软件过程与质量控制

软件过程规范和模型

李娟
lijuan@bjut.edu.cn

大纲

- 1.过程规范
- 2.过程模型

1.1过程规范定义

- 规范
  - 用来控制或治理一个团队的一系列的准则与章程，团队里的成员必须遵守这些规章制度。
- 过程规范
  - 是人们在过程活动中需要遵守的约定和规则，包括已定义的操作方法、流程和文档模板。
  - 软件过程规范在整个软件开发的过程中约束着相关人员按照预定开发流程进行工作。

1.2规范过程的特点

- 有明确的定义
- 有相应的培训
- 具有强制性与可服从性
- 可不断改进

1.3过程规范实例

- **软件公司软件开发过程规范

1.4过程规范的优点

- 可以使团队形成统一协调的工作方式
- 可以提高团队的工作能力
- 在关键时刻是决定生死的因素

问题

过程规范是否具有抑制性？

过程规范是否阻碍创造力发挥 ？

1.5过程规范制定

- 要让过程的执行者参与到过程的设计中
- 过程的执行者可以提供反馈意见

1.6过程规范描述

- 典型的过程描述要点
- 过程描述举例
- 选择过程表达方式

典型的过程描述要点

- 1. 目的
- 2. 高层次的图形
- 3. 参与的角色
- 4. 活动列表
- 5. 产品和评审描述
- 6. 将要使用的资产
- 7. 此过程的裁减
- 8. 附注细节
- 9. 生命周期是一个阶段化的连续过程（迭代的或线性的或……）

典型的过程描述要点

- 阶段描述举例
  - 1. 本阶段目的
  - 2. 阶段主要输入
  - 3. 本阶段主要活动列表
  - 4. 主要发布产品或成果的列表
  - 5. 参与角色的列表
  - 6. 阶段出口准则，通常作为“关卡”
  - 7. 阶段裁减指导
    - 用于选择的可替代的及准则
    - 例外的准则

典型的过程描述要点

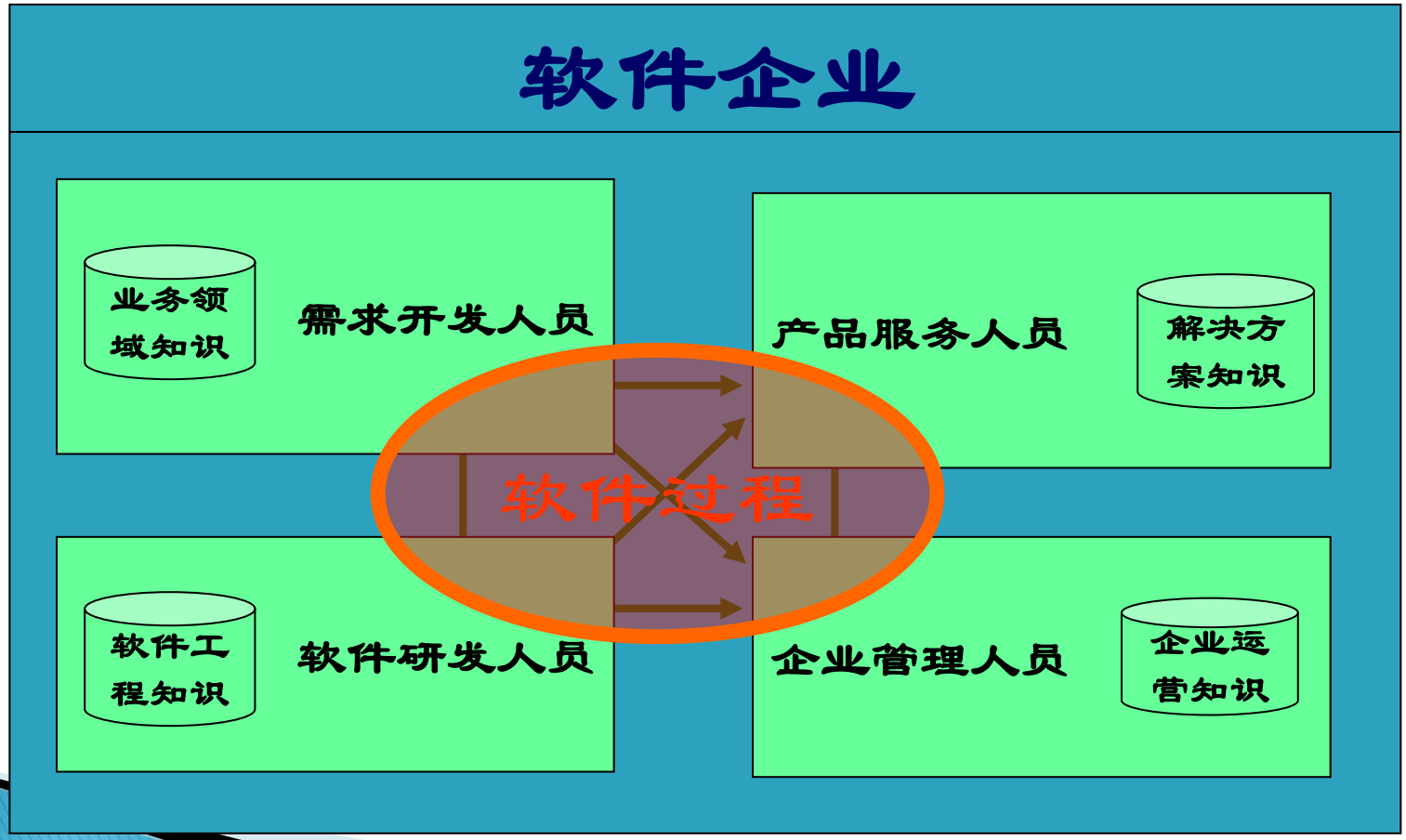
- 活动描述
  - 1. 目的
  - 2. 细化的图表描述－由谁做什么
  - 3. 活动或任务负责人
  - 4. 关键任务或任务组
  - 5. 入口和出口条件
  - 6. 涉及的角色
  - 7. 其他在某些时候包括的项
  - 8. 经验教训
  - 9. 收集的测量
  - 10. 验证和确认活动

