软件工程基础 —— 第4章 过程模型

计算机学院 孟宇龙

第4章 过程模型

4.1 惯用（通用）过程模型

4.2 专用过程模型

4.3 统一过程

4.4产品和过程

关键概念

- 面向方面的软件
- 开发
- 基于构件的开发
- 并发模型
- 演化过程模型
- 形式化方法模型
- 增量过程模型
- 过程建模工具
- 原型开发
- 螺旋模型
- 统一过程
- V模型
- 瀑布模型

过程模型的意义

在找出在混乱世界中的秩序

和适应不断发生变化

这两种要求之间寻求平衡

4.1 惯用过程模型

过程模型的作用是减少开发新软件产品时出现的混乱

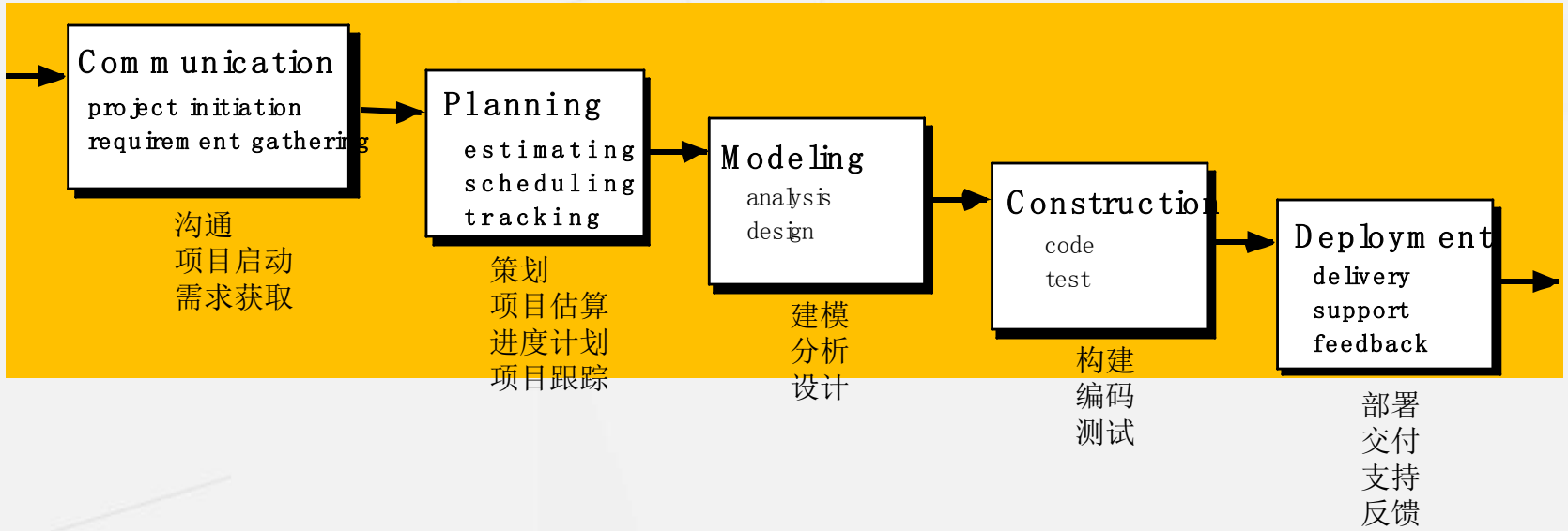
- **惯用过程模型**提倡有序的软件工程方法

  因此导致一些问题...

- 如果传统过程模型力求实现结构化和有序，那么对于富于变化的软件世界，这一模型是否合适呢？
- 如果我们抛弃传统过程模型（以及它们带来的秩序），取而代之以一些不够结构化的模型，是否会使如软件工作无法达到协调和一致？
- 无法简单回答，但软件工程师有很大的选择余地。

