第二章
软件工程与需求工程

13:29

课程内容

- 2.1 软件工程
- 2.2 软件过程模型
- 2.3 需求工程与软件开发
- 2.4 需求工程的过程

2.1 软件工程

- 软件危机：人们难以控制软件的开发和维护。
  - 大型软件系统十分复杂，很难理解和维护；
  - 软件开发周期过长，对软件开发成本和进度的估计常常不准确，费用超出预算，实际进度一再拖延；
  - 软件系统的可靠性差；
  - 软件的可维护性程度非常低；
  - 软件通常没有适当的文档资料；
  - 软件开发生产率的提高赶不上硬件的发展和人们需求的增长

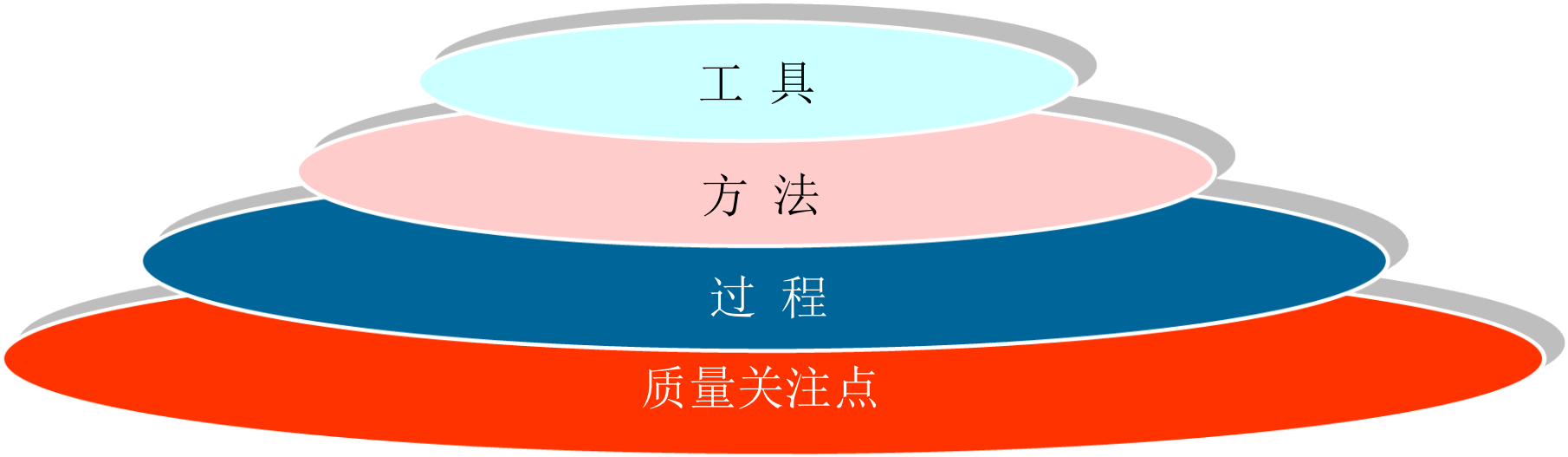
软件工程(续)

- 软件危机的原因
  - 客观：软件本身特点
    - 规模庞大
    - 复杂性
  - 主观：不正确的开发方法
    - 忽视需求分析
    - 错误认为：软件开发=程序编写
    - 轻视软件维护

软件工程 (续)

- 软件工程：用工程方法开发和维护软件的过程和有关技术
- 软件工程研究的基本内容
  - 软件开发过程
  - 软件开发和维护的方法与技术
  - 软件开发和维护工具系统
  - 质量评价和质量保证
  - 软件管理和软件开发环境
  - ……

软件工程 (续)



2.2 软件过程模型

- 软件过程：开发和维护软件及其相关产品所涉及的一系列活动
- 软件过程模型：为获得高质量的软件系统所需完成的一系列任务的框架，规定了完成各项任务的工作步骤

软件过程模型(续)

- 软件过程模型的依据：软件生命周期
- 软件生命周期：软件产品或系统一系列相关活动的全周期
  - 软件计划
  - 需求分析和定义
  - 设计
  - 编码
  - 测试
  - 运行与维护