2软件过程

- 2.1软件过程概念
  - 软件过程是开发和维护软件及其相关产品所涉及的一系列活动。即软件生存周期所涉及的一系列相关过程。

2软件过程

- 2.2软件过程在软件企业中的作用



2软件过程

- 2.3软件过程的核心元素
  - 过程规范
  - 过程培训
  - 过程的监控与强制
  - 过程中的角色与职责

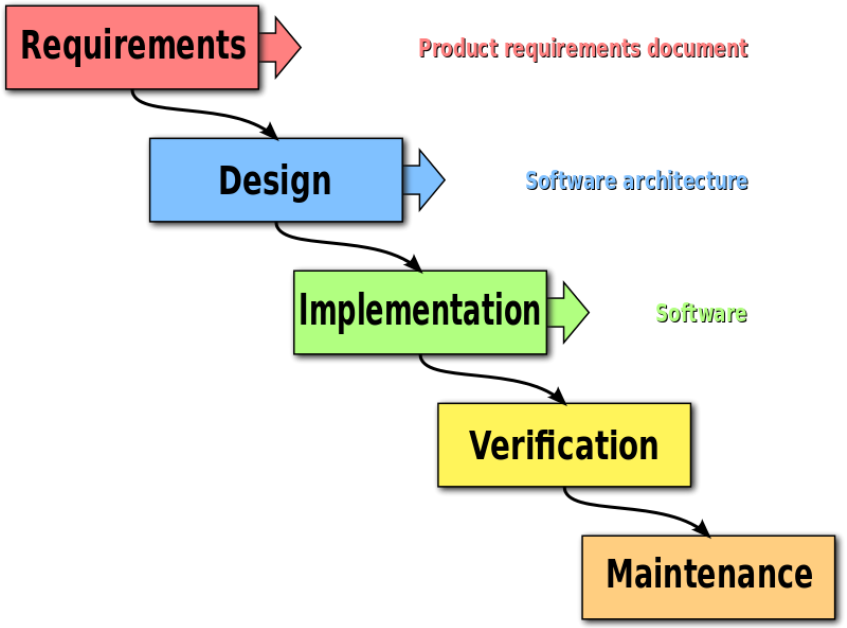
2软件过程

- 2.4软件过程模型
  - 软件过程模型是软件过程的抽象表示。
  - 一个软件过程模型是软件开发全部过程、活动和任务的结构框架。它能直观表达软件开发全过程，明确规定要完成的主要活动、任务和开发策略。
  - 软件过程模型也常称为软件开发模型。

2软件过程

- 2.5软件过程模型
  - 瀑布模型（Waterfall Model）
  - 增量模型（Incremental Model）
  - 迭代模型（Iterative Model）
  - 螺旋模型（Spiral Model）
  - 统一软件开发过程模型（Rational Unified Model ）
  - 原型模型（Prototype Model）
  - 敏捷模型 （Agile Model）

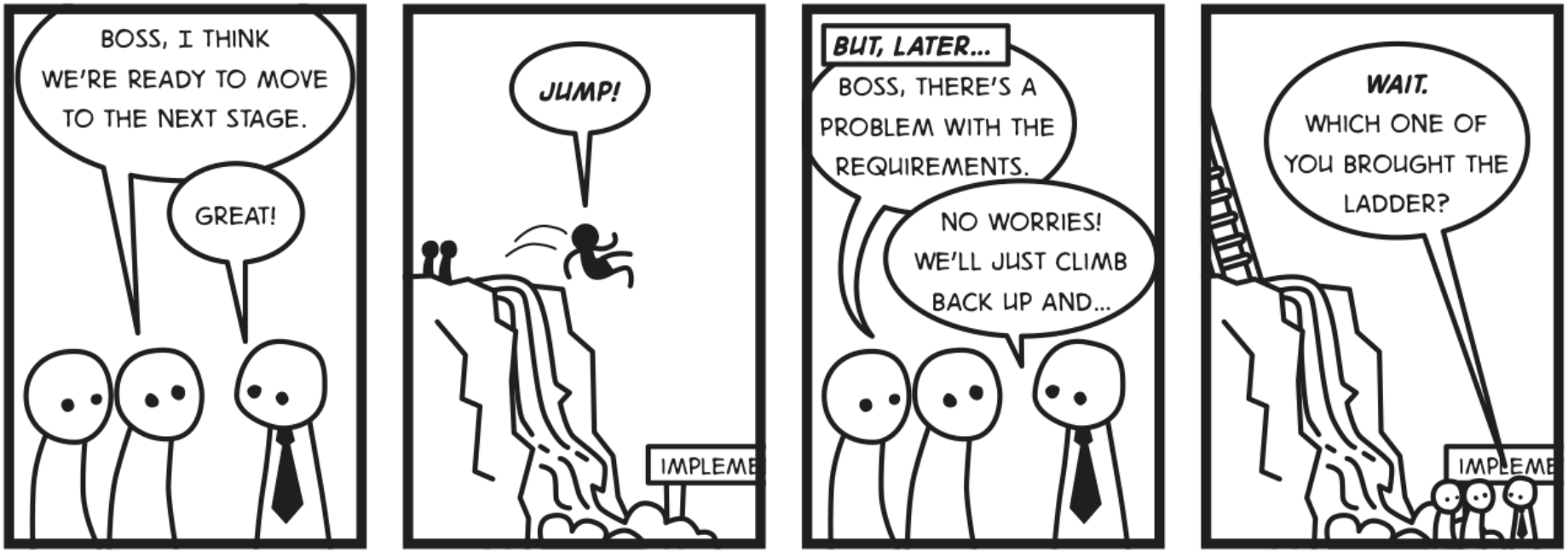
瀑布模型



- 将软件生存周期的各项活动规定为按固定顺序而连接的若干阶段工作，形如瀑布流水，最终得到软件产品
- 瀑布模型是最早出现的软件开发模型，提供了软件开发的基本框架，在软件工程中占有重要的地位
  - 1970年Winston Royce提出了著名的“瀑布模型”，直到80年代早期，它一直是唯一被广泛采用的软件开发模型。

瀑布模型

- 优点
  - Time spent early in the software production cycle can lead to greater economy at later stages
  - 强调文档化，非常重视需求文档、设计文档以及源代码，有利于新成员迅速全面地了解项目信息，促进人员沟通协作
  - 模型简洁，提供结构化方法，易于理解和使用
- 缺点
  - 由于开发模型是线性的，用户只有等到整个过程的末期才能见到开发成果，从而增加了开发风险。
  - 早期的错误可能要等到开发后期的测试阶段才能发现，进而带来严重的后果。
  - 不适应用户需求的变化
- 适用于需求明确的项目


BOSS, I THINK WE'RE READY TO MOVE TO THE NEXT STAGE.
GREAT!
JUMP!
IMPLEME
BUT, LATER...
BOSS, THERE'S A PROBLEM WITH THE REQUIREMENTS.
NO WORRIES! WE'LL JUST CLIMB BACK UP AND...
WAIT.
WHICH ONE OF YOU BROUGHT THE LADDER?
IMPLEME

