1.6 Software life cycle

软件产品或软件系统从设计、开发、投入使用到被淘汰的全过程。

Phases of software life cycle

(1) problem def.

(2) feasibility study

(3) requirement analysis

(4) architecture design

(5) detailed design

(6) coding and implementation

(7) testing

(8) running and maintenance

(1) 问题定义

(2) 可行性分析

(3) 需求分析

(4) 总体设计

(5) 详细设计

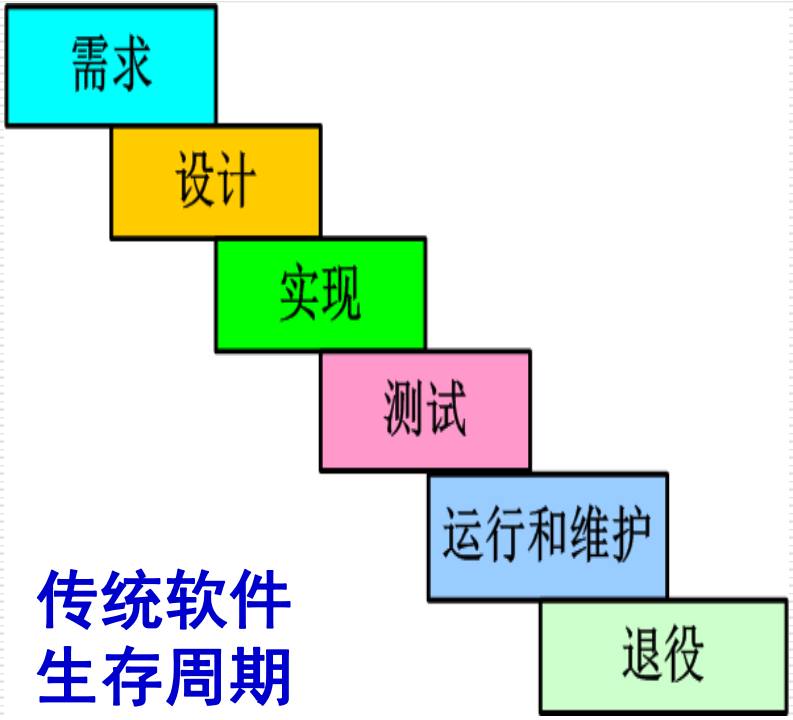
(6) 编码实现

(7) 测试

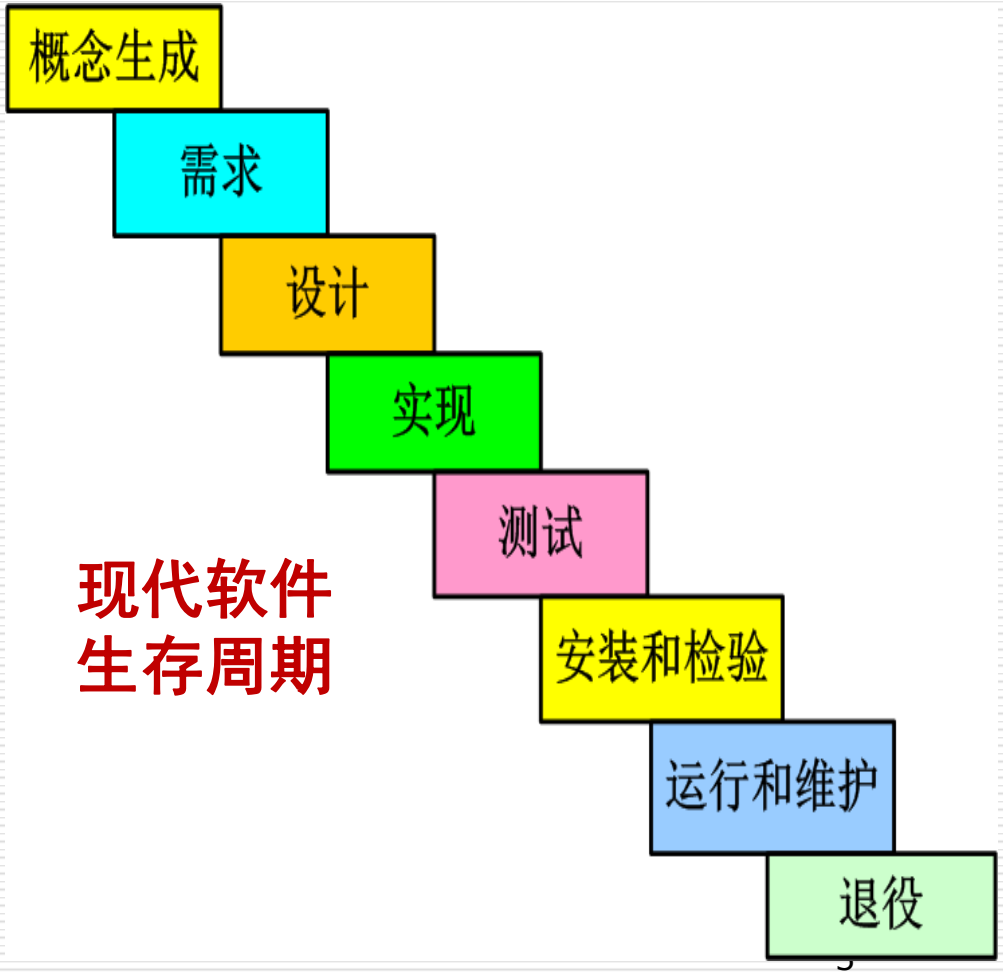
(8) 运行和维护

软件工程的 8个阶段，8个过程

软件生存周期



传统软件
生存周期



现代软件
生存周期

软件过程：
加工，动作

生命周期
产品，状态

1.7 Software Process Model

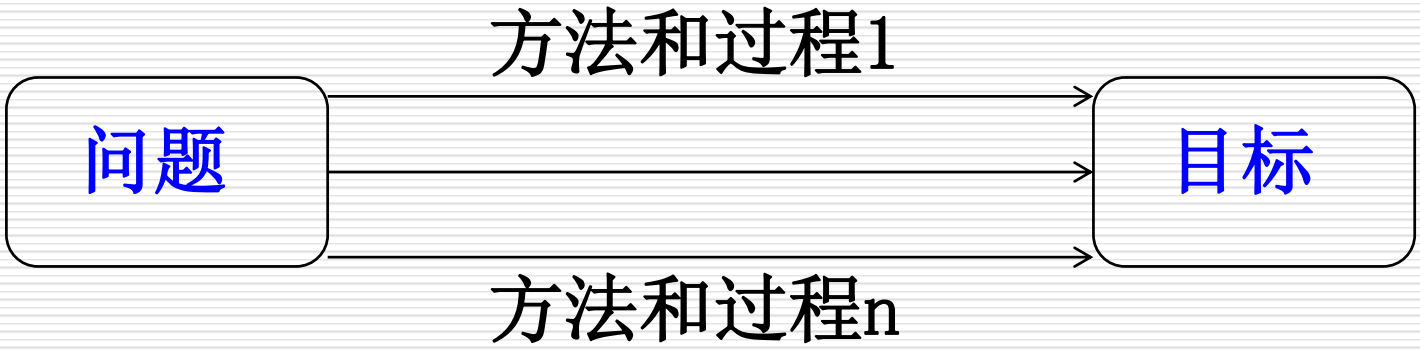
Software process is a set of activities and associated results which produce a software product.

软件过程是为了开发和生产软件所需完成的一系列任务的框架，它规定了完成各个任务的步骤。

Software process model is a simplified description of a software process which is presented from a particular perspective.