4.1.1 瀑布模型

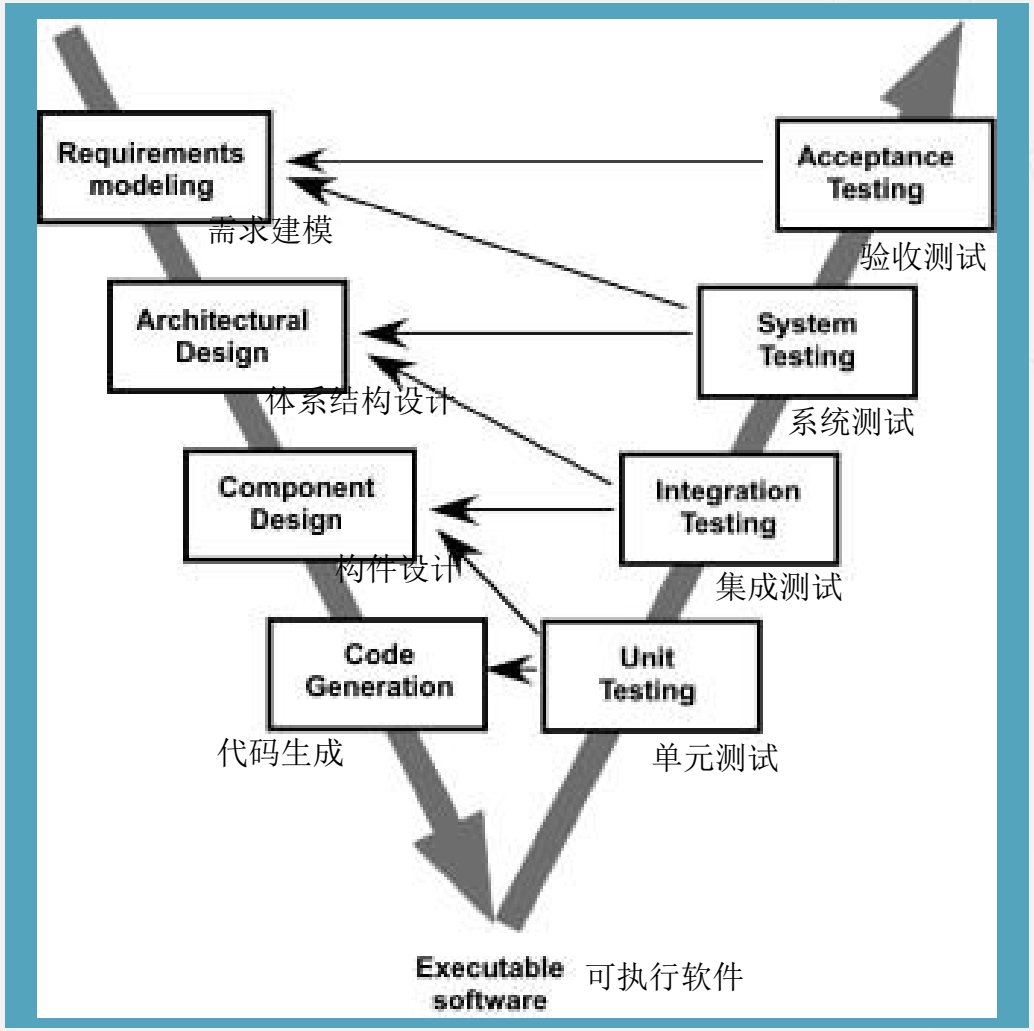
惯用过程模型定义了一组规定的过程元素和一个可预测的过程工作流



- 实际项目很少遵循这个顺序
- 客户通常很难描述需求，但是...
- 客户必须要有耐心，但是...