瀑布模型角色

- 瀑布模型中的角色及职责

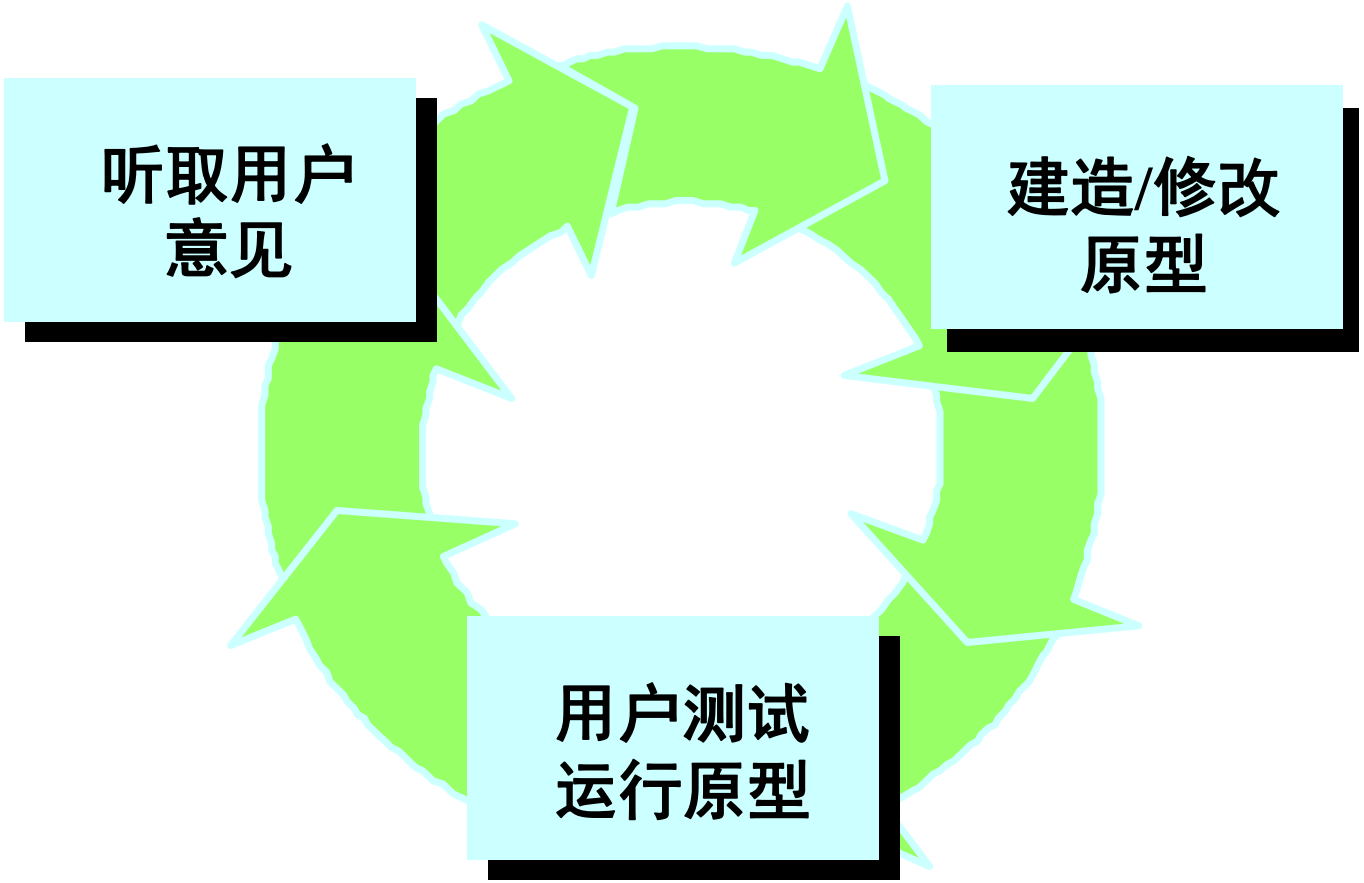
原型模型

- 目标系统的简化版本，描述了系统的主要功能
- 模拟真实系统的运行
- 有一个可运行的子集，用户可以评价需求和设计
- 可包括部分用户界面，通过人机交互，让双方了解目标系统的操作。
- 利用线性系统，不断演化形成最终系统
- 降低了成本，及早发现错误、修改量小、开发周期短

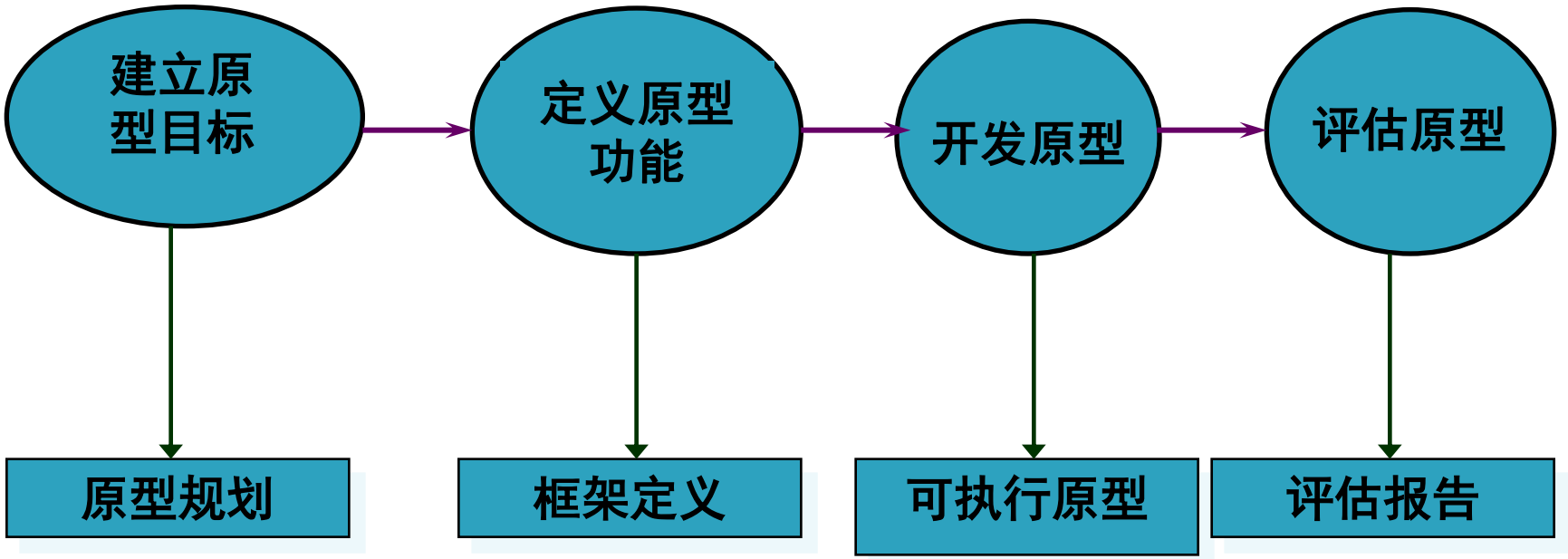
什么时候使用原型

- 当用户不清楚想要什么软件的时候，
- 当开发人员不确定完全理解需求时

原型模型



原型开发过程



原型模型

- 分类
  - 原型是项目系统中的一个方面或者多个方面的工作模型。
  - 抛弃型原型
    - 用于试验某些概念，试验完系统将无用处
  - 进化型原型
    - 原型系统不断被开发和被修正，最终它变为一个真正的系统

原型模型

- 特点
  - 原型驱动
  - 优点：
    - 从实践中学习
    - 改善沟通和用户参与
    - 使部分已知需求清晰化
    - 展示描述的一致性和完整性
    - 提高系统的实用性、可维护性