Model: Abstraction, framework,

软件过程



- 麦当劳快餐质量保证，每个店的味道一样

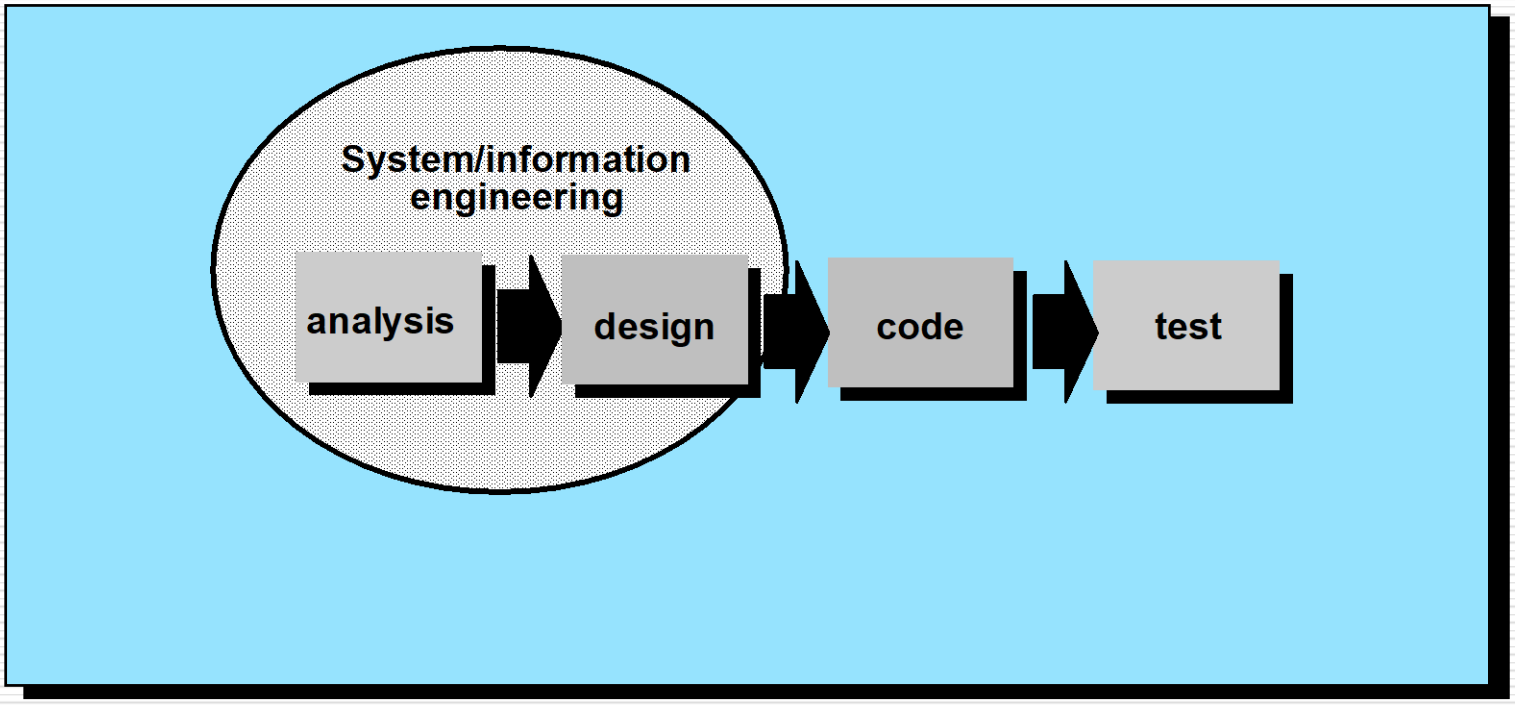
- 学生择校读书过程，不输在起步线上

各种软件过程理论模型

- linear model
- iterative model
- incremental model
- automatic transformation model
- ……

