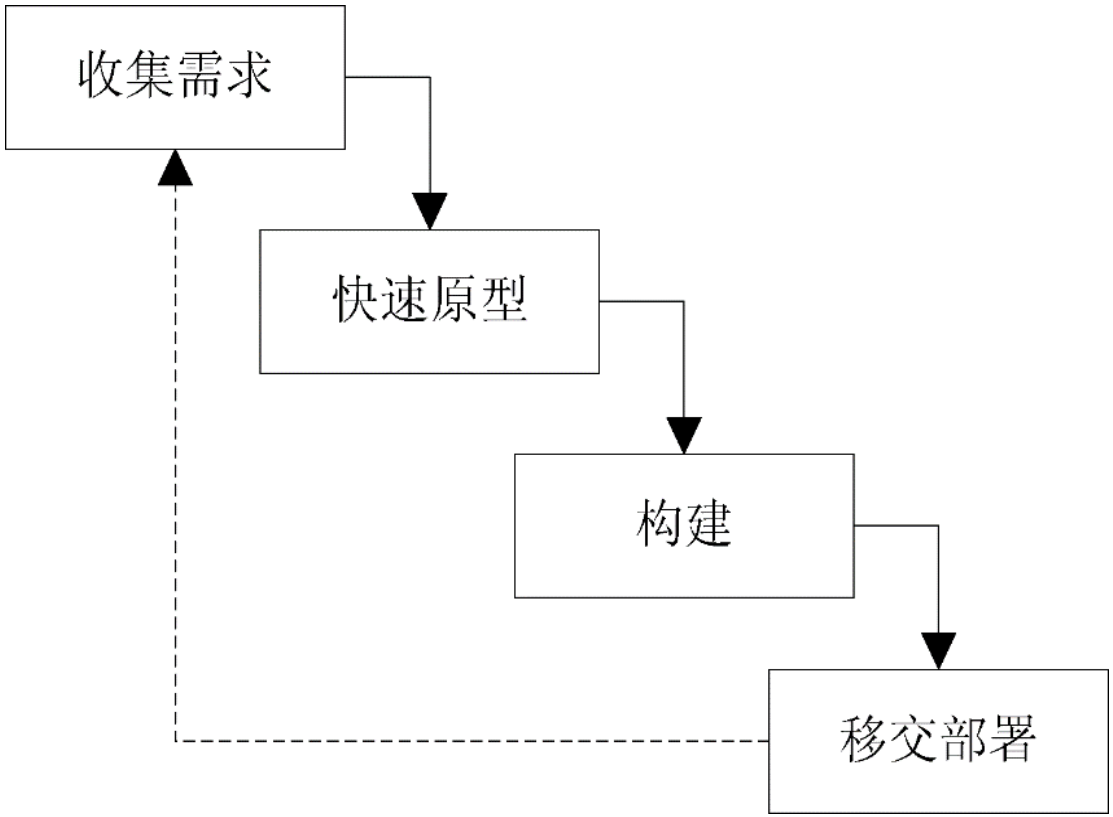
* 开发人员通过建立原型系统已经学到了许多东西，因此，在设计和编码阶段发生错误的可能性很小。

注意：

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。原型的用途是获知用户的真正需求，一旦需求确定了，原型将被抛弃。

2.3.2 快速原型模型

另一类快速原型



主要思想：

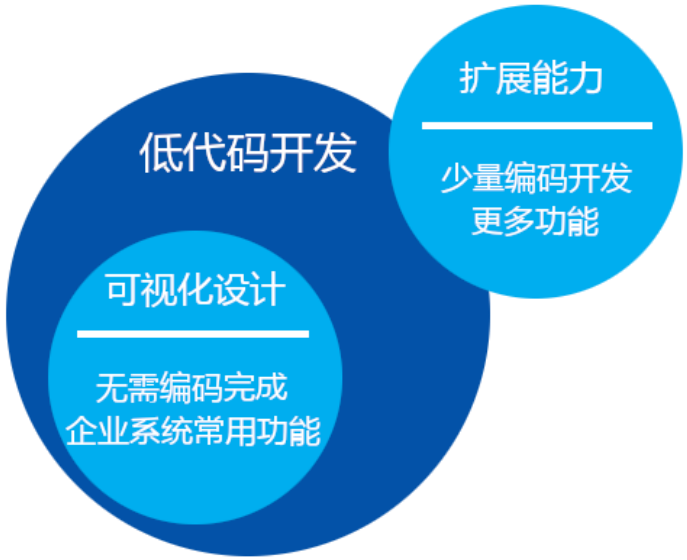
快速建立一个能反映用户主要需求的原型系统，让用户在计算机上试用它，通过实践来了解目标系统的概貌。通常，用户试用原型系统之后会提出许多修改意见，开发人员按照用户的意见快速地修改原型系统，然后再次请用户试用……反反复复地改进，直到原型系统满足用户的要求。

2.3.2 快速原型模型

另一类快速原型

该类快速原型模型适用于具有以下特征的软件开发项目：

1. **已有产品或产品的原型（样品），只需客户化的工程项目**
2. **简单而熟悉的行业或领域**
3. **有快速原型开发工具**
4. **进行产品移植或升级**