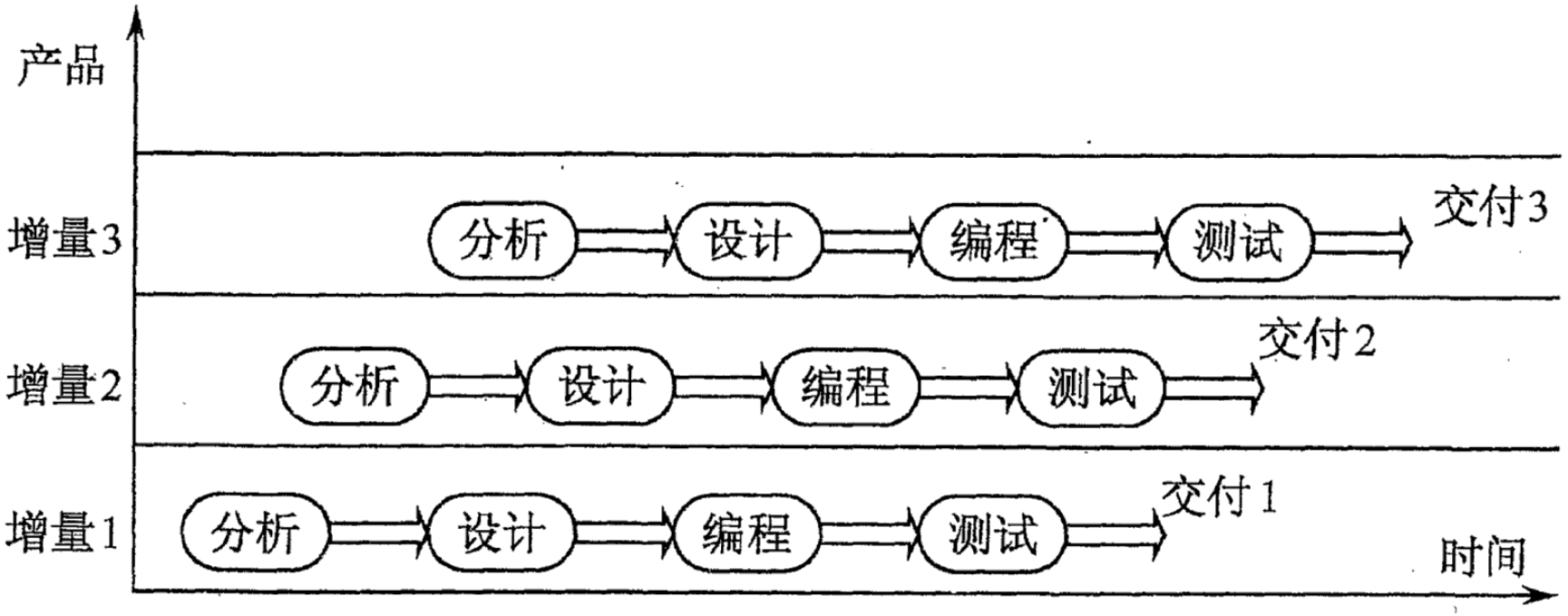
原型模型

- 缺点
  - 用户有时误解了原型的角色
    - 例如他们可能误解原型应该和真实系统一样可靠
  - 缺少控制，由于用户可能不断提出新要求，因而原型迭代的周期很难控制。
  - 额外的花费
    - 研究结果表明构造一个原型可能需要10%额外花费
  - 原型法要求开发者与用户密切接触，有时这是不可能的
    - 例如外包软件
  - 用户与开发者两方面必须达成一致
    - 原型被建造仅是为了定义需求，之后被抛弃（或至少部分被抛弃），实际的软件在充分考虑了质量和可维护行之后才被开发

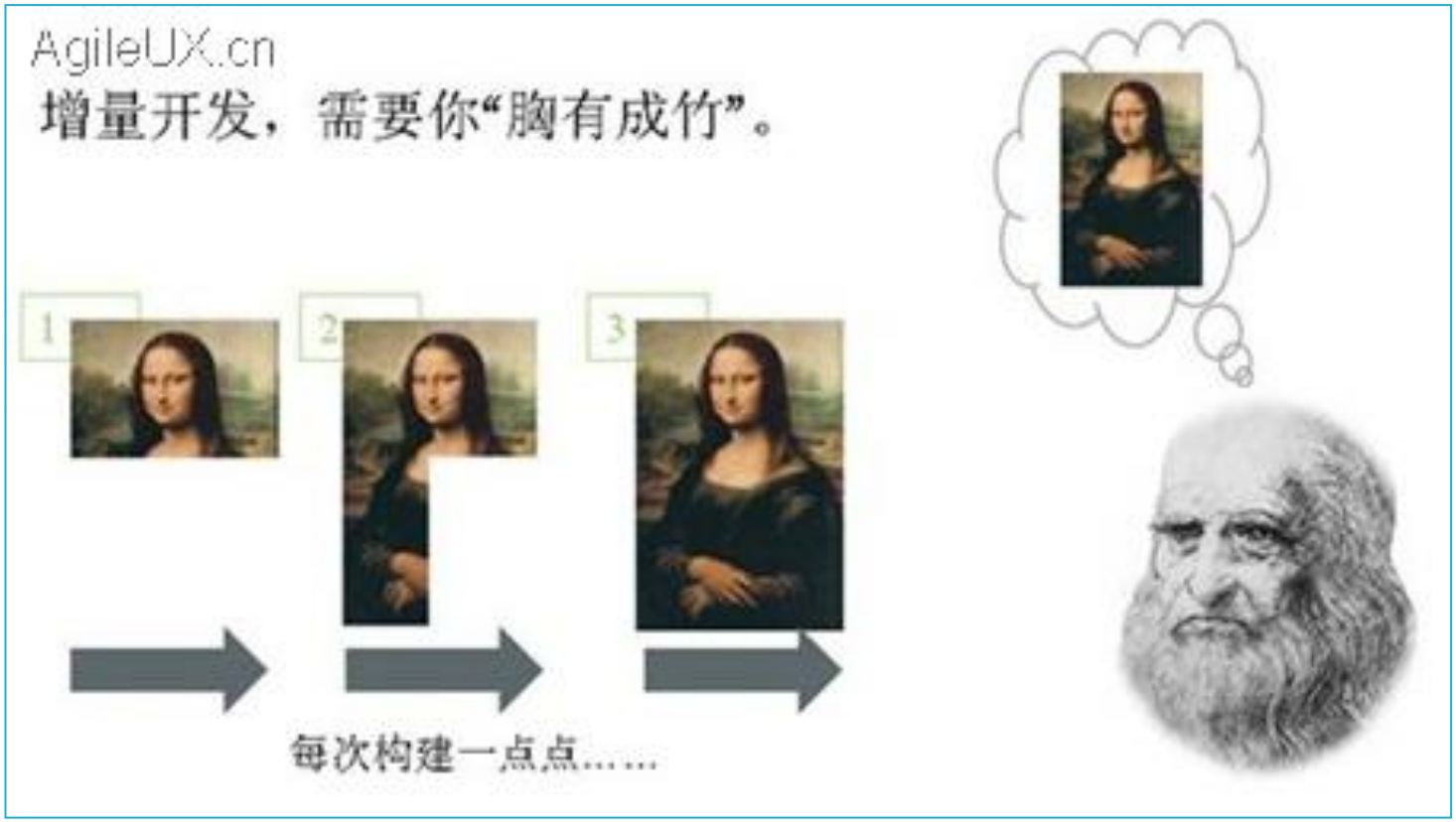
增量模型

- 采用随着日程时间的进展而交错的线性序列，每一个线性序列产生软件的一个可发布的“增量”。
- 引进了增量包的概念，无须等到所有需求都出来，只要某个需求的增量包出来即可进行开发。