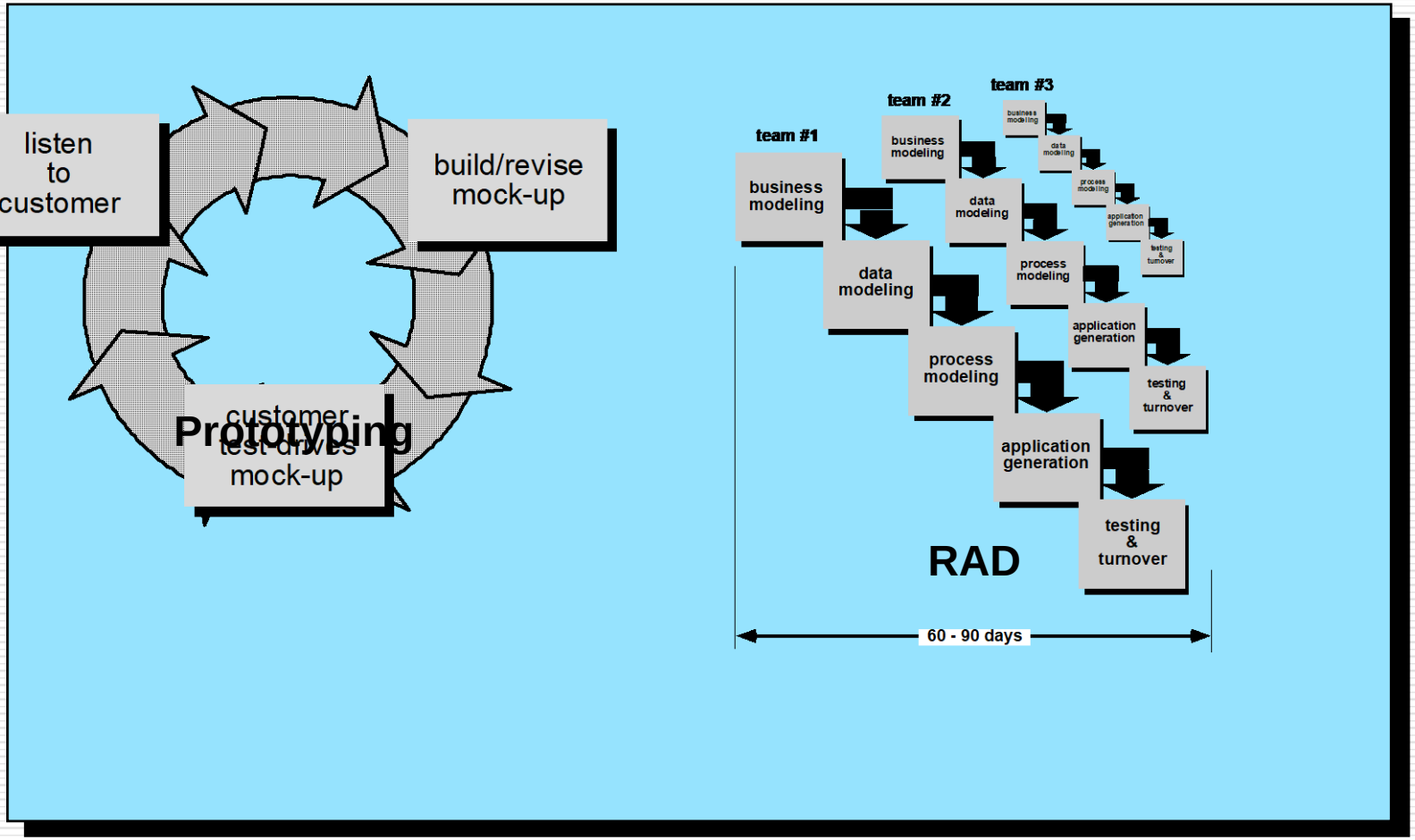
（1）The Linear Model

线性顺序过程模型



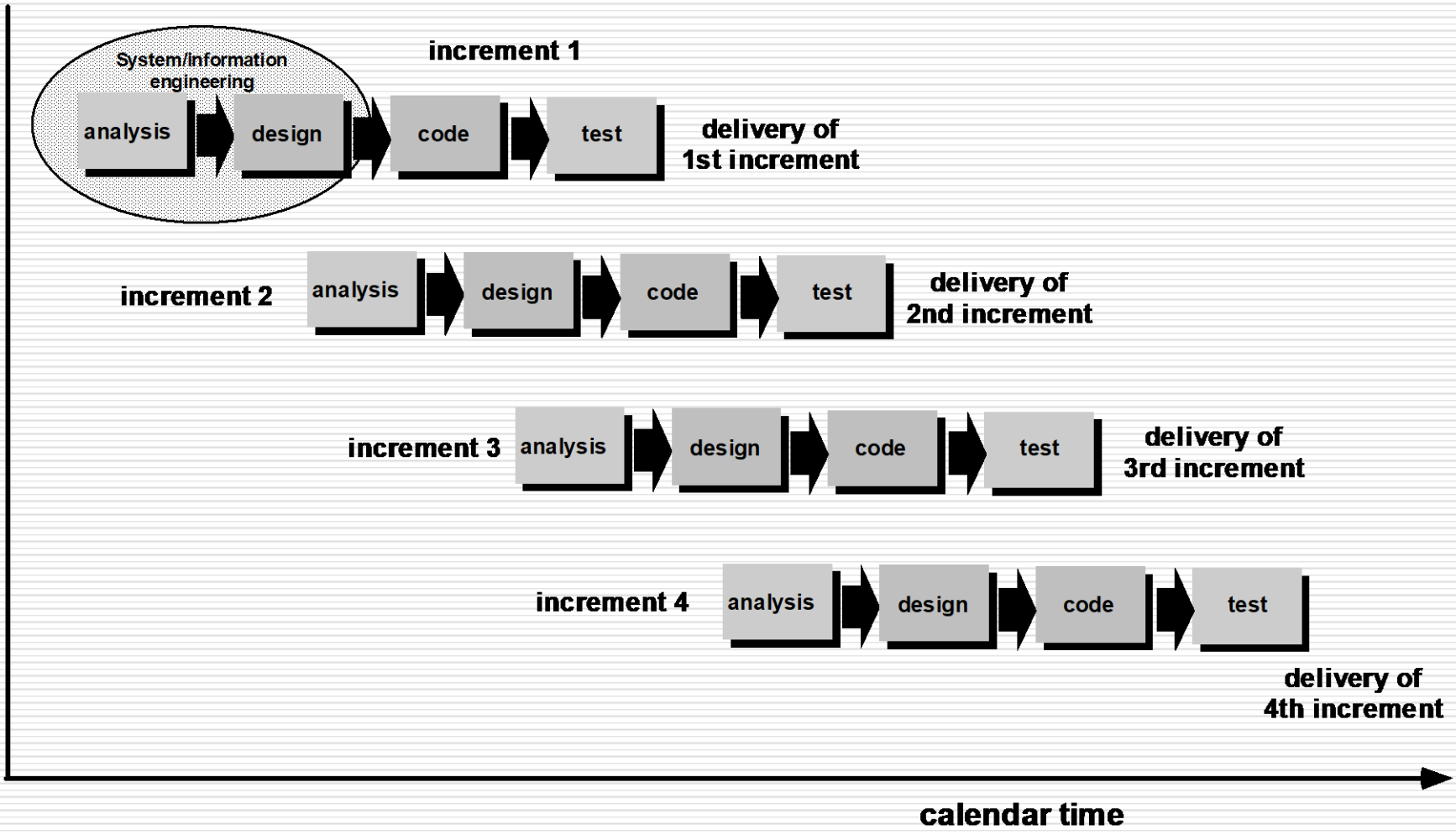
（2）The Iterative Models

循环过程模型



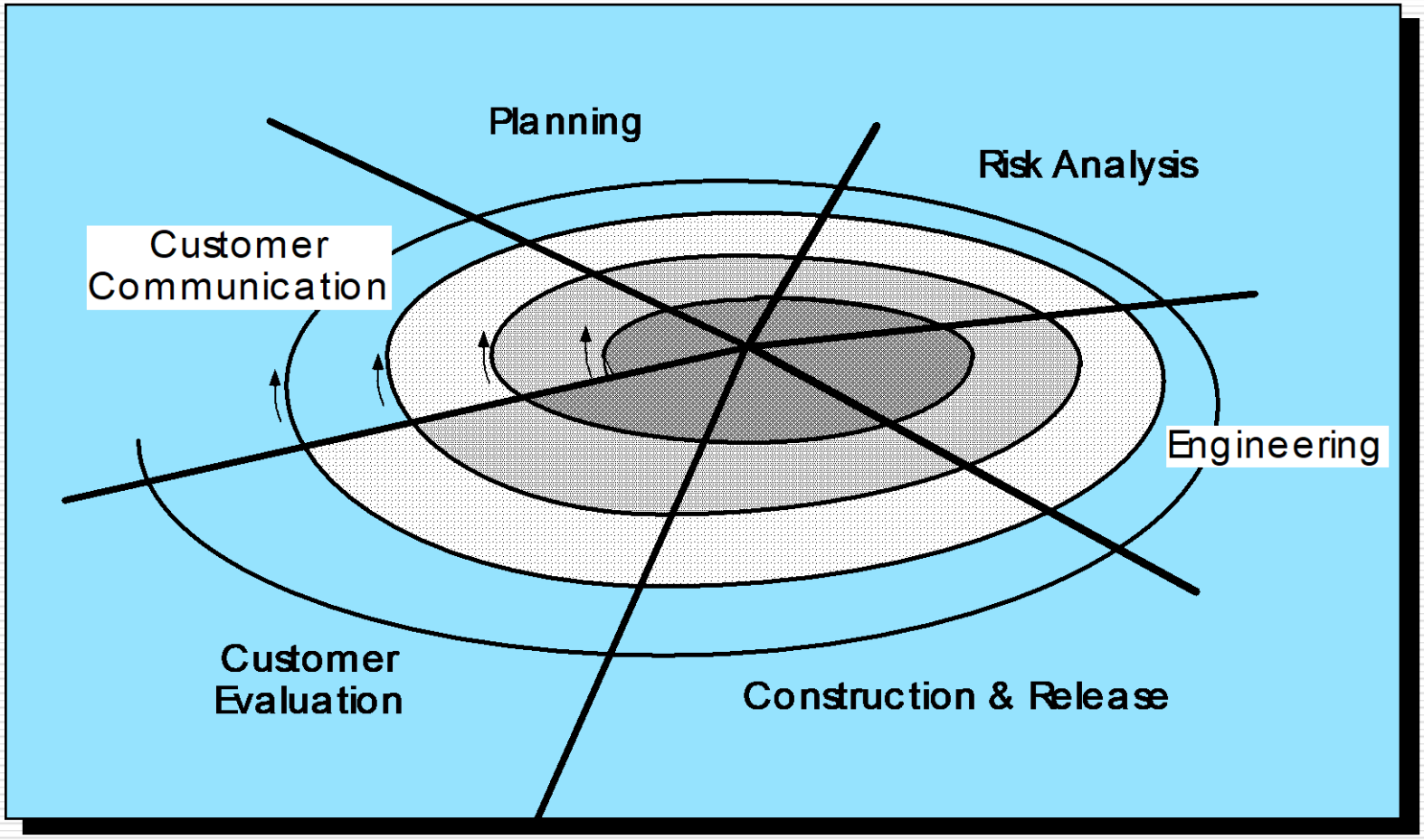
（3）The Incremental Model

增量过程模型



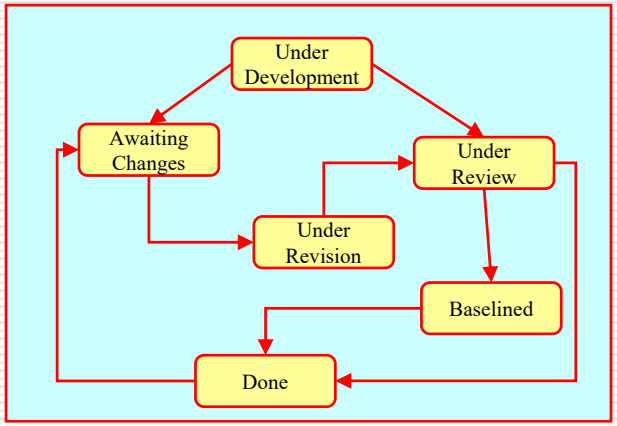