2.3 软件过程模型

2.3.3 增量模型

瀑布模型需要经过很长的周期最终才能产生可运行的软件，能否加以改进，尽早就能获得一个可运行的软件呢？

人们为了解决这个问题，提出了增量模型（Incremental process model），在需求可分段的情况下进行增量开发。

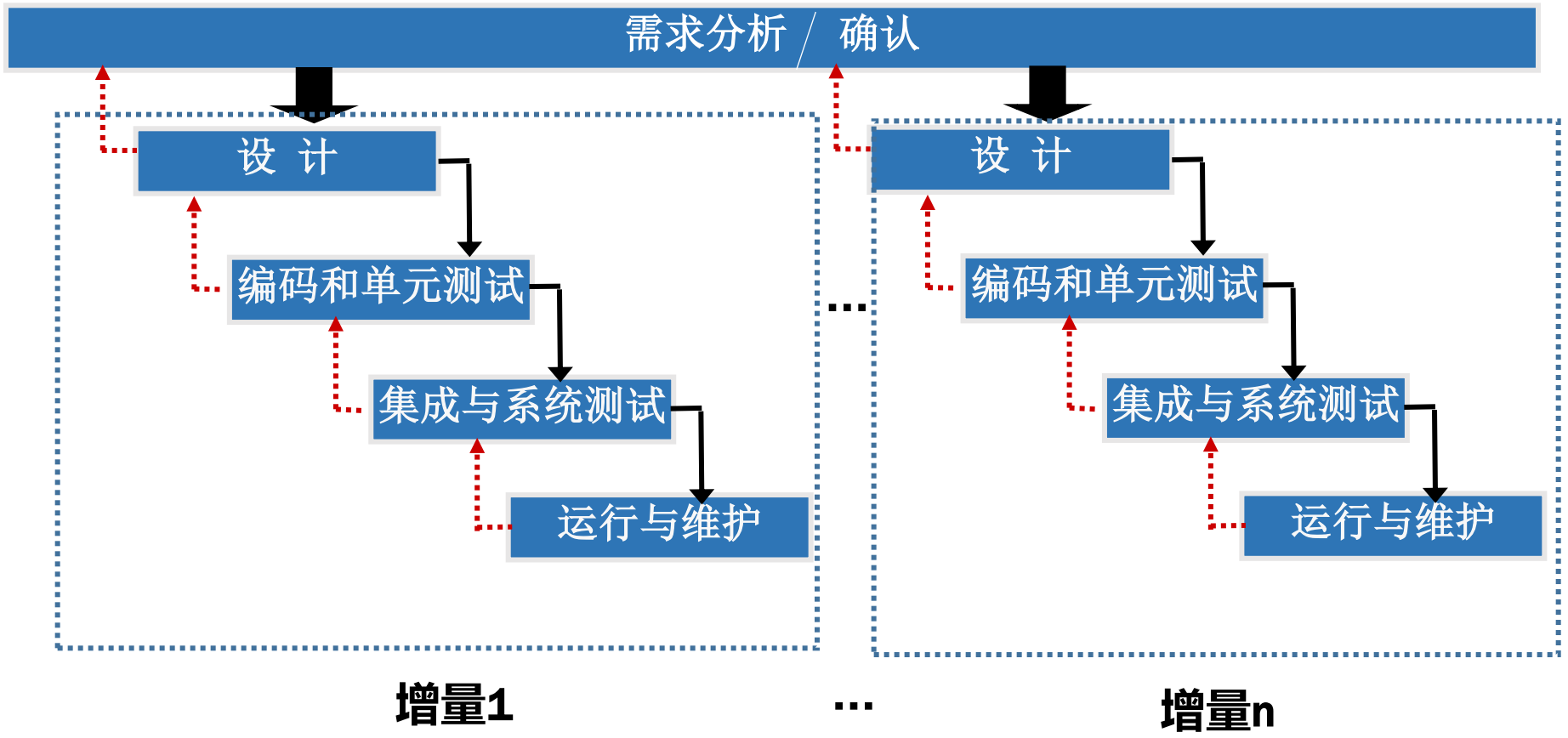
2.3 软件过程模型

2.3.3 增量模型

增量模型是把待开发的软件系统模块化，将每个模块作为一个增量组件，从而分批次地分析、设计、编码和测试这些增量组件。运用增量模型的软件开发过程是**递增式的过程**。相对于瀑布模型而言，采用增量模型进行开发，开发人员不需要一次性地把整个软件产品提交给用户，而是可以分批次进行提交。