4.1.1 瀑布模型

V模型阐明了验证和确认动作如何与早期工程动作相互关系



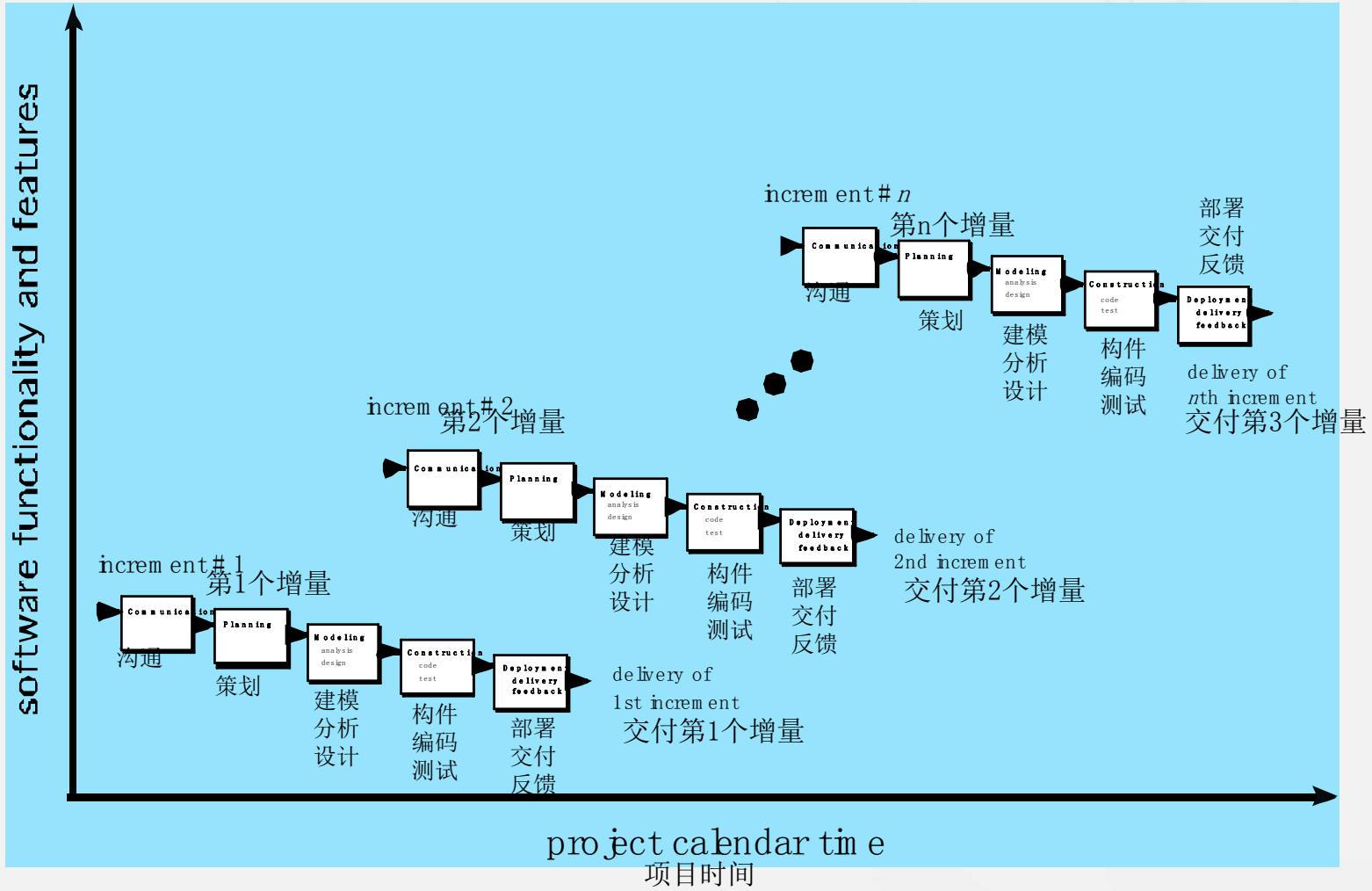
V模型

V模型提供了一种将验证和确认动作应用于早期软件工程工作中的直观方法

瀑布模型适用当需求确定、工作采用线性方式完成时，瀑布模型是一个有用的过程模型。

4.1.2 增量过程模型

增量过程模型交付一系列称为增量的版本，随着每个版本的交付，逐步为用户提供更多的功能



适用情形：

- **初始的软件需求明确，但是整个开发过程却不宜单纯运用线性模型。同时，可能迫切需要为用户迅速提供一套功能有限的软件产品，然后在后续版本中再进行细化和扩展功能。**
- **例如第一个增量往往是核心产品，附加功能进入下个增量计划。**