（4）The Spiral Model

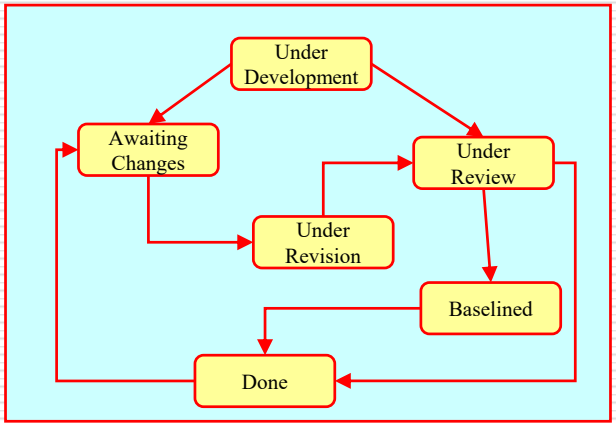
螺旋过程模型

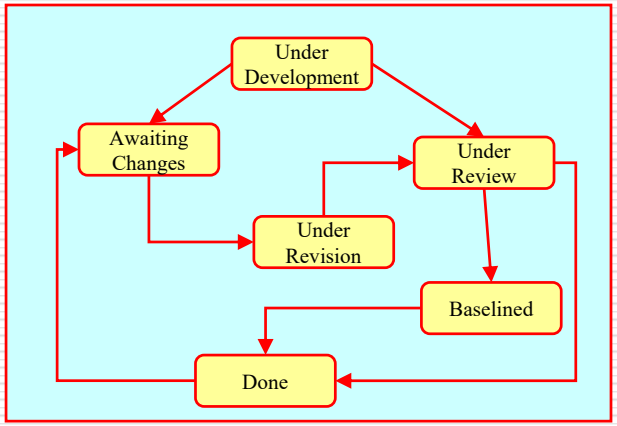


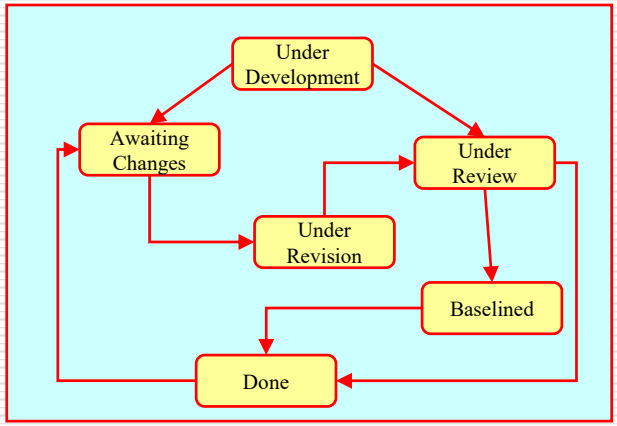
(5) Parallel Model

并行过程模型

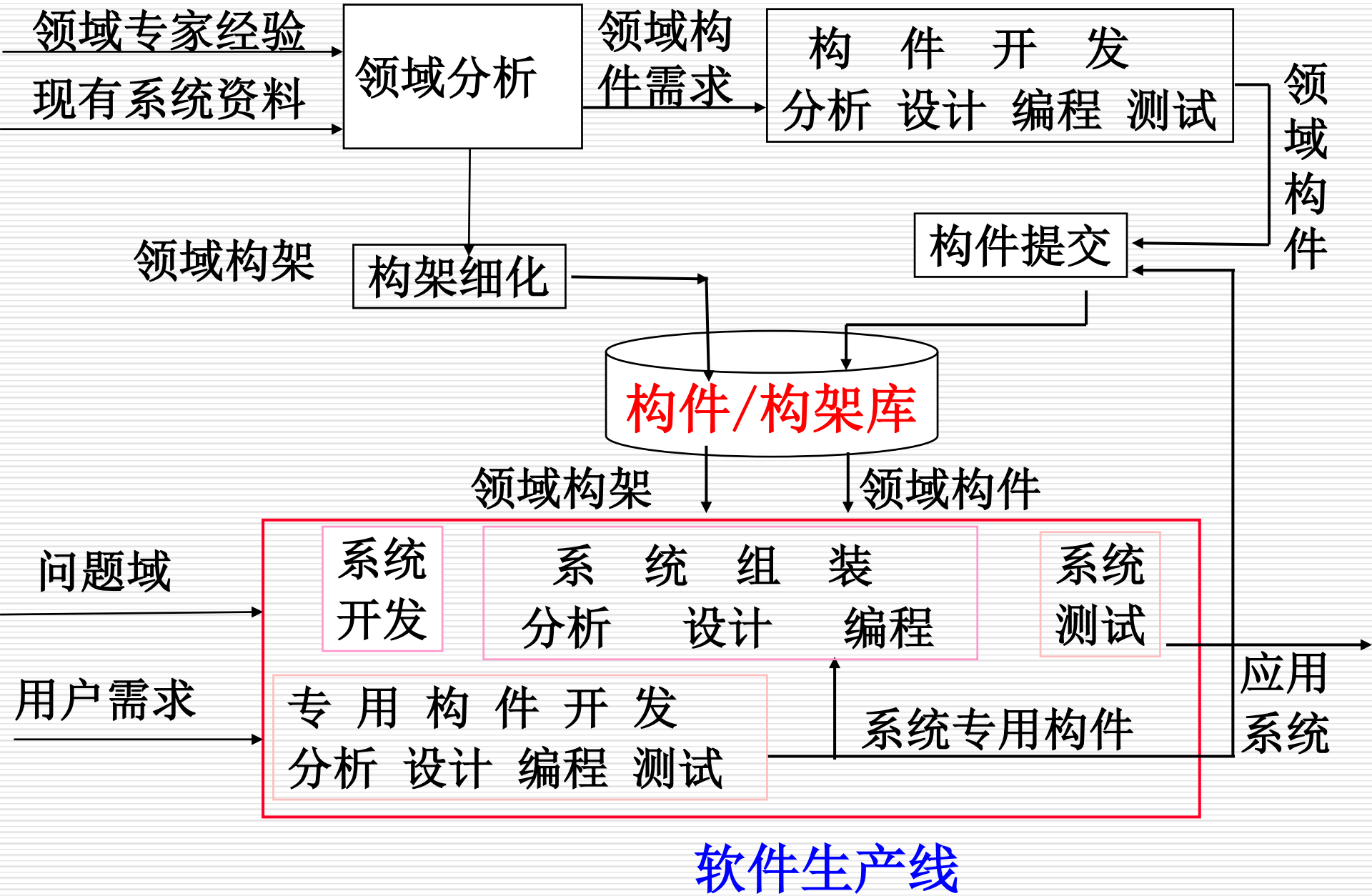






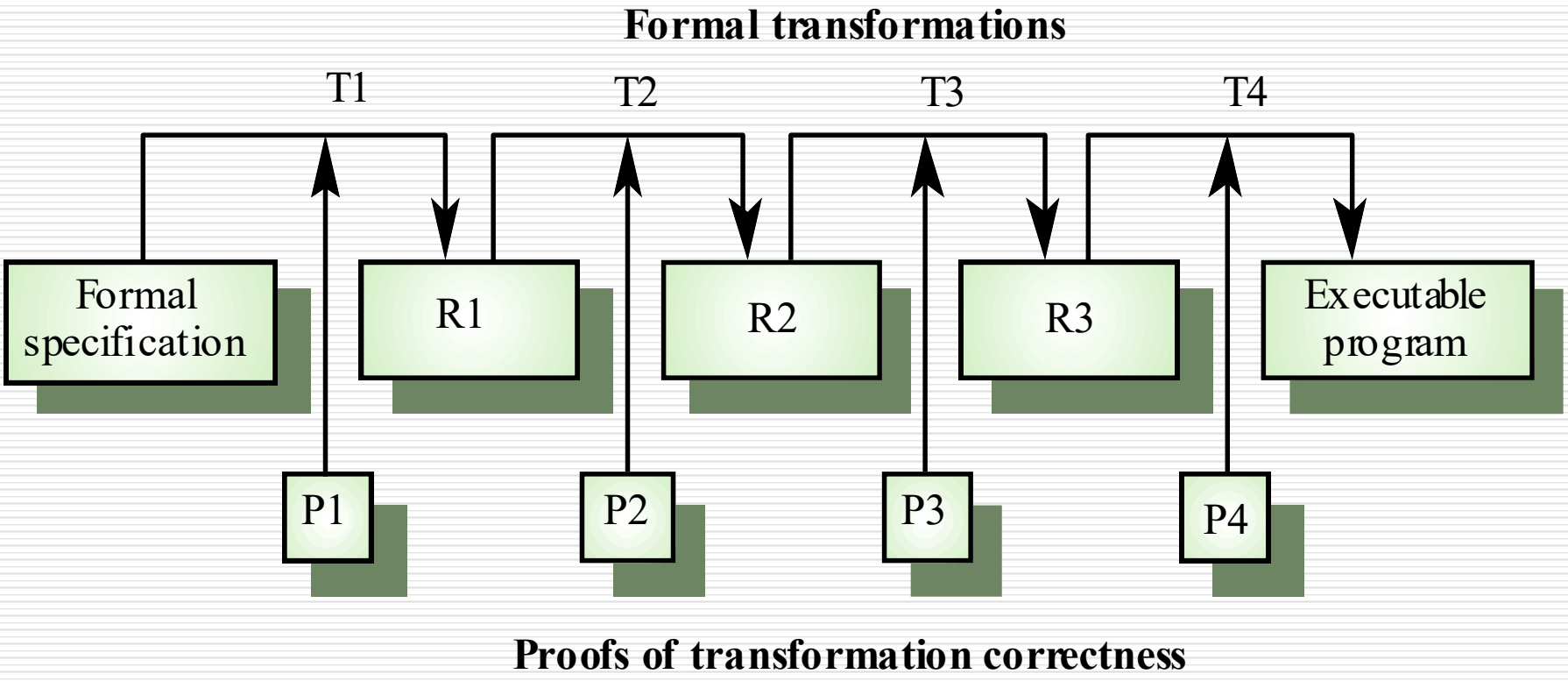