2.3 软件过程模型

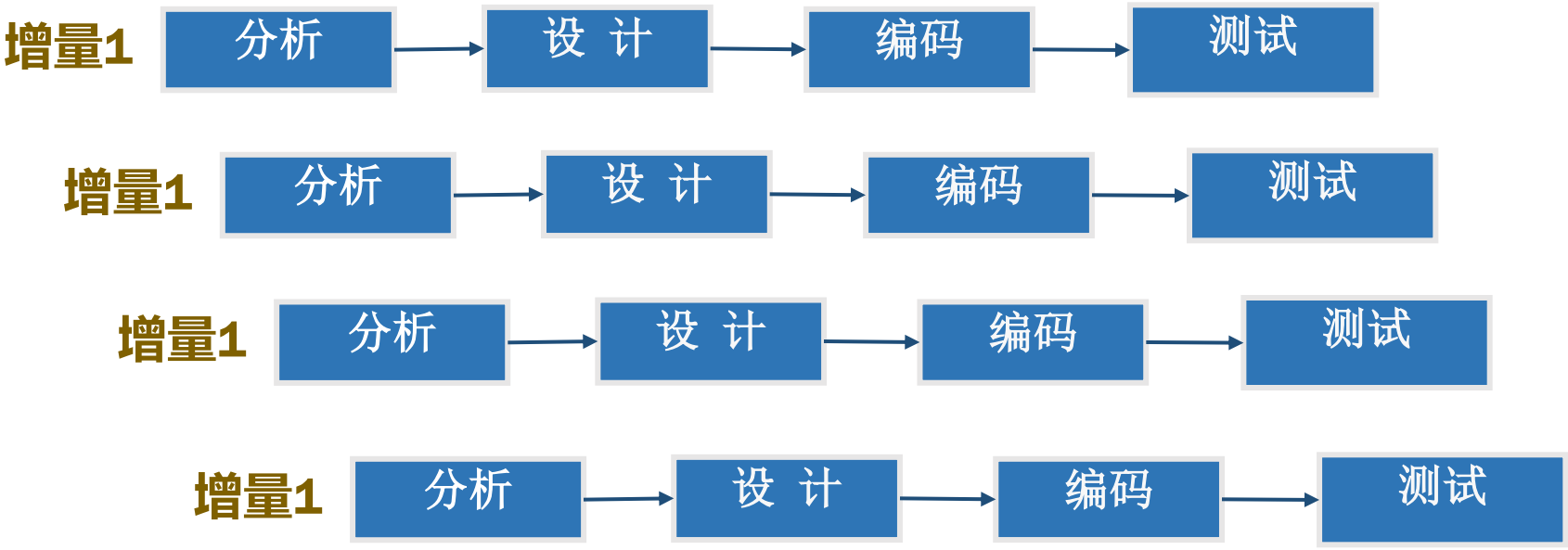
2.3.3 增量模型



渐增式增量模型

2.3 软件过程模型

2.3.3 增量模型



并行度更高的增量模型

2.3 软件过程模型

2.3.3 增量模型

增量模型作为瀑布模型的一个变体，具有瀑布模型的所有优点，此外，**它还有以下优点：**

- 将待开发的软件系统模块化，可以分批地提交软件产品，使用户可及时了解软件项目的进展。
- 以组件为单位进行开发降低了软件开发风险。一个开发周期内的错误不会影响整个软件系统。
- 开发顺序灵活。开发人员可以对构件的实现顺序进行优先级排序。
- 可增量开发，分期投资。