增量模型



增量模型



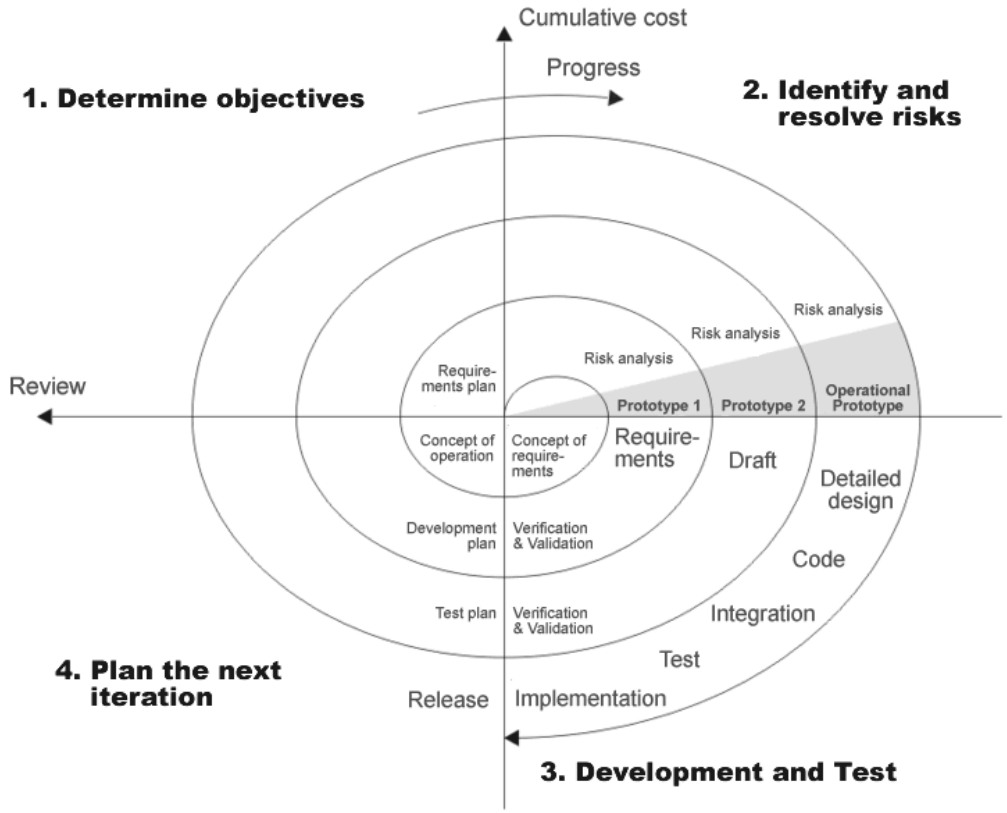
迭代模型

- 最早出现在20世纪50年代末期
- 所有的阶段都可以细分为迭代。每一次的迭代都会产生一个可以发布的产品，这个产品是最终产品的一个子集。

迭代模型

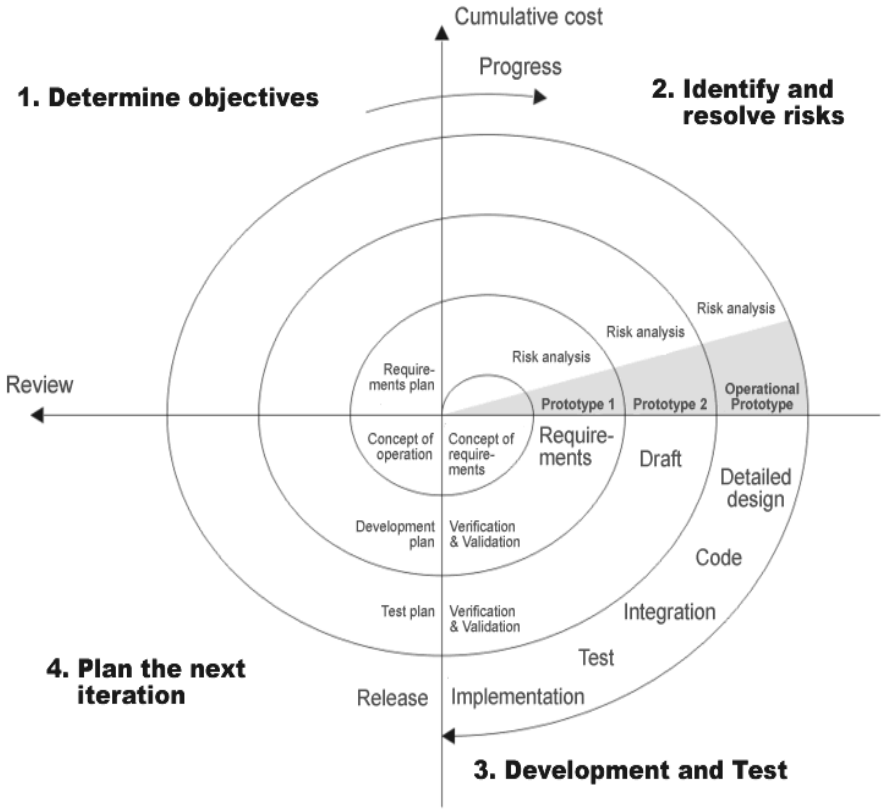
螺旋模型

- 1988年，Barry Boehm正式发表了软件系统开发的“螺旋模型”
- 将瀑布模型和快速原型模型结合起来，强调了其他模型所忽视的风险分析，特别适合于大型复杂的系统。



螺旋模型

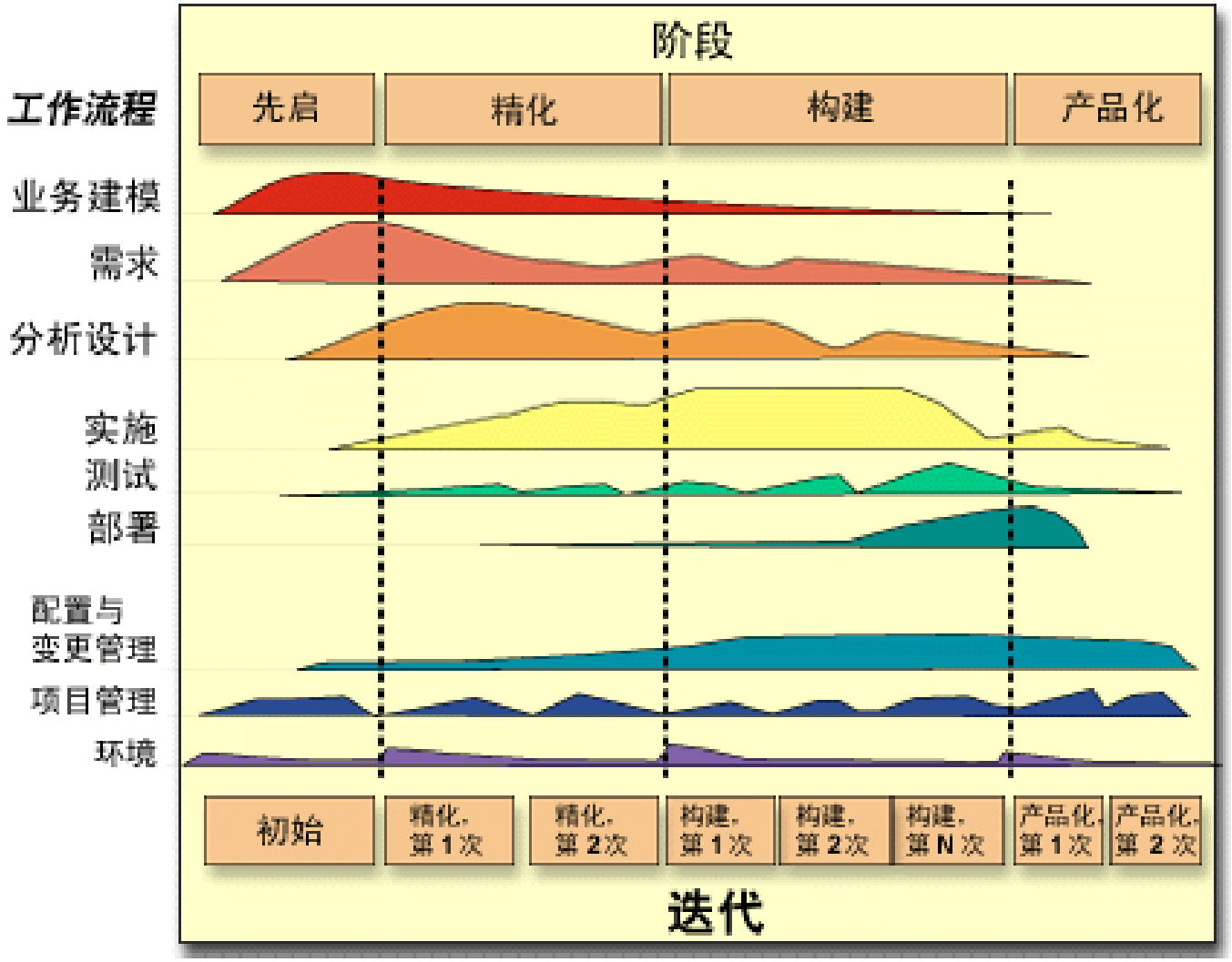
- 四个象限
  - 确定目标：确定软件目标，选定实施方案，弄清项目开发的限制条件；
  - 风险分析：分析评估所选方案，考虑如何识别和消除风险；
  - 开发测试：实施软件开发和验证；
  - 下次迭代计划：评价开发工作，提出修正建议，制定下一步计划。
- 每轮循环
  - 1. 确定目标，可选项，以及强制条件。
  - 2. 识别并化解风险。
  - 3. 评估可选项。
  - 4. 开发并测试当前阶段。
  - 5. 规划下一阶段。
  - 6. 确定进入下一阶段的方法步骤