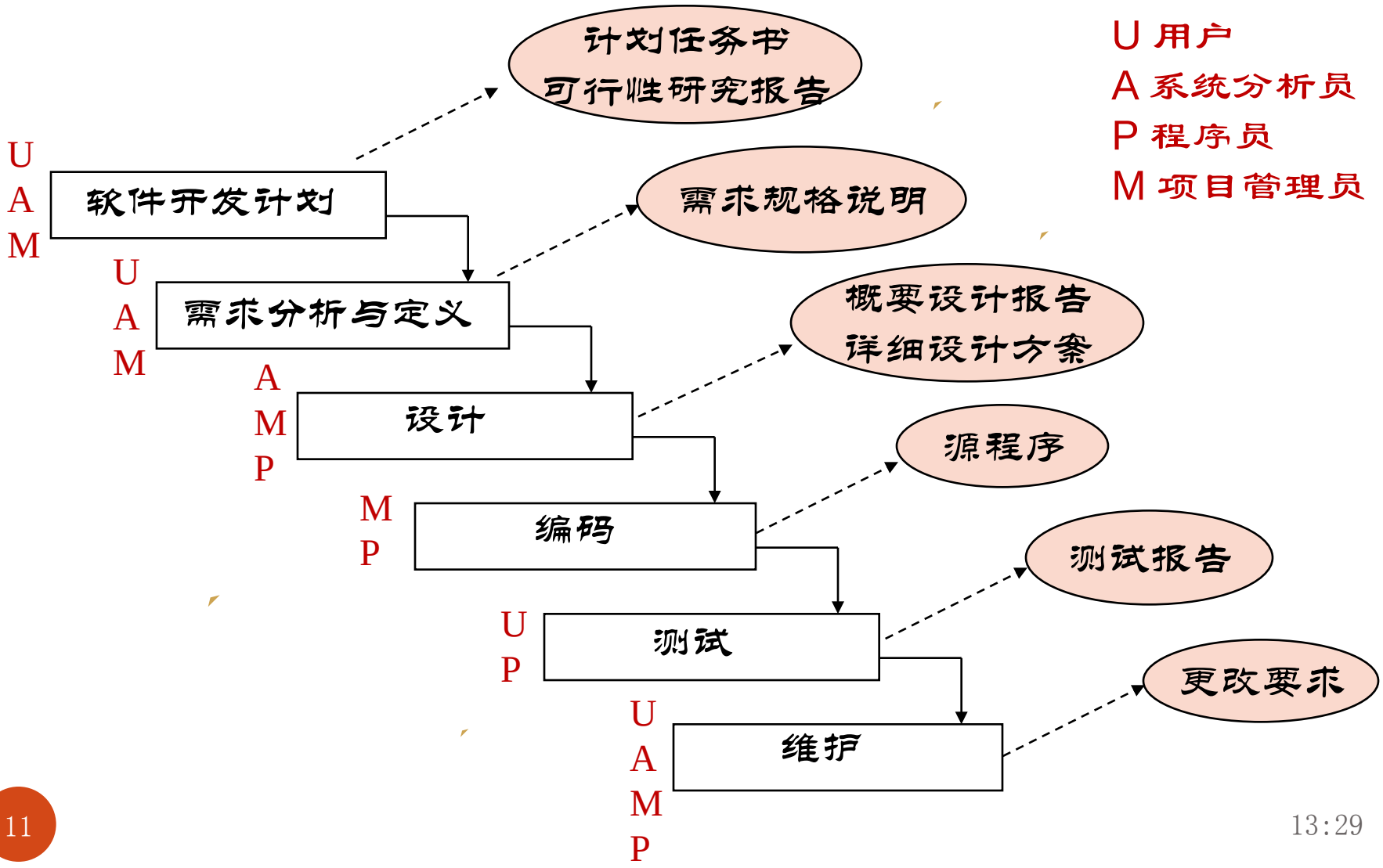
软件过程模型(续)