增量模型的缺点是要求待开发的软件系统可以被模块化。如果待开发的软件系统很难被模块化，那么将会给增量开发带来很多麻烦。

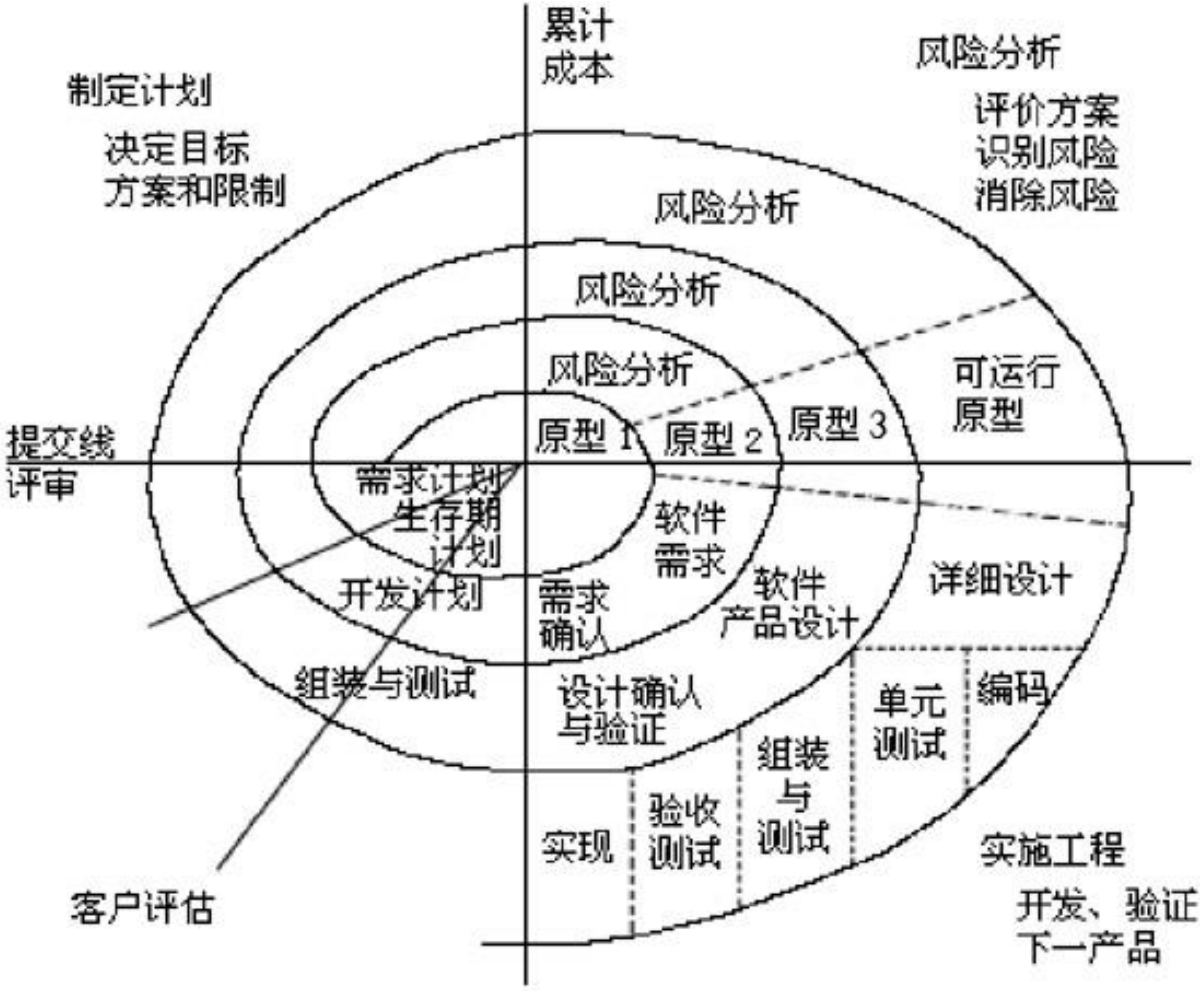
2.3 软件过程模型

2.3.4 螺旋模型

如果软件项目不仅存在需求不确定的风险，还存在其它人员、市场或技术等风险时，如何改进过程模型？

思考：快速原型模型很好地解决了需求不确定问题，要解决其它风险问题，能否在每次对原型进行构造或修改前，加入风险分析呢？

2.3. 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
可运行
原型
原型 1
原型 2
原型 3
提交线
评审
需求计划
生存期
计划
开发计划
软件
需求
需求
确认
软件
产品设计
详细设计
组装与测试
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