螺旋模型

- 优点
  - 强调风险分析，使得开发人员和用户对每个演化层出现的风险有所了解，继而做出应有的反应，因此特别适用于庞大、复杂并具有高风险的系统。
  - 客户始终参与每个阶段的开发,保证了项目不偏离正确方向以及项目的可控性。
  - 螺旋模型由风险驱动，强调可选方案和约束条件从而支持软件的重用，有助于将软件质量作为特殊目标融入产品开发之中。
- 缺点
  - 建设周期长
- 适用于需求不明确的项目，便于风险控制和需求变更

统一软件开发过程模型RUP



RUP简介

- RUP-Rational Unified Process, Rational 统一过程
- RUP是一个软件的开发过程，将用户需求转化为软件系统所需的活动的集合
- RUP是一个通用的过程框架，可用于各种不同类型的软件系统，各种不同的应用领域，各种不同功能级别以及各种不同的项目规模。

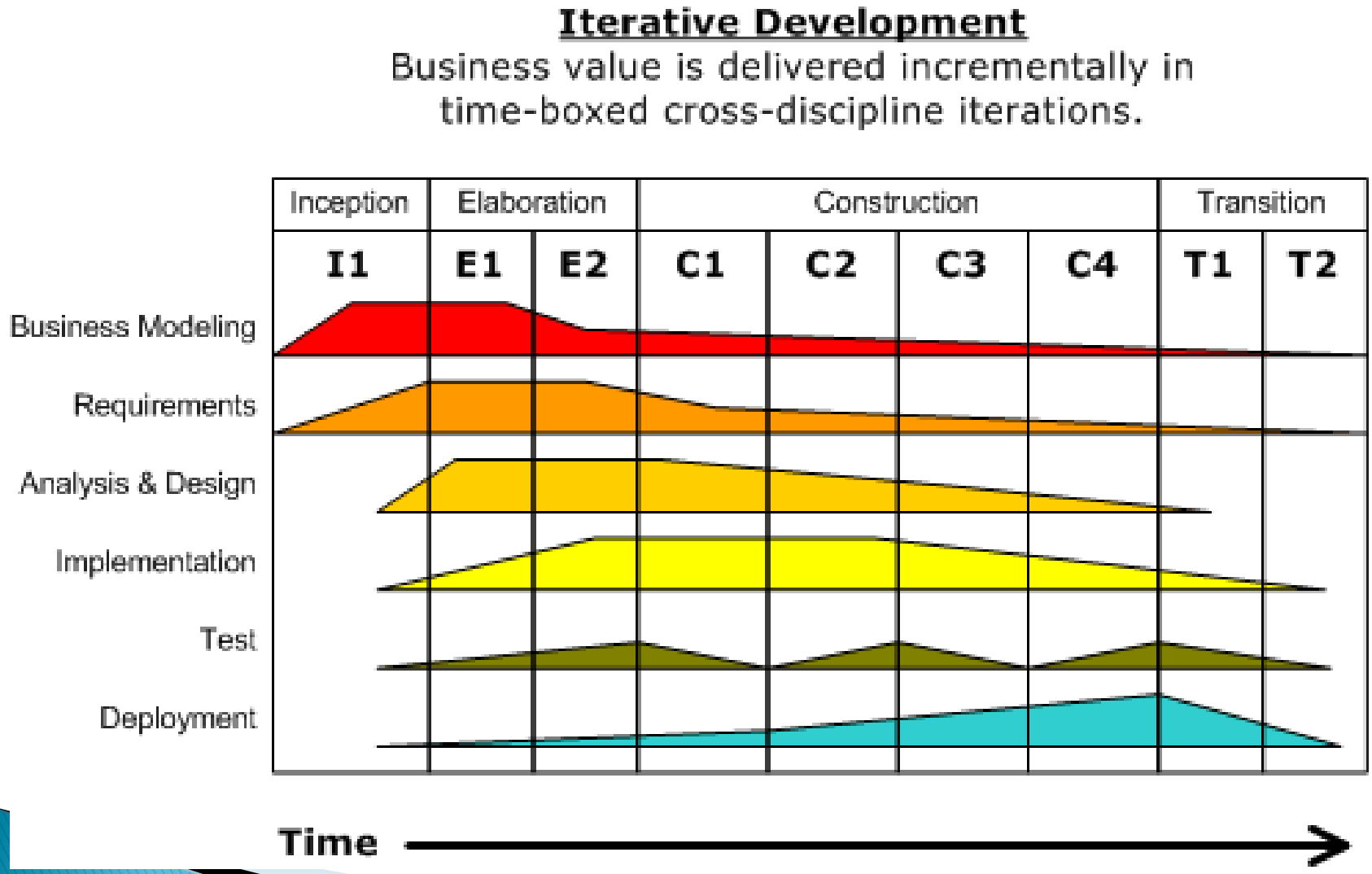
背景

- 20世纪90年代，Rational 统一软件过程（RUP）作为一个集结了软件工程最佳实践的框架，被逐步建立起来。
  - 其中的一些理念，如迭代、简单、关注价值和定期反馈，都被认为对软件工程的成功至关重要。很多人都借鉴统一过程，在不同的项目领域构建了方法论。
- 设计Rational统一软件过程的初衷是想为软件工程提供一个框架。
- 1997年，Rational软件公司宣称已经整合了7类最佳实践：
  - 1.迭代式开发，并以风险作为迭代的主要驱动因素
  - 2.管理需求
  - 3.运用基于组件的构架
  - 4.软件可视化建模
  - 5.持续质量验证
  - 6.变更控制
  - 7.定制化
- Rational构建的框架包含相关过程和实践，可以运用在商业产品开发中，它主要包括3部分：
  - 一个指导开发的可裁剪的过程
  - 一些能使这些过程应用自动化的工具
  - 一些能加快实施过程和工具的服务
- 2003年，IBM收购了Rational软件公司，并继续对RUP进行商业开发和包装