1. 瀑布式模型
2. 快速原型模型
3. 渐增式模型
4. 螺旋式模型
5. 面向对象的开发模型

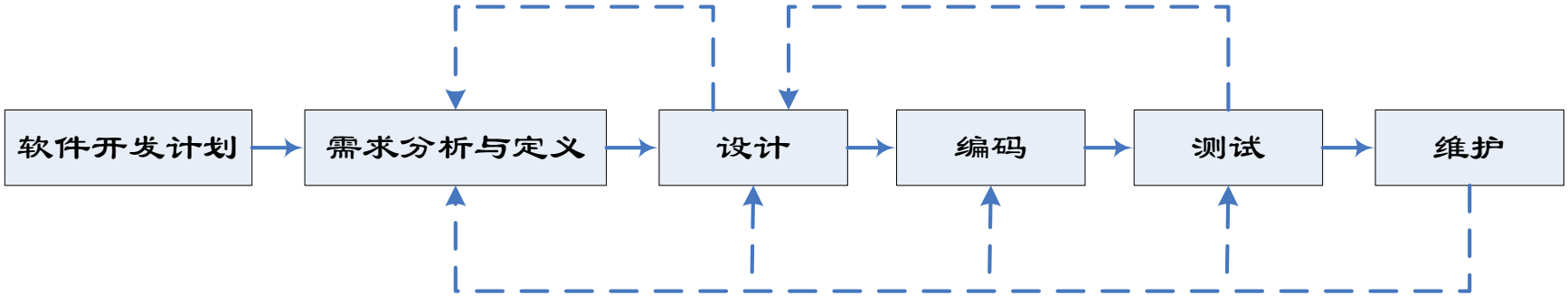
1. 瀑布式模型

- 软件开发过程分为六个阶段，每个阶段都有明确的分工和任务，并产生一定的书面结果
- 各阶段之间是紧密相关的，后一阶段的工作依据前一阶段的工作结果而开展

瀑布式模型(续)


计划任务书
可行性研究报告
U
A
M
软件开发计划
需求规格说明
U
A
M
需求分析与定义
概要设计报告
详细设计方案
A
M
P
设计
源程序
M
P
编码
测试报告
U
P
测试
更改要求
U
A
M
P
维护
U 用户
A 系统分析员
P 程序员
M 项目管理员

瀑布式模型(续)



瀑布式模型(续)

- 特点
  - 阶段间具有顺序性和依赖性
  - 质量保证：各阶段必须完成规定的文档；每个阶段结束前必须完成文档审查
  - 只提供了一个指导性框架，缺乏具体的实施方法和技术
  - 并非以线性方式进行，在实际的软件开发中存在反复

瀑布式模型(续)

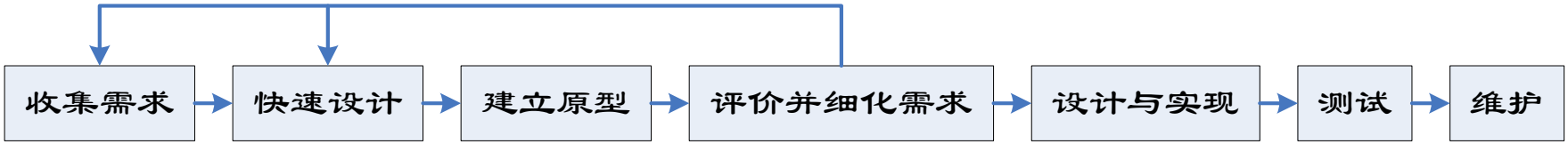
- 不足
  - 要求用户一开始就提出清晰完整的需求；
  - 阶段间移交信息（文档）的过程中，由于个人的理解不同，容易产生误解；
  - 用户的参与程度不够。

2. 快速原型模型

- 原型：一种与原物的结构、大小和一般功能接近的形式或设计
- 软件原型：待开发的软件系统的部分实现
- 快速原型：在完成最终可运行软件系统之前快速建立实验性的、可在计算机上运行的程序（原型）

快速原型模型(续)