（6）Component Based Model



（7）Formal Transformations Model

形式化变换模型



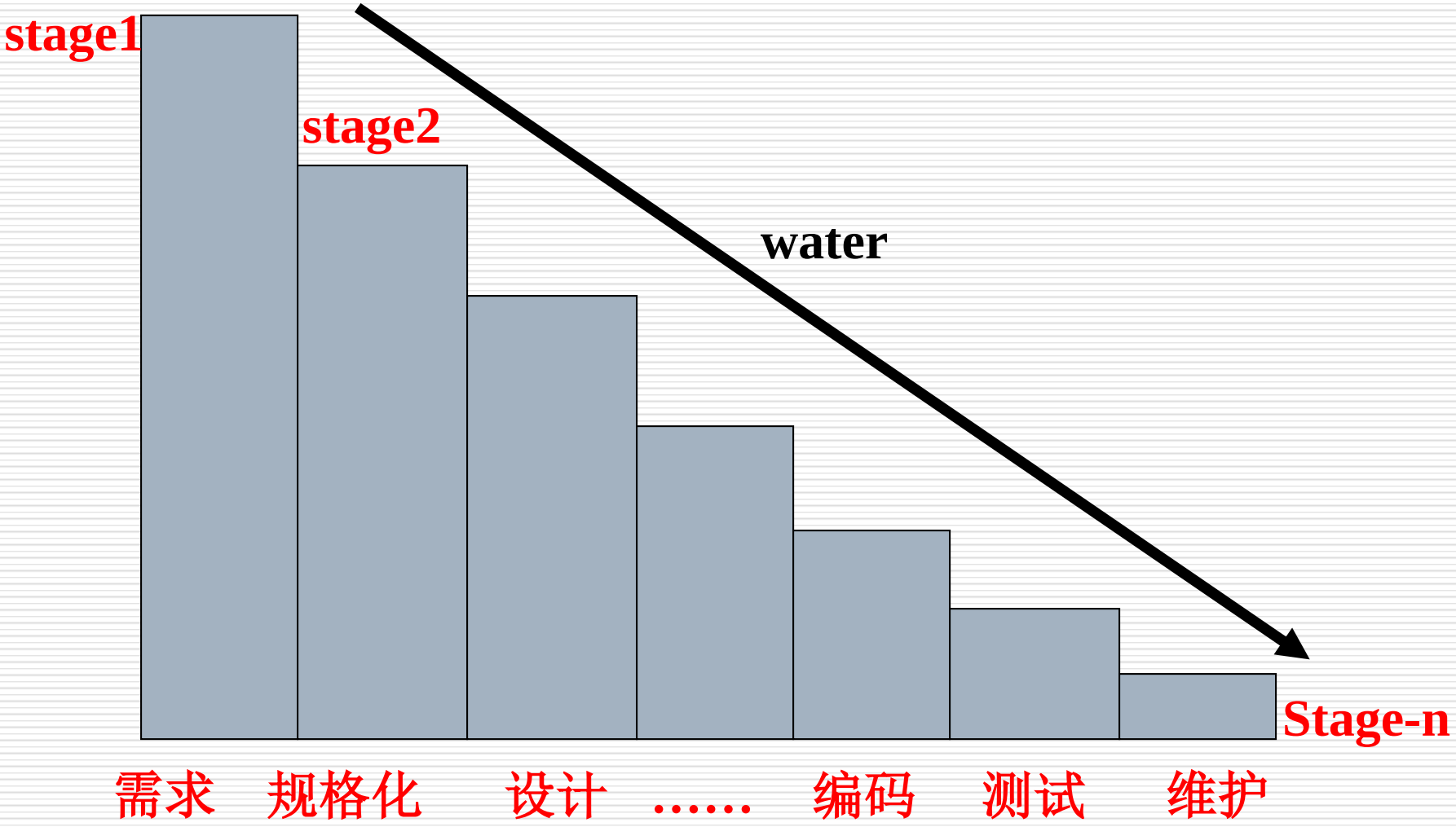
Automatic Transformations

实用的软件工程过程模型（教材）

- 瀑布模型
- 快速原型
- 增量模型
- 螺旋模型
- 喷泉模型 （自学）
- **Rational** 统一过程模型 （自学）
- 敏捷过程和极限编程模型（自学）
- 微软过程模型（自学）