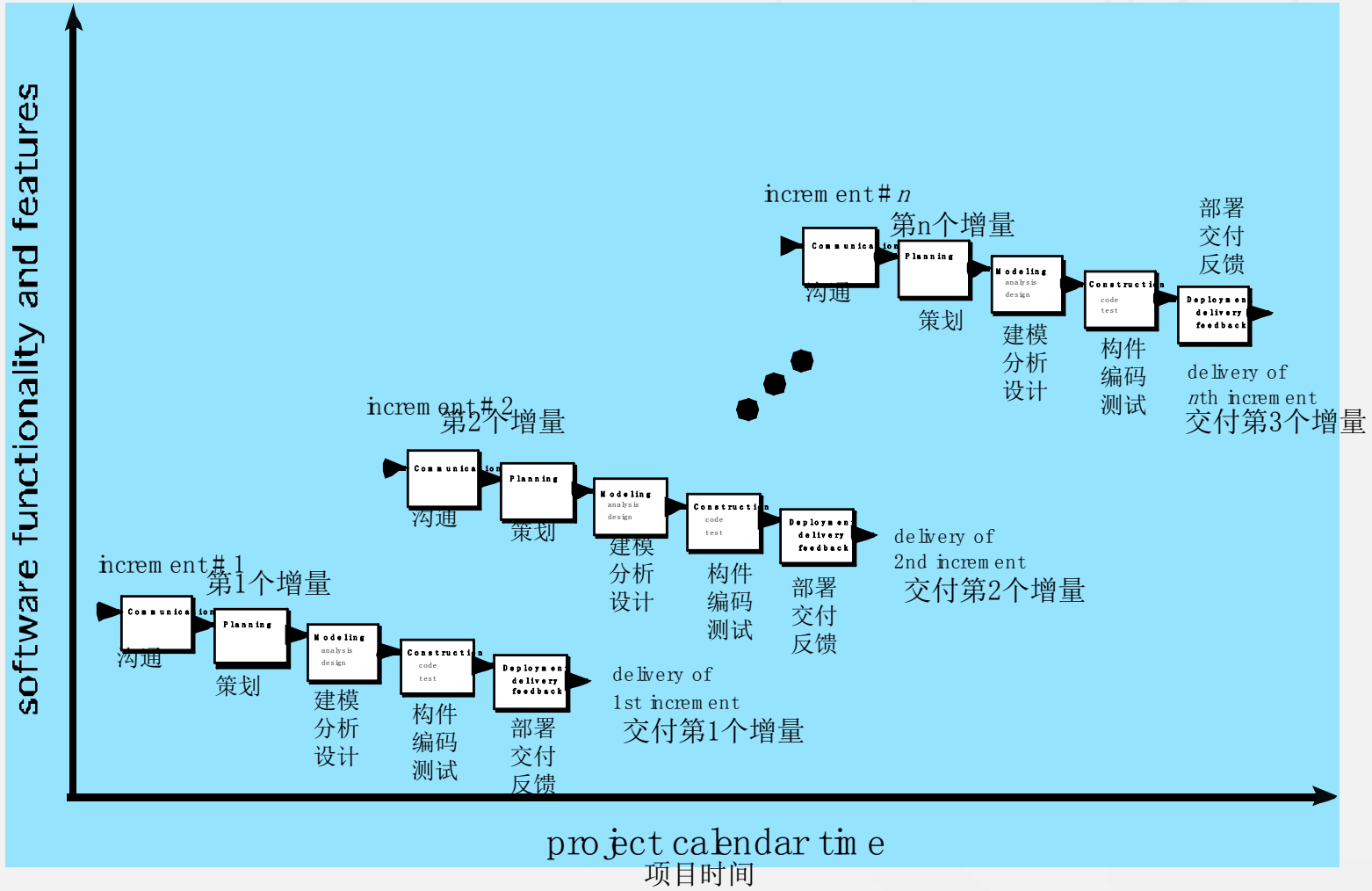
特点：

综合了线性过程流和并行过程流的特征。

每个增量都提交一个可以运行的产品。

4.1.2 增量过程模型

如果客户要求在一个不可能完成的时间提交产品，那么向他建议只提交一个或几个增量，此后再提交其它增量



例如：

- 文字处理软件，
  - 第一个增量是基本的文件管理、编辑和文档生成（核心功能）；
  - 第二个增量是复杂的编辑和文档生成；
  - 第三个增量是拼写检查和语法检查功能；
  - 第四个增量是高级页面排版功能；
  - 第2至4个增量是附加功能。