- **基本思想**：快速建立一个实现了若干功能(不要求完全)的可运行模型来**启发、揭示和不断完善**用户需求，直到满足用户的全部需求为止



快速原型模型(续)

- 原型系统的内部结构并不重要
- 重要的是，必须迅速地构建原型，然后根据用户意见迅速地修改原型
- 当快速原型的某个部分是利用软件工具由计算机自动生成的时候，也可以将这部分用到最终的软件产品中

快速原型模型(续)

- 目的
  - 明确并完善需求
  - 探索设计选择方案
  - 可以发展为最终的产品

快速原型模型(续)

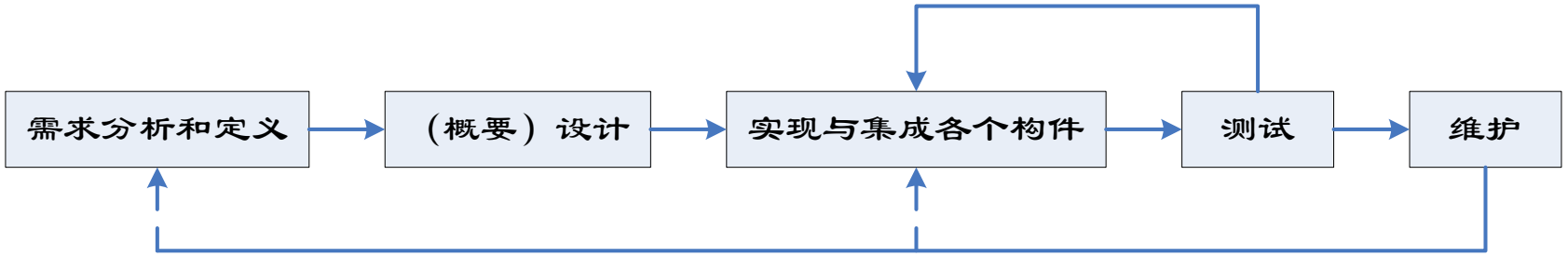
- 特点
  - 弥补了瀑布式模型的一些不足，用户可以充分地参与到软件开发中
  - 使用户的需求明确化，可减少用户需求的遗漏或用户频繁修改需求的可能性
  - 本质是“快速”

快速原型模型(续)