- ✓ 掌握特点
- ✓ 学会选择
- ✓ 综合应用

(1) Waterfall model



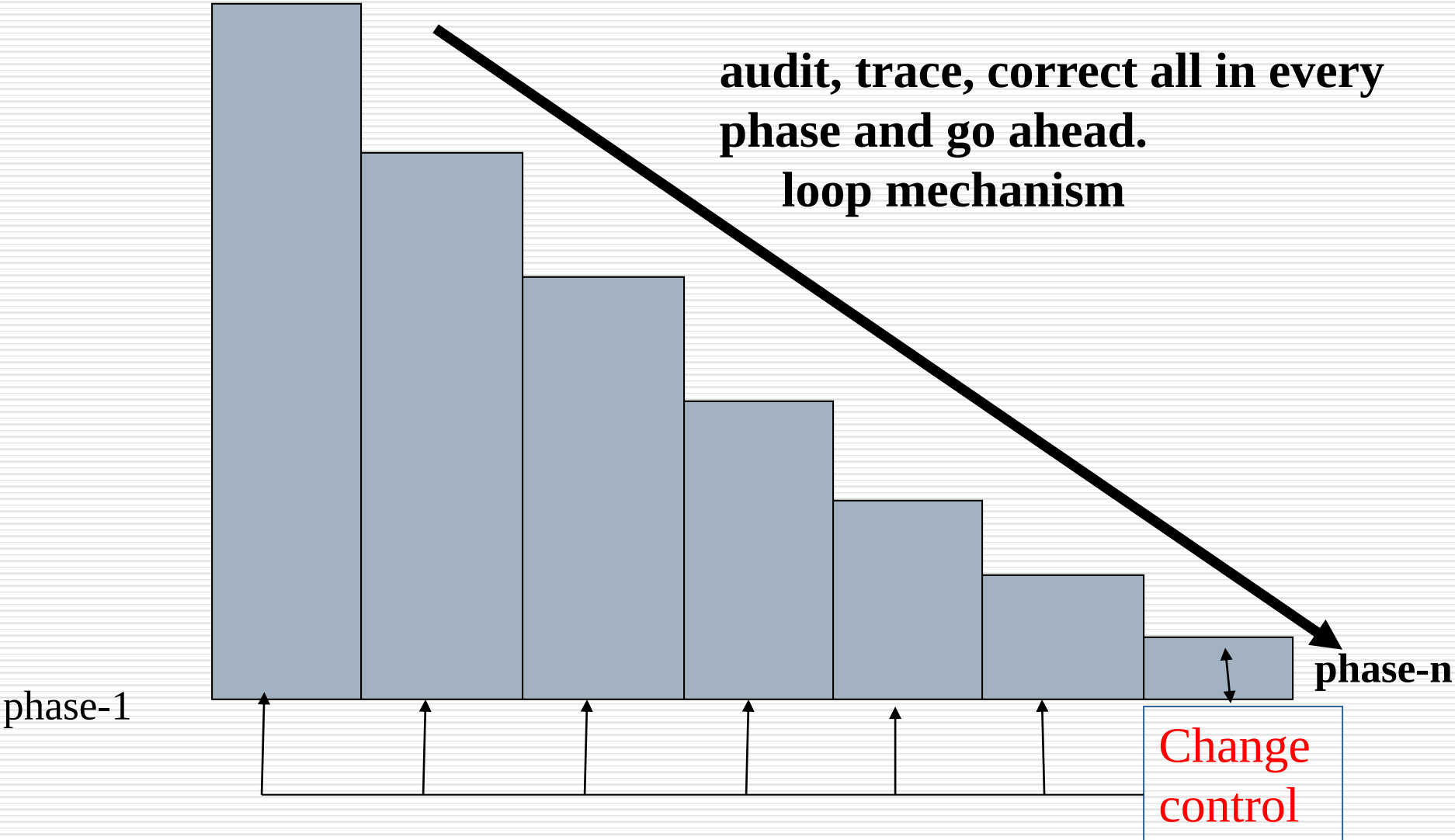
特点

- time sequence and dependence (顺序性和依赖性)
- delay to implementation (推迟实现)
- document driven (每个阶段必须完成规定的文档，文档驱动)
- serial, not in parallel （不能并行工作）
- process cannot work inversely （过程不能逆转）
- error amplifying （错误放大）
- software product in final phase （在最后阶段才能够出现）
- delivered product may not meet client's needs （不满足需求）
- classical model, to facilitate management （经典，容易管理）

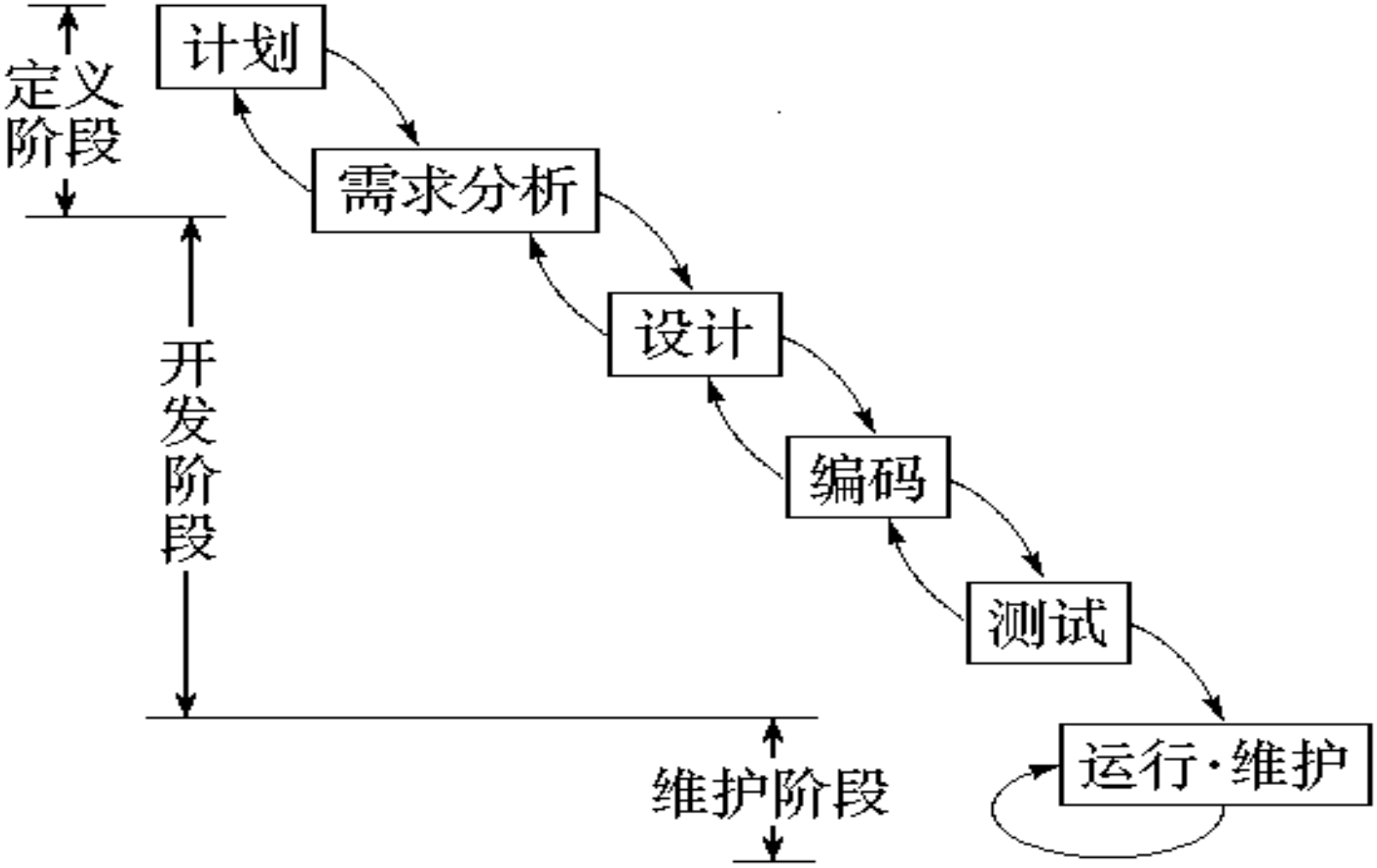
瀑布模型 — 问题

- 实际项目很少严格按照该模型给出的顺序进行
- 用户常常难以清楚地给出所有需求
- 用户必须有耐心
- 开发者常常被不必要地耽搁

Improving


audit, trace, correct all in every phase and go ahead.
loop mechanism
phase-n
phase-1
Change control

瀑布模型的改进



When, where to use waterfall model ?

- ✓Requirements are well known.
- ✓Requirements are unchanged during developing.
- ✓Structure (process) oriented method.