4.1.3 演化过程模型

有时候，虽然能很好地理解核心产品和系统序曲，但是产品系统扩展的细节还没有定义。

为什么？我们要怎么做？

4.1.3 演化过程模型

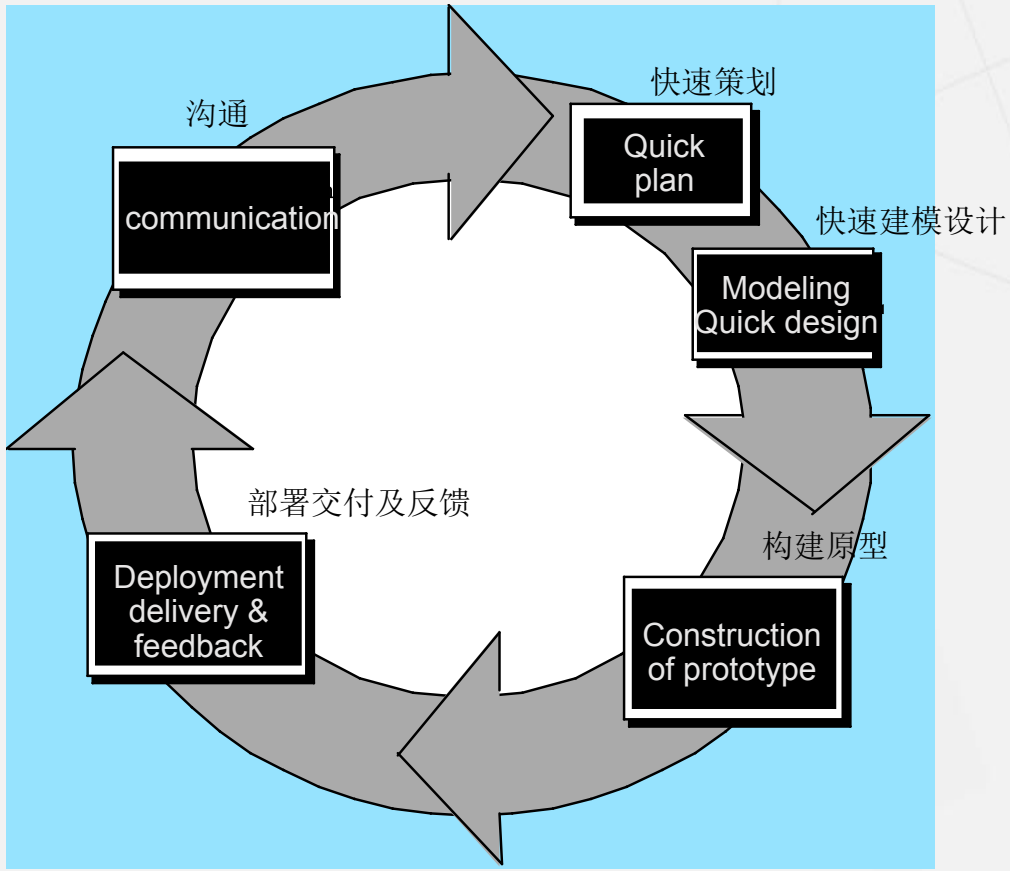
当需求很模糊的时候

原型开发模型

能帮助软件开发人员和利益相关者更好地理解究竟要做什么！

4.1.3 演化过程模型

演化过程模型中，每个迭代产生软件的一个更完整的版本



原型开发模型

原型开发：

- 客户提出了一些基本功能，但没有详细定义功能和特性需求
- 开发人员可能对算法的效率、操作系统的兼容性和人机交互的形式等情况并不确定

特点：

- 很少是好用的，可能太慢太大，难以使用。
- 一般作为被丢弃的系统。

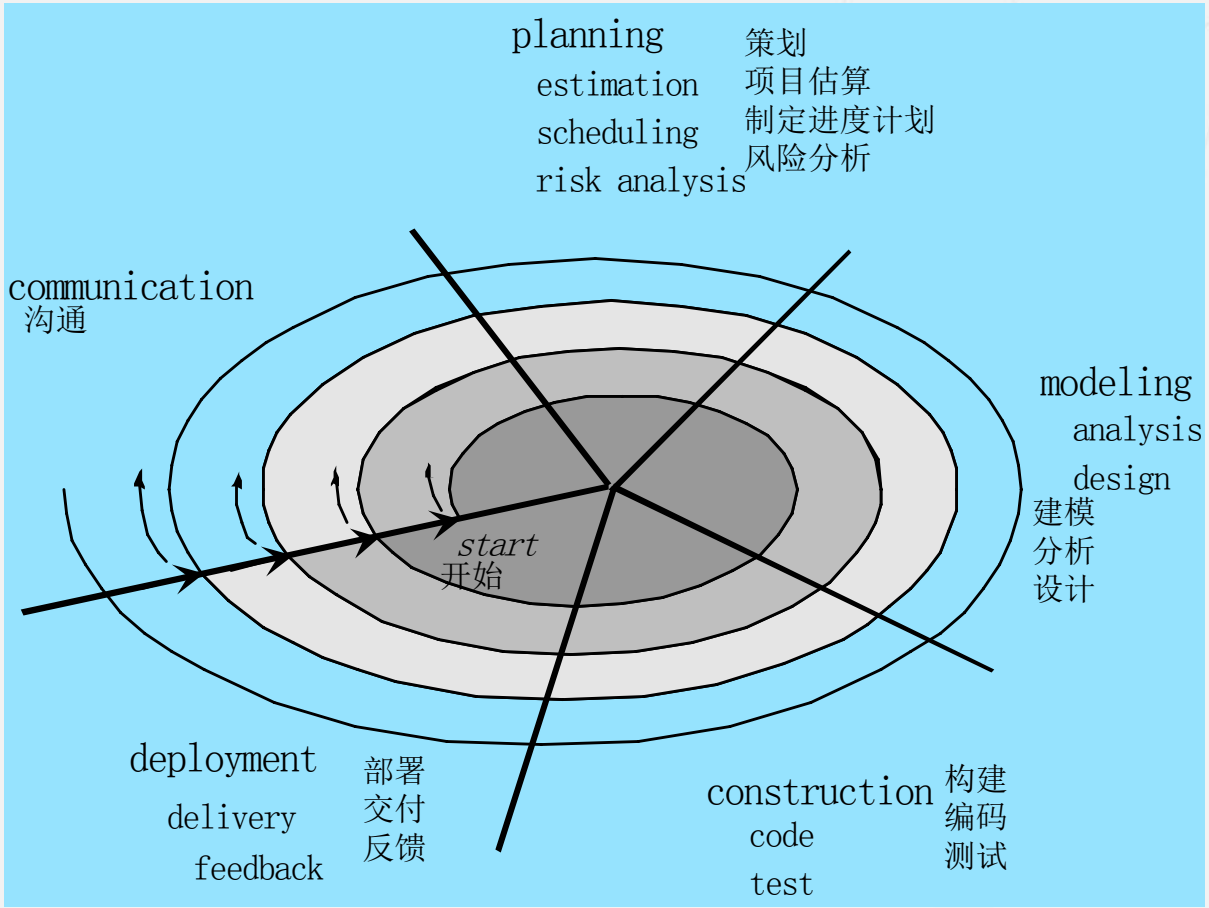
原型系统存在的问题可能是：

（1）整体软件质量不尽如人意

（2）可能会采用一种折衷手段而导致不完美

4.1.3 演化过程模型

螺旋模型能运用在应用系统开发的整个生命周期，从概念开发到维护