2.3 软件过程模型

2.3.4 螺旋模型

优点：

- 扩展了风险范围
- 风险驱动

缺点：

- 一个周期执行时间太长
- 要有方法和自动化工具支持，否则无法实施
- 开发人员必须能够胜任风险分析

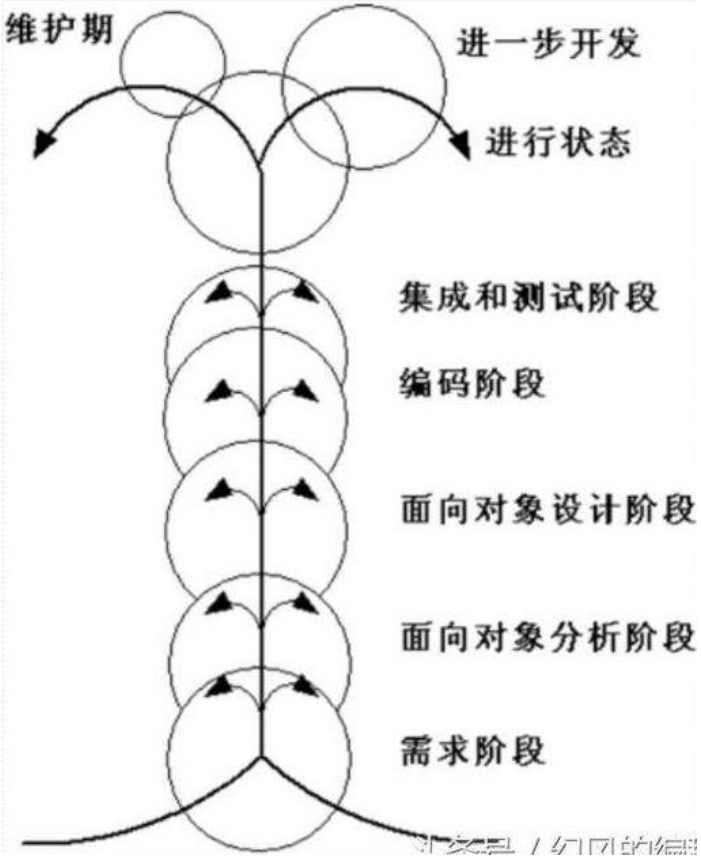
适用范围：

- 只适用于大型项目，如果项目的开发风险很大，可使用螺旋模型。

2.3 软件过程模型

2.3.5 喷泉模型

喷泉模型（Fountain Model）是由B. H. Sollers和J. M. Edwards于1990年提出的一种新的开发模型。



主要思想：

“喷泉”这个词体现了面向对象软件开发过程迭代和无缝的特性。为避免使用喷泉模型开发软件时开发过程过分无序，应该把一个线性过程作为总目标。

2.3 软件过程模型

2.3.5 喷泉模型

优点：

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，可能随时加入各种信息、需求与资料使得审核的难度加大。

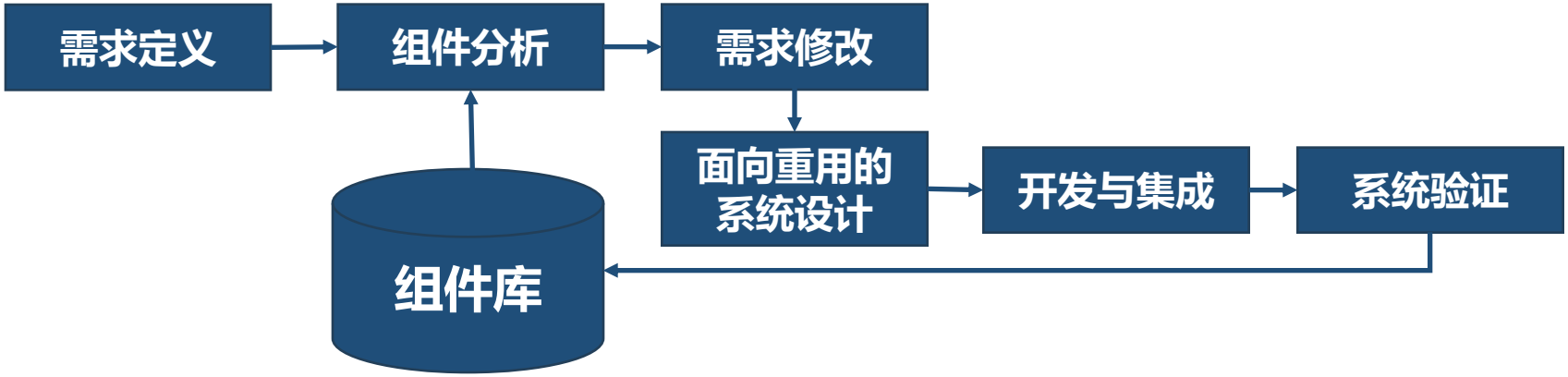
适用范围：

适应于面向对象的软件开发过程。

2.3 软件过程模型

2.3.6 基于组件的开发模型

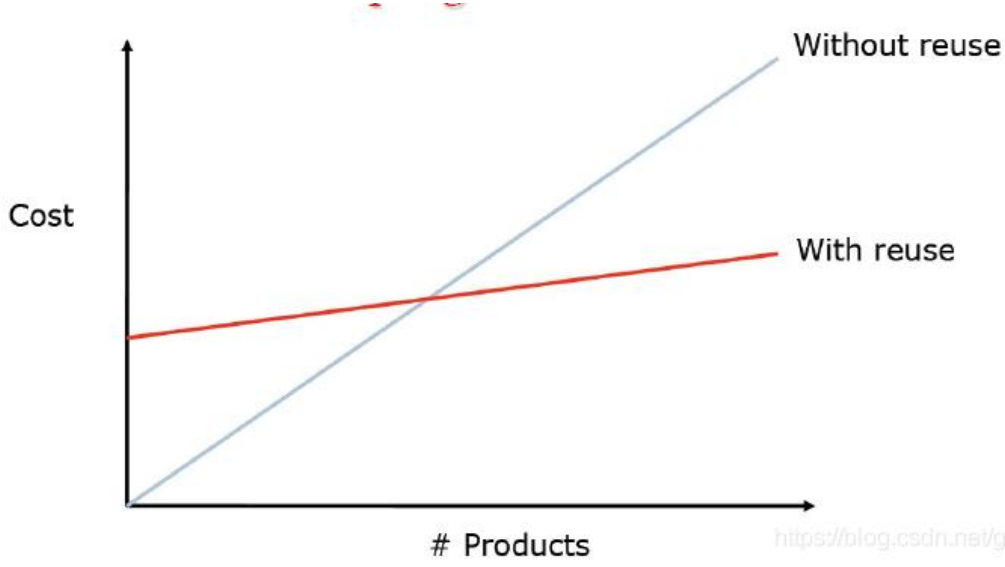
基于组件的开发模型使用现有的组件以及系统框架进行产品开发。在确定需求之后，开发人员开始从现有的组件库中筛选合适的组件，并对组件功能进行分析。在对组件分析之后，开发人员可能适当修改需求来适应现有组件，也可能修改组件或寻找新的组件。组件筛选完成之后，开发人员需要根据需求设计或使用现有的成熟开发框架复用这些组件，一些无法利用现有组件的地方，则需要进行单独的开发，新开发的组件在经历时间考验之后也会加入到组件库中。最后将所有组件集成在一起，进行系统测试。



2.3 软件过程模型

2.3.6 基于组件的开发模型

基于组件的开发模型充分的体现了软件复用的思想，降低了开发成本和风险，并加快了产品开发。



2.3 软件过程模型

2.3.7 Rational统一过程

前面的软件过程模型都有自己的局限性，能否总结软件工程最佳实践，建立一个完善的过程模型？

基于这种思想，Rational公司三位杰出的软件大师Grady Booch，Ivar Jacobson，James Rumbaugh总结软件工程的最佳实践于1996年提出了Rational统一过程——RUP（Rational Unified Process）。