概述

- 基于构件的
- 用UML制定软件系统所有蓝图
- 突出特点：用例驱动、以架构为中心、迭代和增量过程
- 四个阶段：初始、细化、构造、移交
- 五个核心工作流：需求获取、分析、设计、实现、测试

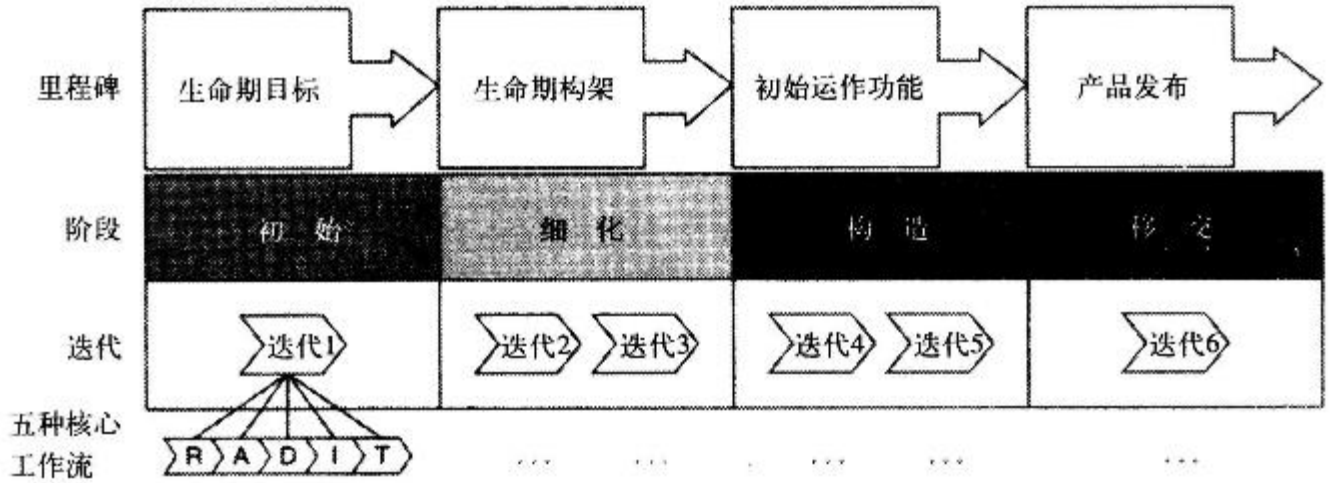
主要框架



http://en.wikipedia.org/wiki/IBM_Rational_Unified_Process

四个阶段

- 每个阶段开始都有特定目标，结束时有里程碑。每个阶段中存在一个或多个迭代。在每个迭代中，可以有多个工作流。



四个阶段

- 初始阶段
  - 目标：确定项目范围和边界条件，识别系统关键用例，展示系统的候选架构，估计整个项目需要的费用和时间安排评估项目风险
  - 重点：需求分析和系统分析，如需要构造原型系统，需做一些设计与实现。
- 细化阶段
  - 目标：迅速定出实用的架构，规划完成项目活动，估算完成项目所需资源，细化初始阶段模型，为构造阶段定出准确的计划
  - 重点：关注需求、分析和设计工作流

四个阶段

- 构造阶段
  - 目标：实现所有用例，制定移交阶段的准确计划，实现关键特性和核心功能，制定产品发布的验收标准，制定初步的用户手册，进行产品质量的详细分析
  - 重点：关注系统的实现工作流
- 交付阶段
  - 目标：试用产品并改正试用中发现的缺陷，制作安装版并培训客户，提供在线支持
  - 重点：关注系统的测试和配置工作流

五个核心工作流

- 工作流是由活动组成的序列，开发过程可以被分为五个核心工作流
  - 需求获取
  - 分析
  - 设计
  - 实现
  - 测试
- RUP的一次迭代包括五个核心工作流，每个阶段经过多次迭代，但每个阶段重点不同。

Best practices in RUP (Fitzgerald & al. 2002)

Best practice	Implemented in RUP through
Develop software iteratively	Frequent executable requirements
Manage requirements	Use cases to capture functional requirements
Use component-based architectures	Architecture definition using components
Model software visually	Unified Modelling Language (UML)
Verify quality continuously	Testing throughout lifecycle
Control changes to software	Change isolation and procedures