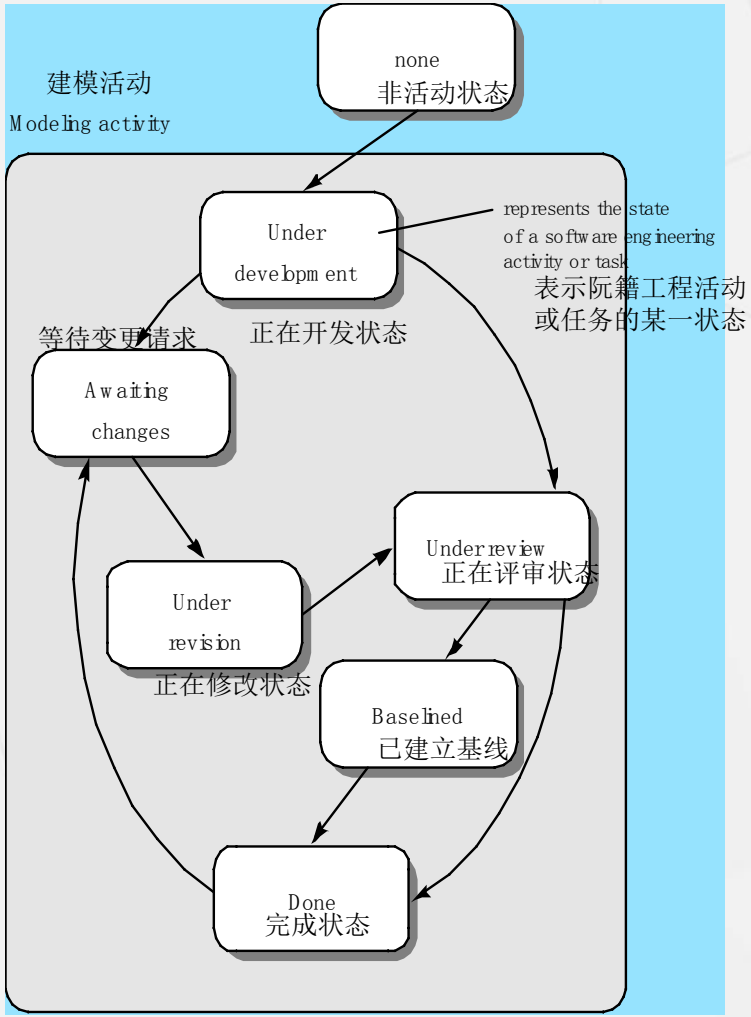
典型的螺旋模型

特点：

- 结合了原型的迭代性质、瀑布模型的可控性和系统性
- 是一种风险驱动型的过程模型生成器。对于软件集中的系统，可以指导多个利益相关者协同工作。
- 采用循环的方式逐步加深系统定义和实现的深度，同时降低风险。
- 确定一系列里程碑，确保利益相关者都支持可行的和令人满意的系统解决方案。
- 第一圈开发出系统的规格说明，第二圈开发出产品的原型系统，以后逐次完善，开发不同的软件版本。不断调整项目计划，根据交付用户的反馈调整预算和进度、以及迭代次数。

4.1.4 并发模型

必须将项目计划看成是活的文档，必须经常对进度进行评估并考虑变更情况，从而对其进行修改



协同过程模型的一个元素

特点：

- 在某一特定时间，建模活动可能处于图中所示的任何一种状态中。其他活动、动作或任务，可以用类似的方式表示。
- 所有的软件工程活动同时存在并处于不同的状态。
- 并发建模定义了一系列事件，这些事件将触发软件工程活动、动作或者任务的状态转换