2.3 软件过程模型

2.3.7 Rational统一过程

1.Rational Unified Process是什么？

* 是一种软件工程过程，它提供了如何在开发组织中严格分配任务和职责的方法。
* 是一个过程产品。
* 有自己的过程框架。
* 捕获了现代软件开发中的最佳实践。

2.3 软件过程模型

2.3.7 Rational统一过程

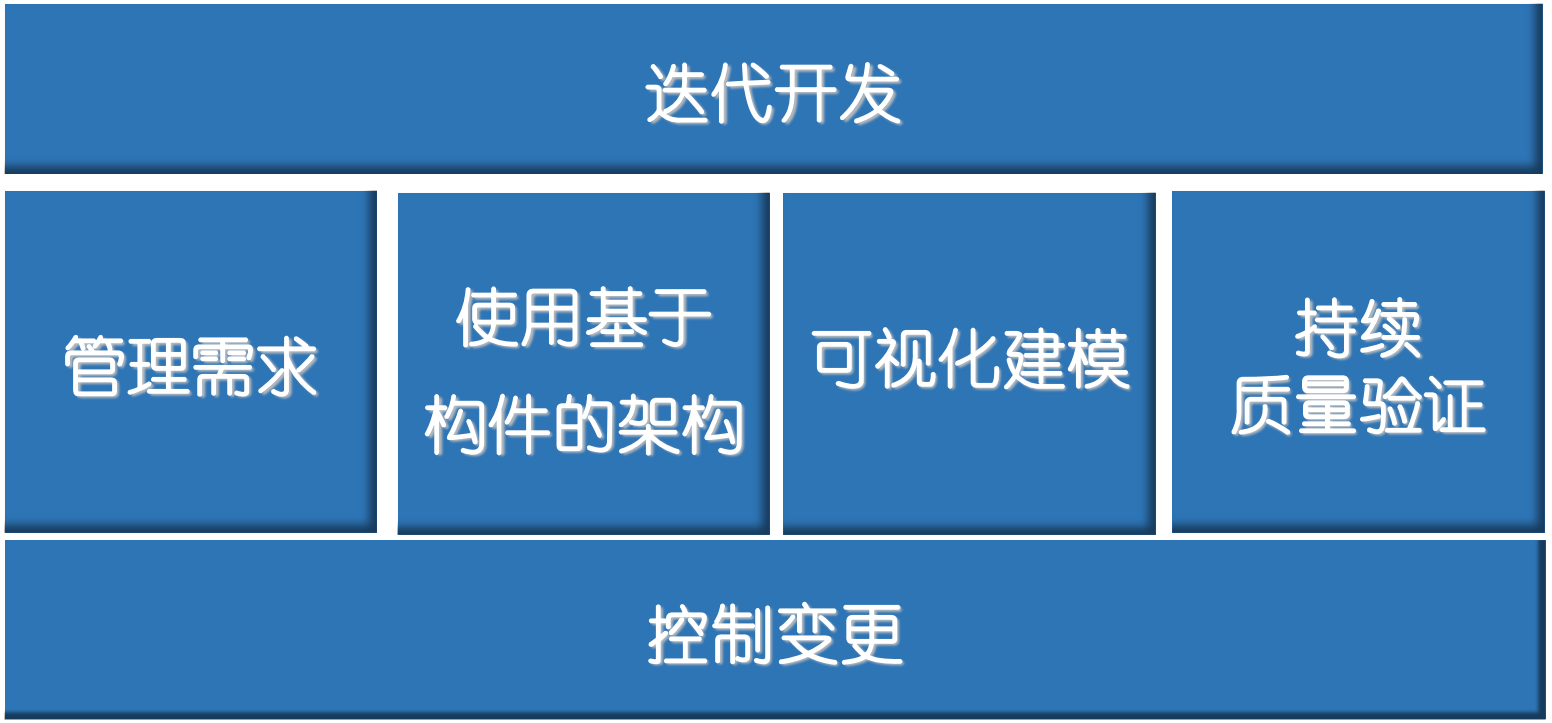
2.Rational Unified Process三大特点：

- 用例驱动
- 以架构为中心
- 迭代和增量开发

2.3 软件过程模型

2.3.7 Rational统一过程

3.RUP捕获的6项最佳商业实践



被证明是解决软件开发过程中根本问题的方法。

2.3 软件过程模型