(2) Rapid Prototype Model

原型是什么？

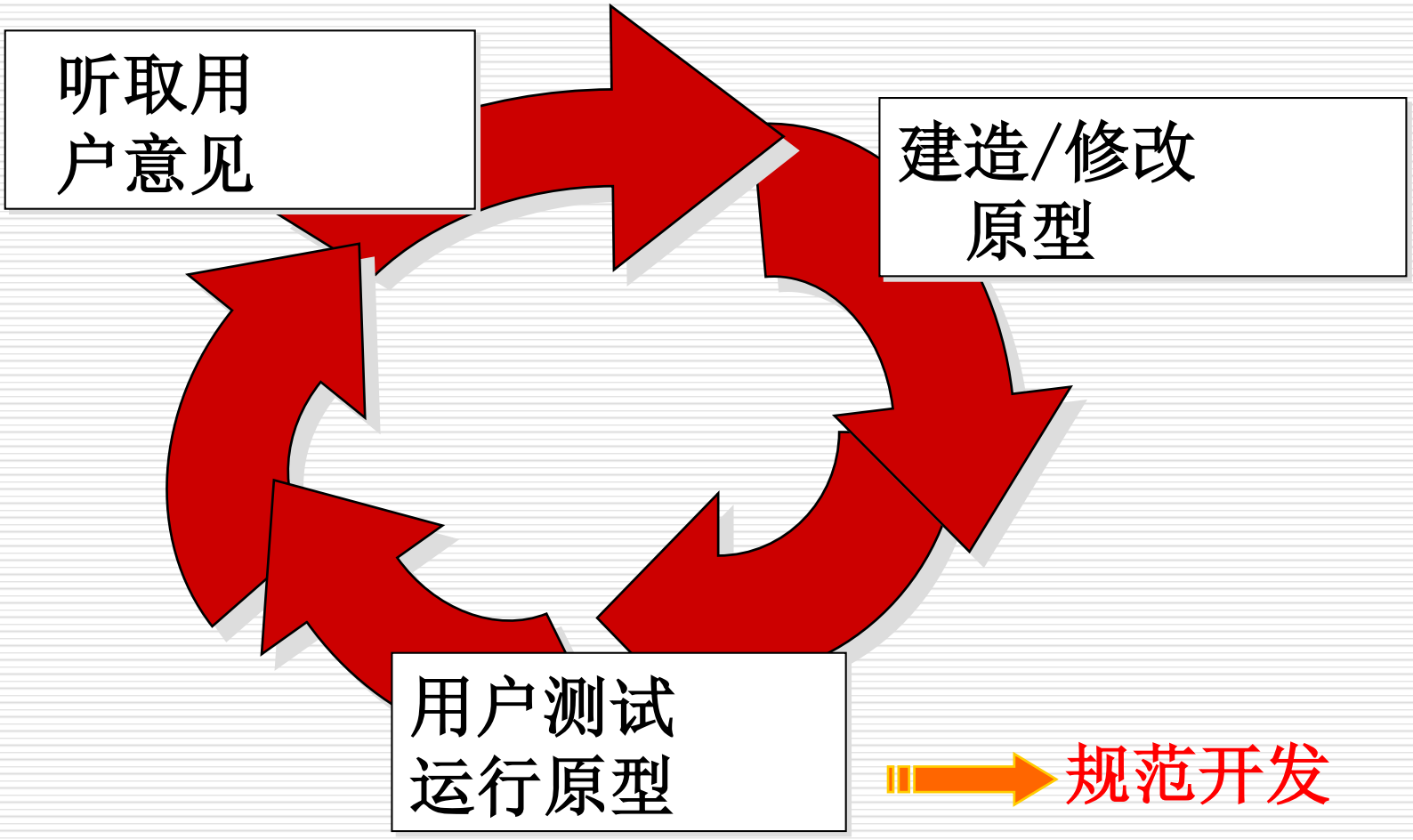
建筑工程原型是什么？

铸造工程原型是什么？

计算机领域原型是什么？

- 原型：不是真型，雏形，样品，样机，粗糙的模型，…　模拟，仿真
- 帮助早期描述、建立真实产品、对象、实体、系统等的一种手段

(2) Rapid Prototype Model



根据需求，快速建立可以在计算机上运行的程序

快速原型模型

- 原型模型从需求收集开始。开发者和用户在一起定义软件的总体目标，标识出已知的需求，并规划出进一步定义的区域。
- 然后是“快速设计”，快速设计集中于软件那些对用户可见部分的表示。“快速设计”导致原型的建造。
- 原型由用户评估，并进一步精化待开发软件的需求，逐步调整原型使其满足客户的要求。同时开发者对将要做的事情有更好的理解，这个过程是反复迭代的。

Characteristics

- ✓ requirement driving (需求驱动，客户开心)
- ✓ product, early realization
- ✓ no clear phases, difficult in management
- ✓ systems are often poorly structured
- ✓ special skills
- ✓ build-and-fix， again and again

When and where to use it?

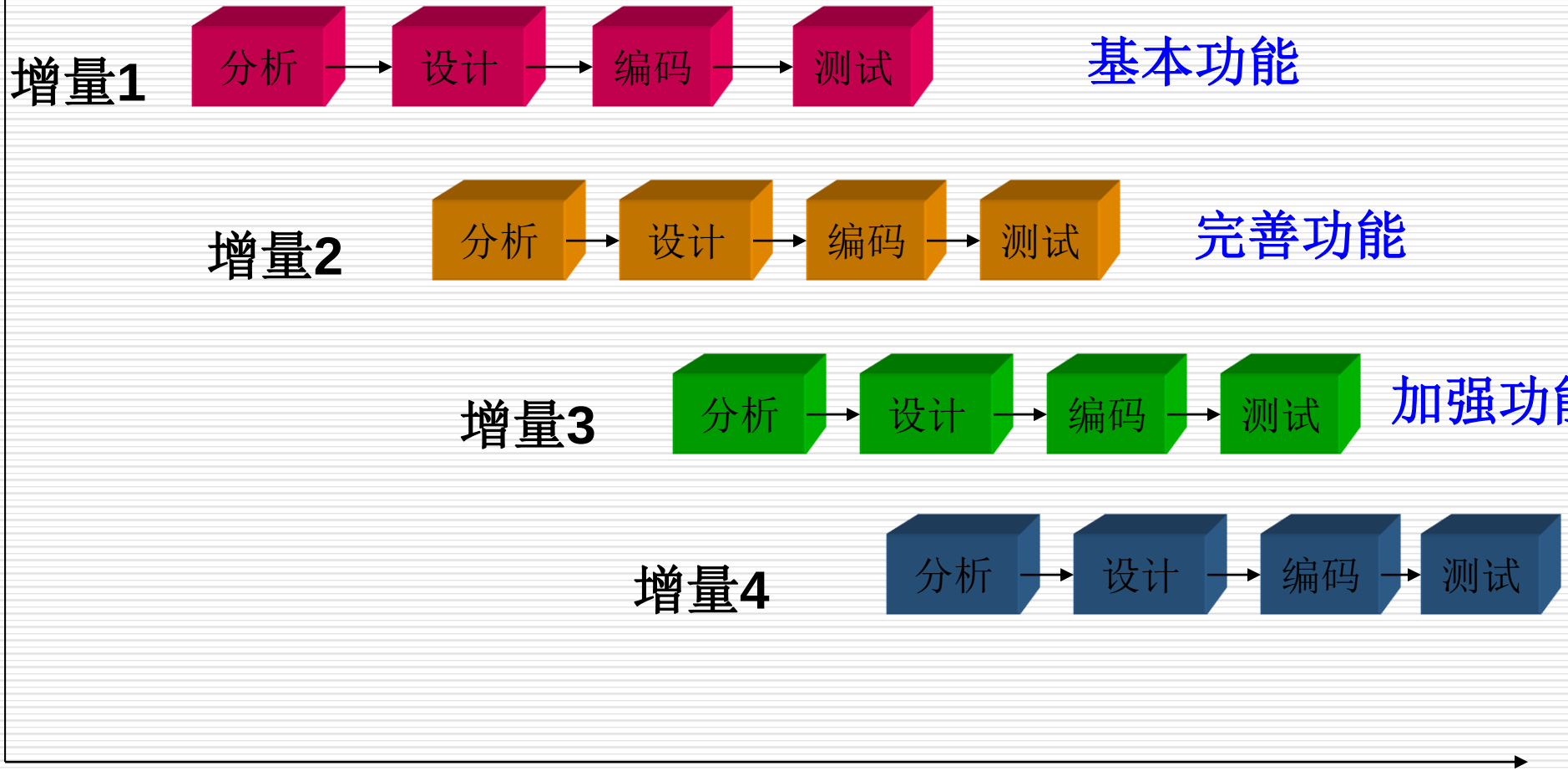
- ✓ requirement is not too clear.
- ✓ client is will to take part in the project
- ✓ there is a tool to develop such prototype.