其他过程模型

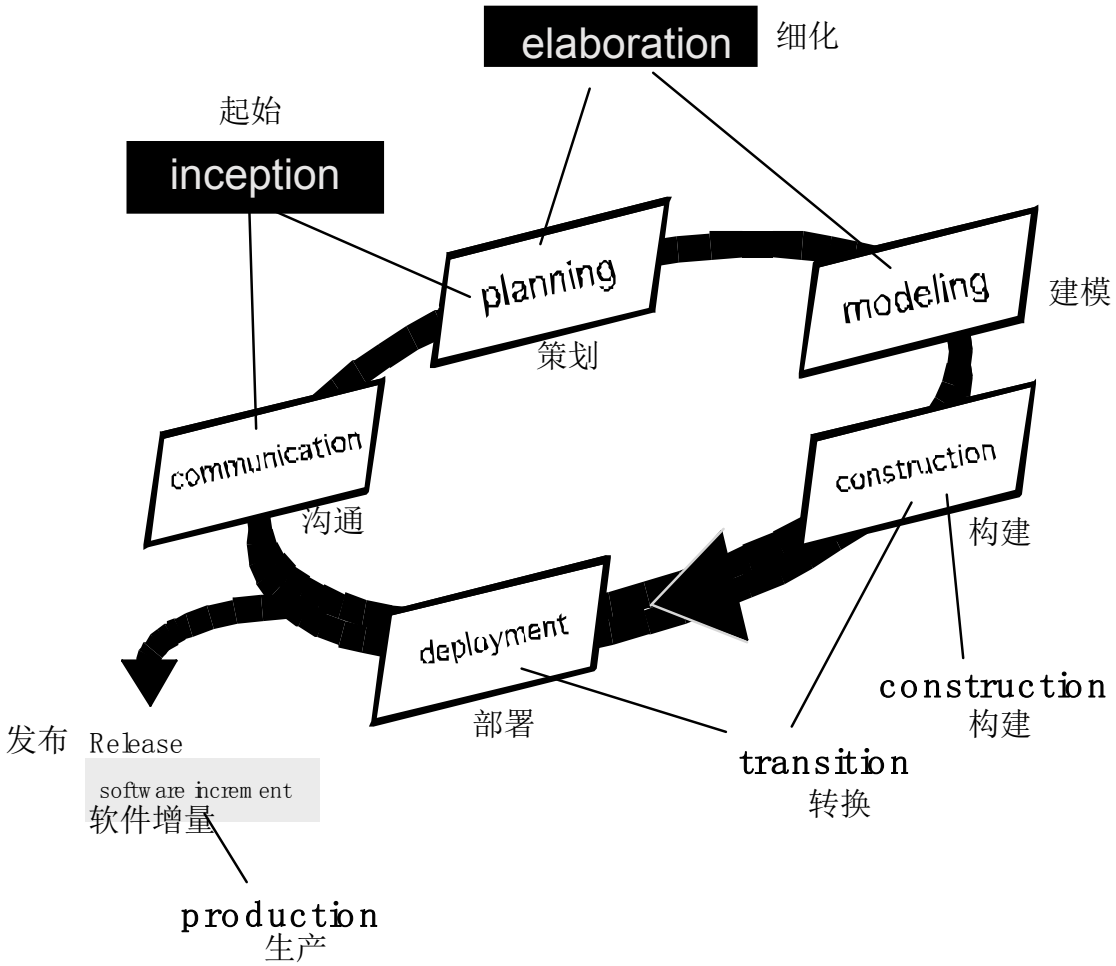
- 基于构件的开发——这个过程模型能够使软件复用，是一个发展目标
- 形式化方法——强调需求的数学规范说明

  一个变型是净室（ cleanroom)软件工程。
- 面向方面的软件开发（AOSD）—为定义、说明、设计和构建方面提供过程和方法

  如果某个关注点涉及系统多个方面的功能、特性和信息，可称为横切关注点，由方面性需求来定义。
- 统一过程——一种“用例驱动，以架构为核心，迭代并且增量”的软件过程与统一建模语言的紧密结合