2.3.7 Rational统一过程

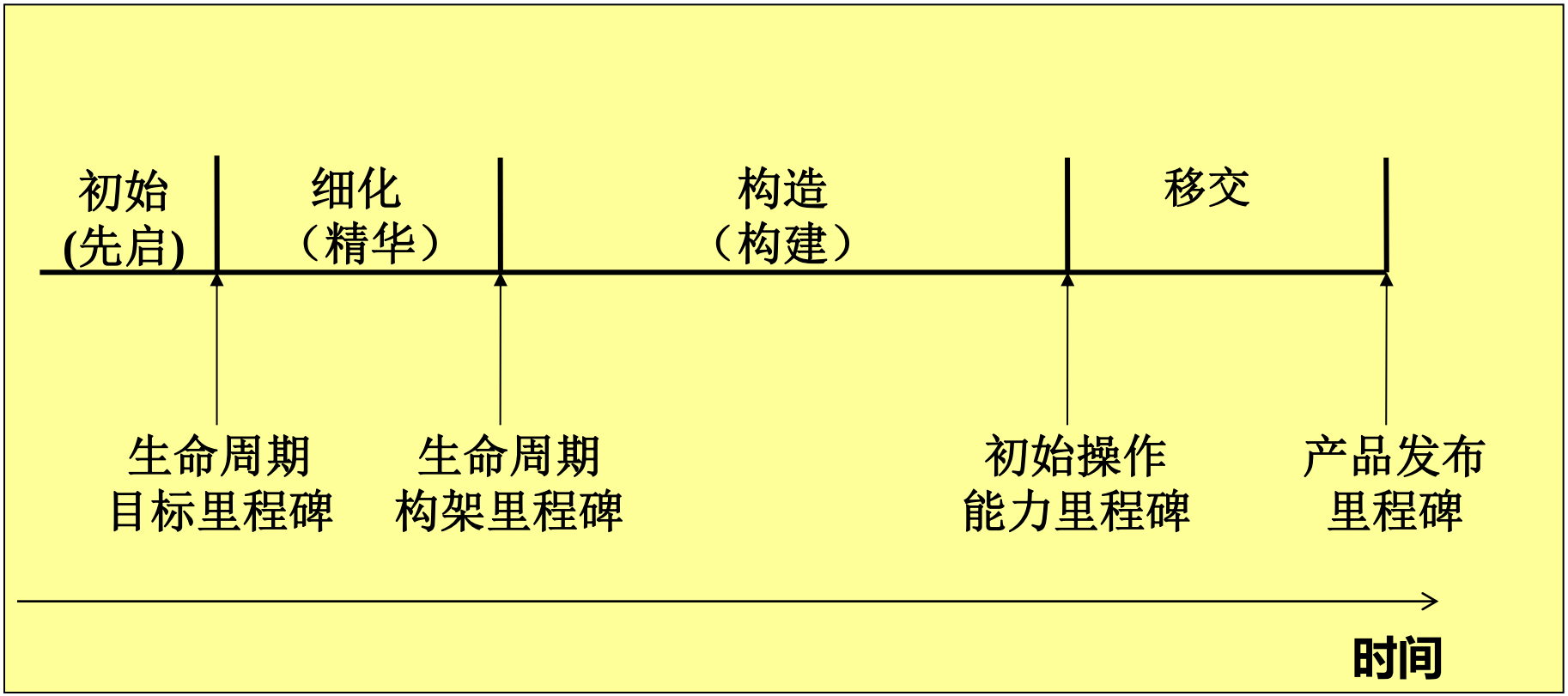
4.RUP的九大工作流

* 需求工作流
* 分析设计工作流
* 实现工作流
* 测试工作流
* 部署工作流
* 配置与变更管理工作流
* 项目管理工作流
* 环境工作流

2.3 软件过程模型

2.3.7 Rational统一过程

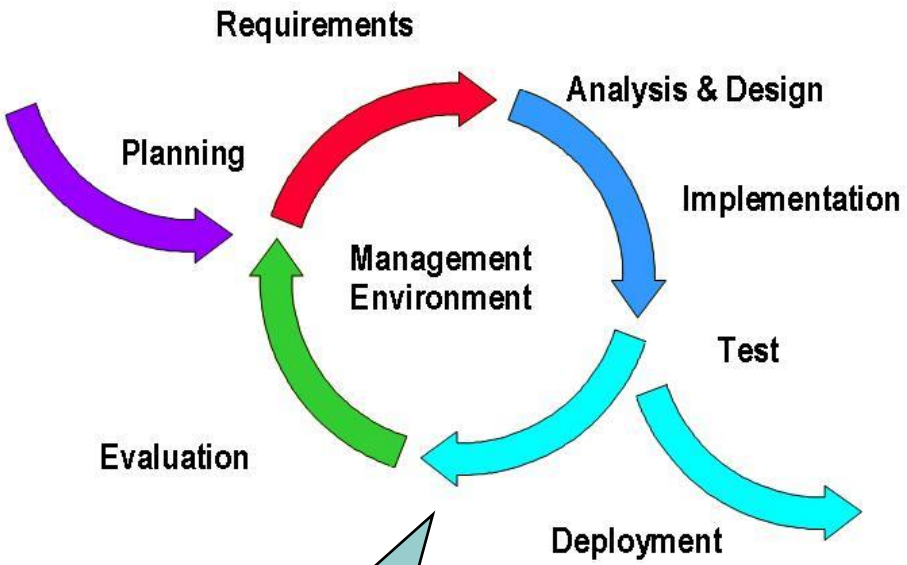
5.四个工作阶段



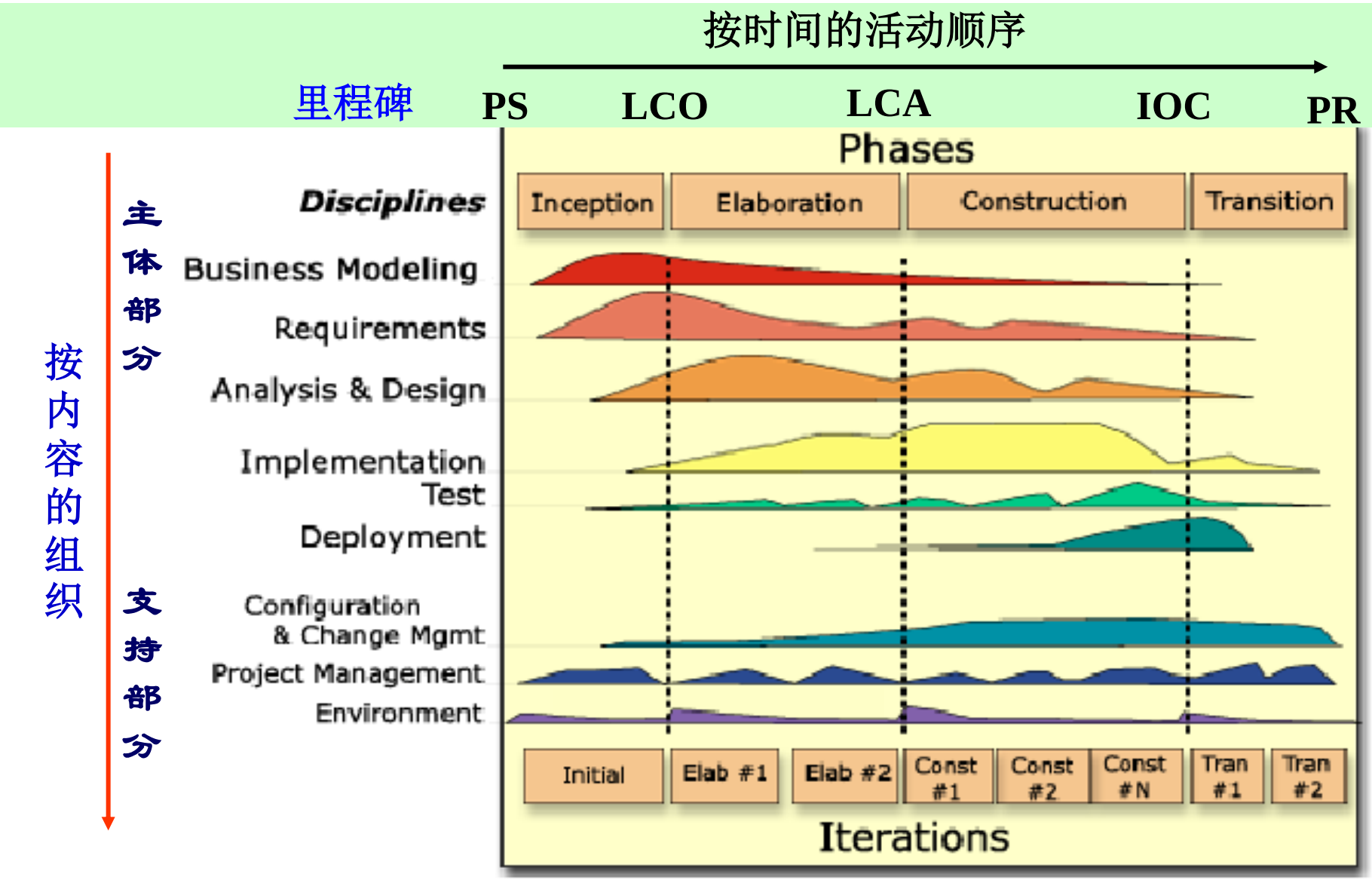
2.3 软件过程模型

2.3.7 Rational统一过程

6.RUP迭代式开发



Each iteration results in an executable release.