三大特点

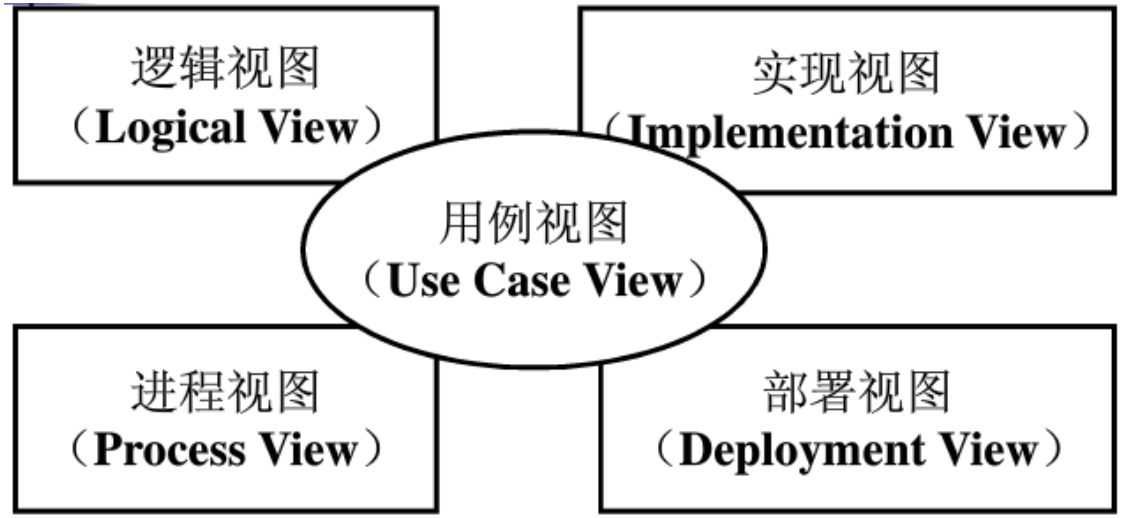
- 用例驱动
- 以体系结构为中心
- 迭代和增量

用例驱动

- Use Case表示需求
- Use Case贯穿整个软件开发周期
  - 需求、设计、开发、测试

体系结构为中心

- 软件体系结构是关于构成系统的元素、元素间交互、元素和元素间组成模式以及作用在这些模式上的约束等方面的描述
- 采用4+1视图模型描述软件体系结构



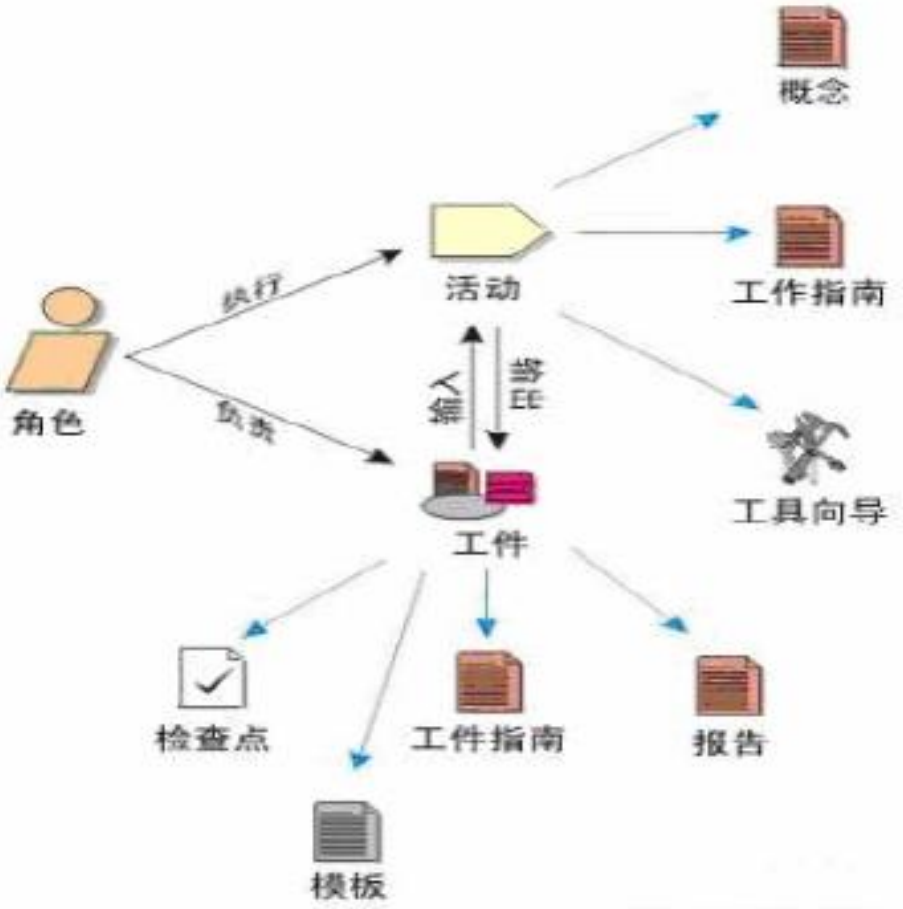
迭代和增量

- 采用迭代和增量的开发方法，将整个项目分为多个迭代过程
- 每次迭代仅考虑一部分需求，进行分析、设计、实现、测试和部署。每次迭代在已完成部分的基础上进行，每次增加新的功能，直至项目完成。

三大过程要素

- 角色
- 活动
- 工件

- 每个角色完成指定的活动
- 每个活动产生合格的工作产品
- 每个工件拥有相关的指南、模板和检查点

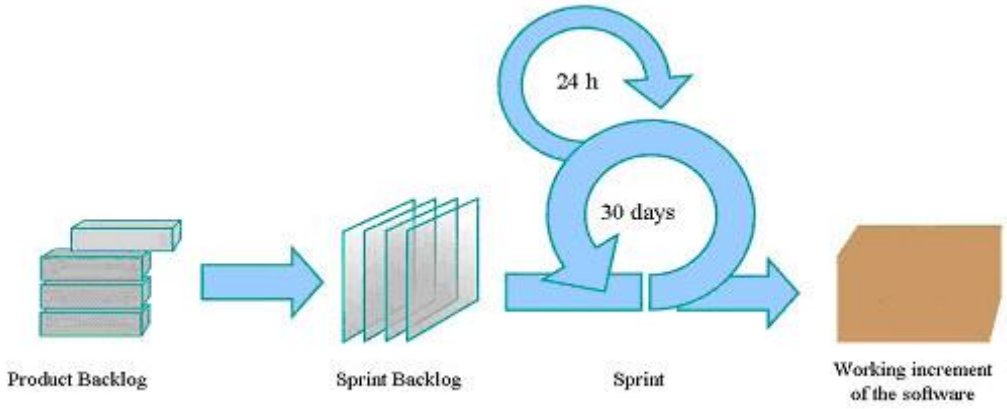


RUP的应用

- 可用于新项目，也可用于正在开发的项目
- 使用方法就是将RUP裁剪为我们所需要的
- 裁剪步骤
  - 确定项目需要哪些工作流
  - 确定每个工作流需要哪些制品
  - 确定四个阶段如何演进。确定阶段间演进要以风险控制为原则，决定每个阶段保留哪些工作流，制品需要哪些等。
  - 确定每个阶段内的迭代计划。规划每次迭代开发的内容
  - 规划工作流内部结构
    - 涉及角色、活动和制品，复杂程度与项目规模有关，用活动图表示内容结构

敏捷模型

- 一种软件开发方法,基于迭代和增量开发,通过自组织,跨团队,沟通协作完成开发工作
- 以用户的需求进化为核心，采用迭代、循序渐进的方法进行软件开发
- 简洁
- 协作



敏捷模型

- 定义了一系列的核心原则和辅助原则
  - 主张简单、拥抱变化、你的第二个目标是可持续性、递增的变化、令Stakeholder投资最大化、有目的的建模、多种模型、高质量的工作、快速反馈、软件是你的主要目标、轻装前进
- 定义了一组核心实践（practice）和补充实践
  - Stakeholder的积极参与、正确使用artifact、集体所有制、测试性思维、并行创建模型、创建简单的内容、简单地建模、公开展示模型、切换到另外的Artifact 、小增量建模、和他人一起建模、用代码验证、使用最简单的工具
- 敏捷开发更是一种思想和理念

思考

- 1、以下对软件过程的理解是否正确？阐明原因。
  - 1）我们已经有了一整套用于记录软件开发以及项目管理的标准文档，我想我们已经拥有了过程。
  - 2）因为我们对软件工程师很信任，并且对他们进行了关于过程的培训，再加上他们在自己的工作中运用了所学到的知识，所以我们没有必要强制他们去做这做那了，当然也不用再对他们的工作进行监督。
  - 3）我们已定义好了软件过程，而且将其进行了文档化，同时对所有成员也进行了相应的培训。现在一切都理顺了，毫无疑问，过程自己会运行起来，我们不用再费劲地去对过程实施的情况进行检查了。

思考

- 4）我们已定义了过程，进行了文档化，做了培训，并且为了保证过程实施的效果还采用了强制的手段。现在整个过程的实施已很稳定，看起来以后我们也不用做什么改动就能保持目前这种高效的运行效果。
- 5）我们有一个高层的管理者，他对过程改进所能带来的效益深信不移。因此我们就不用再关心过程改进的投入产出比了。
- 6）我们已建立起软件过程规范化的环境。过程已被定义，也进行了培训。现在再也没有什么可以影响过程的实施了。

思考

- 2、选择两种你熟悉的软件过程模型，为该模型定义相应的过程规范，阐明每一过程活动涉及的角色，每位角色所承担的工作和职责、应该提交的产品。

谢谢！