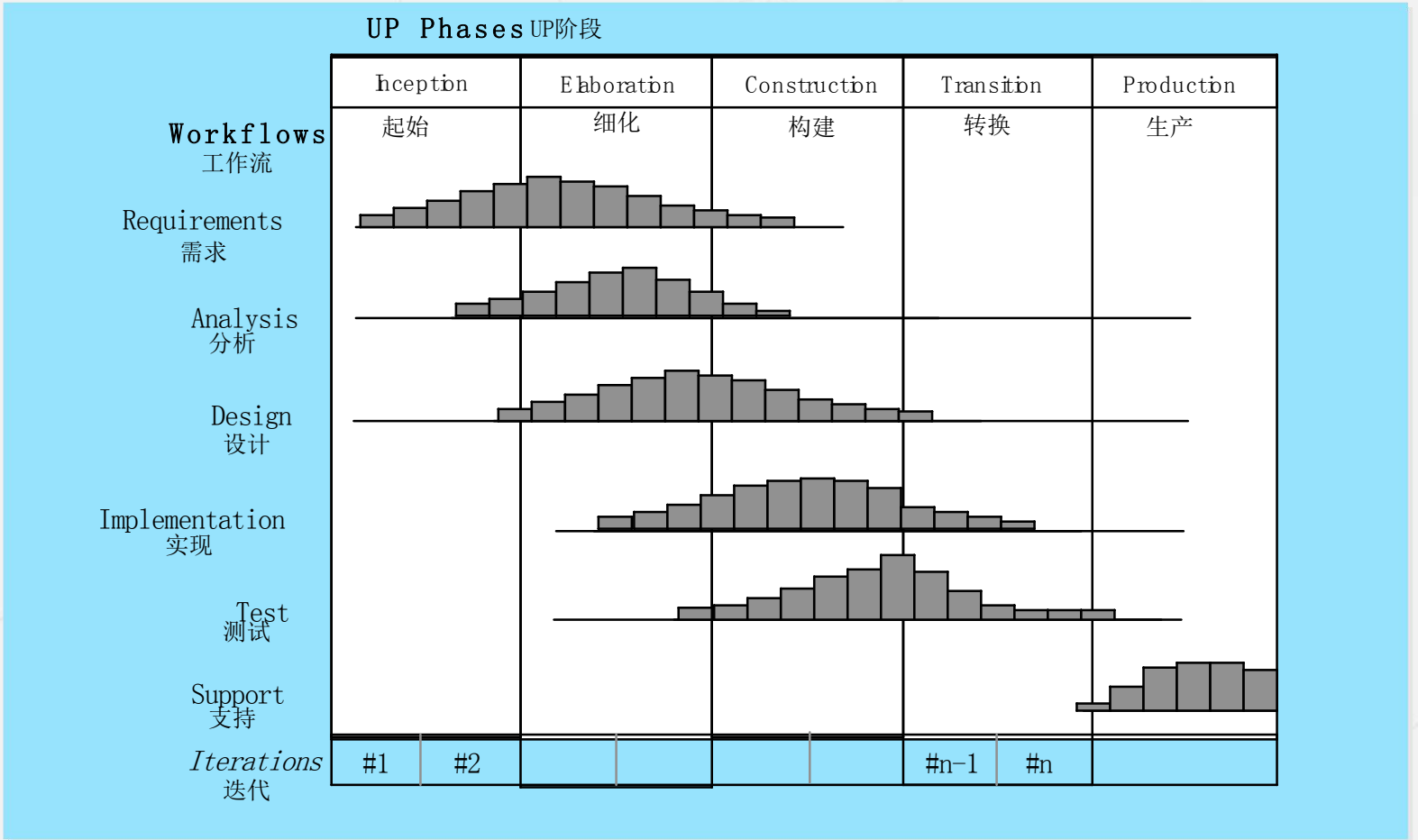
4.3 统一过程

UML


elaboration
细化
起始
inception
planning
策划
modeling
建模
communication
沟通
construction
构建
deployment
部署
发布
Release
software increment
软件增量
production
生产
transition
转换
construction
构建

4.3.2 统一过程的阶段

UP阶段的目的与通用框架活动的目的是类似的



其他过程模型

UP工作产品

起始阶段

愿景文档
初始用例模型
初始项目术语
初始商业案例
初始风险评估
项目计划，
阶段和迭代。
商业模型，如果必要
一个或更多的原型开发

细化阶段

用例模型
包括非功能性的补充需求
分析模型
软件体系结构描述
可执行体系结构原型
初步设计模型
修订风险列表
包含迭代计划的项目策划
调整工作流里程碑
技术工作产品
初步用户手册

构建阶段

设计模型
软件构件
集成软件
增量
测试计划和步骤
测试用例
支持文档
用户手册
安装手册
当前增量说明

转换阶段

交付软件增量
Beta测试报告
一般用户反馈

其他过程模型

个人软件过程（PSP）

- **策划。**这个活动将需求活动分离出来，估算项目的规模和所需资源，并估算缺陷（工作中预测的缺陷数目）。所有的度量都用工作表或是模板记录。最后，识别开发任务，并建立项目进度计划。
- **高层设计。**搭建每个构件的的外部规格说明，并完成构件设计。如果有不确定的需求，则建立原型系统。所有问题都要记录和跟踪。
- **高层设计评审。**使用正式的验证方法（参见第21章）来发现设计中的错误。对所有的重要任务和工作结果都进行度量。
- **开发。**细化和评审构件级设计。完成编程，对代码进行评审，并进行编译和测试。对有的重要任务和工作结果都进行度量。
- **后验。**根据收集到的测量和度量结果（需要进行大量数据的统计分析），确定过程的有效性。度量和测量结果为提高过程的有效性提供指导。

其他过程模型

团队软件过程（TSP）

- 建立自我管理团队来计划和跟踪他们的工作、确定目标、建立团队自己的过程和计划。团队既可以是纯粹的软件开发队伍，也可以是集成的产品队伍（IPT），可以由3~20名工程师组成。
- 指示管理人员如何指导和激励其团队，及如何帮助他们保持团队的最佳表现。
- 使CMM第5级的行为常规化并如预期一样，这样可加速软件过程改进。
  - 能力成熟模型(CMM)，一种衡量软件过程效率的技术，将在第30章中讨论。
- 为高成熟度的软件组织提供改进知道。
- 协助大学传授工业级团队技能。

?

A

PSP 能使您成为一名更好的专业软件工程师

B

PSP 能提高工作效率与编码速度

VS

A

TSP 使产品交付时每万行代码的缺陷数小于 1

B

TSP 能大大缩短系统测试时间

C

TSP 使领导对交货日期和开发成本了如指掌

4.4 产品和过程

- 二象性理论
- 所有人类活动都可以看作是一个过程
- 将复用目标融入软件开发