按时间的活动顺序
里程碑
PS
LCO
LCA
IOC
PR
按内容的组织
主体部分
支持部分
Phases
Disciplines
Inception
Elaboration
Construction
Transition
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment
Configuration & Change Mgmt
Project Management
Environment
Initial
Elab #1
Elab #2
Const #1
Const #2
Const #N
Tran #1
Tran #2
Iterations

2.3 软件过程模型

2.3.7 Rational统一过程

主要困难：

- 多层次持续的规划与评估
- 判断构架中关键风险的经验
- 高效率的验证和评价手段
- 多工种之间的频繁沟通
- 多版本工作产品的管理

基础保障：

- 核心人员必要的管理与技术经验
- 自动化的验证和评价工具
- 团队成员之间有(高)效的沟通工具
- 软件配置与变更管理工具

2.3 软件过程模型

2.3.8 几种模型之间的关系

1．瀑布模型与RUP模型之间的关系

在宏观上，瀑布模型是静态模型，RUP模型是动态模型。RUP模型的每一次迭代，实际上都需要执行一次瀑布模型，都要经历先启、细化、构建、产品化这4个阶段，完成瀑布模型的整个过程。

在微观上，瀑布模型与RUP模型都是动态模型。瀑布模型与RUP模型在每一个开发阶段（先启、细化、构建、产品化）的内部，都需要有一个小小的迭代过程，只有进行这样的迭代，开发阶段才能做得更好。

2.3 软件过程模型

2.3.8 几种模型之间的关系

2．瀑布模型与增量模型之间的关系

增量模型是把待开发的软件系统模块化，将每个模块作为一个增量组件，一个模块接着一个模块地进行开发，直到开发完所有的模块。

在开发每个模块时，通常都是采用瀑布模型，从分析、设计、编码和测试这几个阶段进行开发。所以，增量模型中有瀑布模型，即宏观上是增量模型，微观上是瀑布模型。

增量模型也体现了迭代思想，每增加一个模块，就进行一次迭代，执行一次瀑布模型，所以，增量模型本质上是迭代的。

2.3 软件过程模型

2.3.8 几种模型之间的关系

3．瀑布模型与快速原型模型之间的关系

快速原型的基本思想是快速建立一个能反映用户主要需求的原型系统，在此基础上之后的每一次迭代，都可能会用到瀑布模型。

快速原型模型中不但包含了迭代模型的思想，而且包含了瀑布模型的思想。

2.3 软件过程模型

2.3.8 几种模型之间的关系

4．瀑布模型与螺旋模型之间的关系

螺旋模型是瀑布模型和快速原型模型的结合，快速原型模型是原型模型的简化，原型模型又是迭代模型和瀑布模型的组合，这些模型之间是相互依存的、彼此有关的。

螺旋模型每一次顺时针方向旋转，相当于顺时针方向迭代一次，都是走完一次瀑布模型，这就是瀑布模型与螺旋模型之间的关系。实际上，瀑布模型与喷泉模型也有关系。

2.3 软件过程模型

2.3.9 选择软件过程模型

在具体的软件开发过程中，可以选择某种软件过程模型，按照某种开发方法，使用相应的工具进行软件开发。

在选择软件过程模型时需要考虑以下几点：

- 符合软件自身的特性，如规模、成本和复杂性等
- 满足软件开发进度的要求
- 对软件开发的风险进行预防和控制
- 具有计算机辅助工具的支持
- 与用户和软件开发人员的知识和技能相匹配
- 有利于软件开发的管理和控制

2.4 软件过程实例

例2-1

假设要为一家生产和销售长筒靴的公司开发一个软件，使用此软件来监控该公司的存货，并跟踪从购买橡胶开始到到长筒靴、发货给各个连锁店，直至卖给顾客的全过程。以保证生产、销售过程的各个环节供需平衡，既不会出现停工待料的现象，也不会出现供不应求的现象。可以为这个项目选择什么软件过程模型?

【解析】采用瀑布模型或增量面模型。

原因：项目需求明确，技术方案比较成熟，可采用瀑布模型。亦可以明确划分模块/组件，选择增量模型。

2.4 软件过程实例

例2-2

假设计划研发一种产品，技术含量很高，与客户有关的风险也很多，你会采用哪种软件过程模型？请说明理由。

【解析】采用螺旋模型或RUP模型。

原因：存在除需求不确定之外的其它风险，技术含量高，至少要用螺旋模型，如果规模大，需要RUP模型。

小结

本章首先介绍了软件工程过程的基本概念；然后讲述了软件生命周期各阶段的主要任务和活动。软件生命周期各阶段活动的不同组织方式形成了不同的软件过程模型，本章重点讲解了瀑布模型、快速原型模型、增量模型、螺旋模型、喷泉模型、基于组件的开发模型，以及统一软件开发过程模型。最后给出了选择软件过程模型的几点原则。**重点培养学生根据软件项目特点合理选择软件过程模型的能力**。