- 用户定义了一组一般性目标，但不能标识出详细的输入、处理及输出需求；
- 开发者可能不能确定算法的有效性、操作系统的适应性或人机交互的形式；
- ……

(3) The Incremental Model

- 为了商业上抢占市场，尽早推出软件产品
- 先对软件体系结构进行设计
- 将需求划分为一系列增量，排序，急需的增量排在前面，不急需的放在后面。
- 每个增量都历经需求、设计、编码、测试、移交几个阶段。
- 根据增量依赖关系、项目实际，采用串行或并行
- 不断开发、集成、交付，及时反馈，得到最终软件制品。

(3) The Incremental Model



增量模型的特点

- 融合了瀑布模型的基本成分和原型的迭代特征，采用随着日程时间的进展，交错线性活动序列。
- 每次开发过程量力而行，循序渐进
- 软件产品功能不断增强
- 软件产品需要长远精心规划，预留接口
- 前后软件产品要求兼容

An example

例如，使用增量模型开发字处理软件: **MS-word**

1. 基本的文件管理、编辑和文档生成功能。
2. 更完善的编辑和文档生成能力。
3. 实现拼写和文法检查功能。
4. 完成高级的页面布局功能。

增量模型（Cont.）

- 第一个增量往往是核心产品
- 每一个增量均发布一个可操作产品
- 早期的增量是最终产品的“可拆卸”版本
- **Version**

- **EDLINE，EDIT, PEFECT, WORD 93，WORD 97, WORD 2010，WORD 2013, …**

Spiral Model 螺旋模型

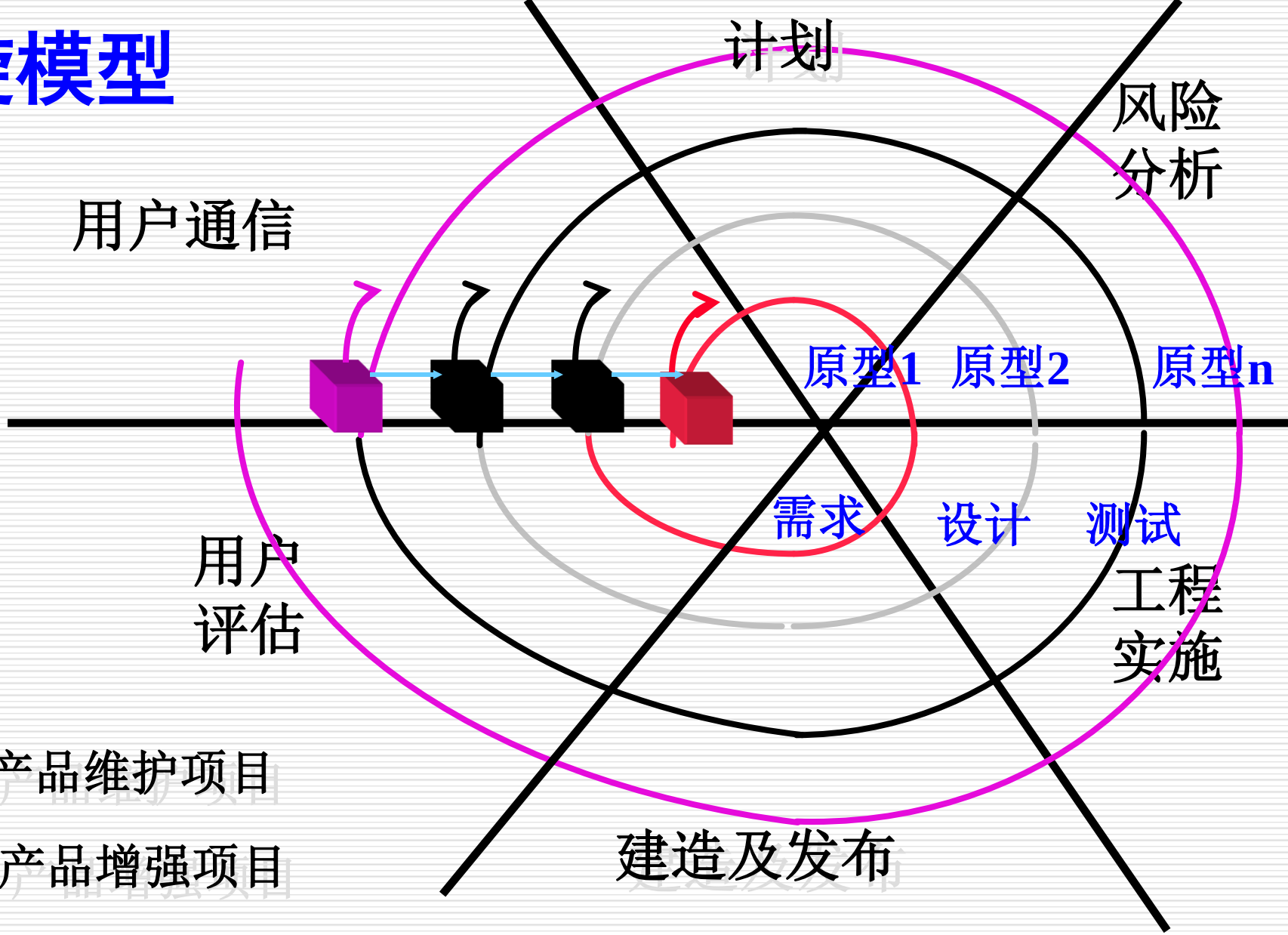
- 基本思想

✓ 关注软件开发分险（超过预算，员工流失，，竞争对手等）

✓ 使用原型及其他方法来尽量降低风险

✓ 软件过程每个阶段之前，进行分险分析

螺旋模型



产品维护项目

产品增强项目

新产品开发项目

概念开发项目

螺旋模型的特点

- ✓Combine prototype with linear model
- ✓Quick Incremental Model
- ✓Partitioned task area
- ✓risk driving （风险驱动）
- ✓cost control (成本可控)

Characteristics

Weak:

1、 需要相当的风险分析评估的专门技术，且成功依赖于这种技术。

2、 一个大的没有被发现的风险问题，将会导致问题的发生，可能导致演化的方法失去控制。

3、 这种模型应用不广泛，其功效需要进一步的验证。

Strong：

1、对于大型系统及软件的开发，这种模型是一个很好的方法。开发者和客户能够较好地对待和理解每一个演化级别上的风险。

其它过程模型

- 喷泉模型 P21
- Rational 统一过程 P22
- 敏捷开发过程与极限编程 P25
- 微软过程 P29

1.8 Methods of software design and development

- 面向机器编程
- 结构化方法 （structure method)
- 面向数据结构方法(data oriented)
- 面向对象方法 （object oriented)
- 面向构件方法 （component oriented ）
- 面向方面 （aspect oriented ）
- 面向agent
- 面向服务 （service oriented ）
- 基于搜索的软件编程开发方法

(1) structure oriented method

also named as process oriented method, or function oriented method

- **features:**

  proposed in 60’, matured in 70’, successful in 80’,

  basic, fortran, cobol, c language

  real-time, computer system software, control software,

  simple, easy to study

(2) object oriented method

Main concepts:

object, class, encapsulation, inheritance, message,

80’, 90’,

c++, java, UML, Rose

strong programming capability, easy to maintain,

interactive network/Internet application programs.

(3) data oriented method

80, 90, Oracle, Sybase, Informix, DB2

powerdesigner, OracleDesigner, 4GL,

easy to understand, MIS

1.9 Principles of Software Engineering

8 rules

- ✓(1) the life-cycle is divided separate phases, which manage software development.
- ✓(2) audit in every phase
- ✓(3)strictly performing product and version control
- ✓(4)employing modern program design technology
- ✓(5)program results should be inspected clearly, documents needed
- ✓(6)software development team has small members but competent.
- ✓(7)increasingly improving experience and technology day after day
- ✓(8)two-eight law

软件工程的 8项基本原则

(1) 用分阶段的生命周期计划严格管理软件工程过程

(2) 坚持在软件工程过程中进行阶段评审

(3) 实行严格的产品控制

(4) 采用现代的开发技术进行软件的设计与开发

(5) 工作结果应当是能够清楚地审查的

(6) 开发小组的人员应该“少而精”

(7) 承认不断改进软件工程实践的必要性

(8) 2-8定理

2-8定理

80% finished task, but 20% not in fact,

80% fault in 20% codes,

20% modules implement 80% functions,

20% persons solve 80% issues,

80% information function spend 20% investment.

Till today, these rules are effective/valid yet