- 不足
  - 用户易视原型为正式产品
  - 快速原型系统对于软件系统的开发环境要求较多，在一定程度上影响了其使用的范围和实用价值

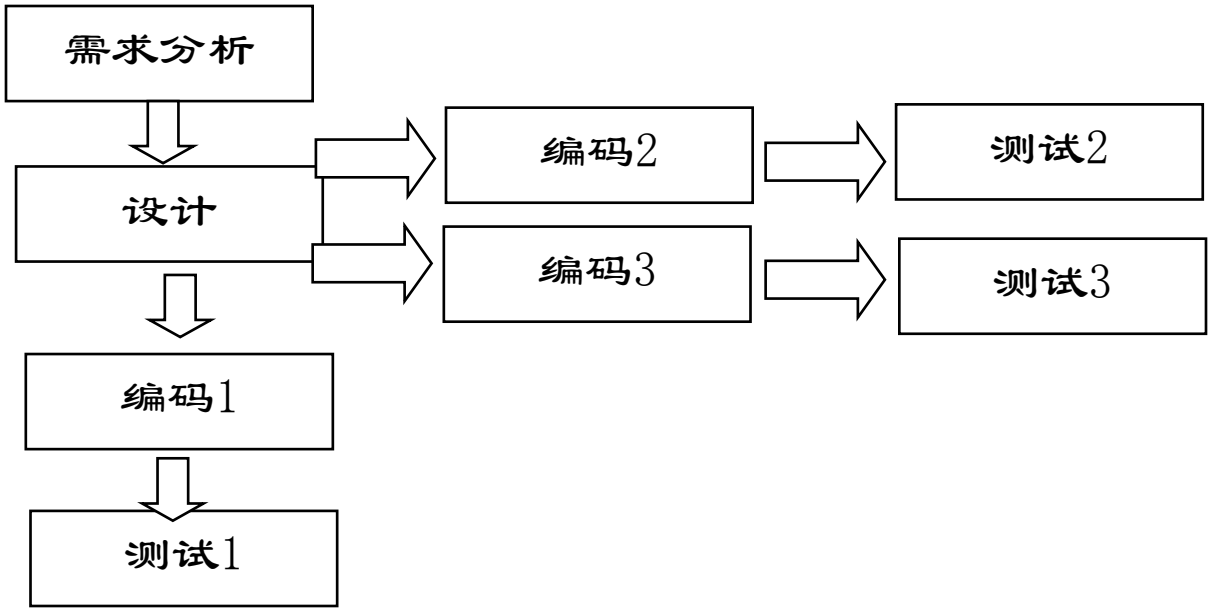
3. 渐增式模型

- **基本思想**：从核心功能开始，通过不断地改进和扩充，使得软件系统能适应用户需求的变动和扩充，从而获得柔性较高的软件系统



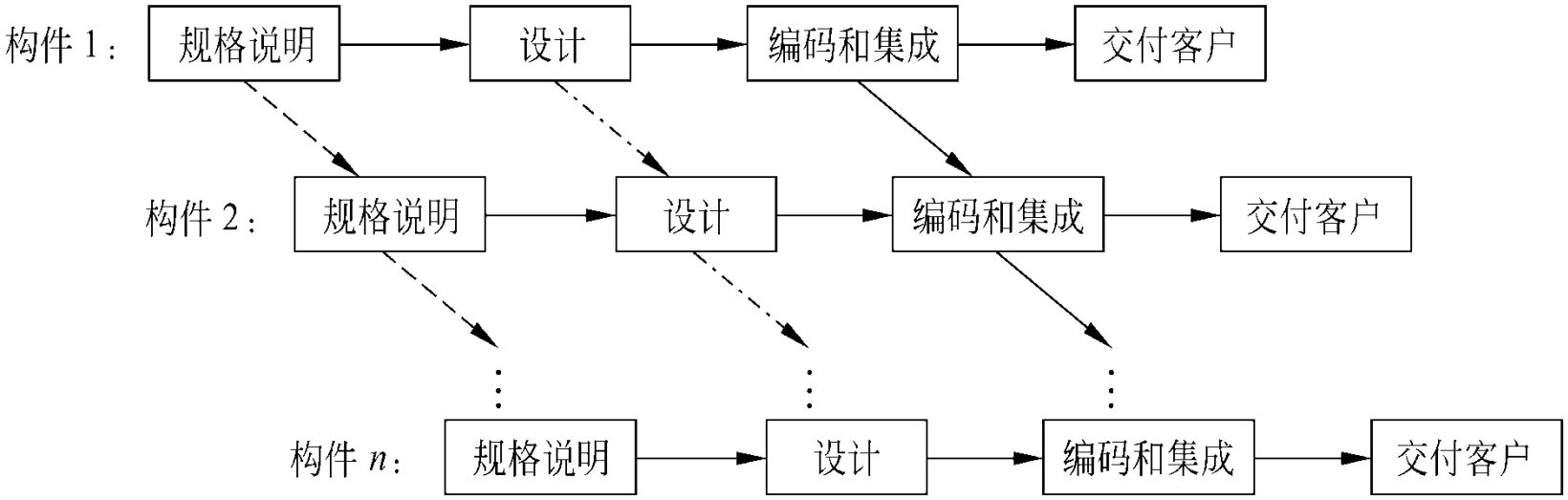
渐增式模型(续)

- 增量构造模型：在瀑布模型基础上，对一些阶段进行整体开发，对另一些阶段进行增量开发



渐增式模型(续)

- 演化提交模型：在瀑布模型的基础上，所有阶段都进行增量开发，即不仅是增量开发，也是增量提交



渐增式模型(续)

- 例：渐增式模型开发文字处理系统
  - ① 实现基本的文件处理、编辑和文档生成功能
  - ② 实现拼写和语法检查功能
  - ③ 完成高级的页面排版功能

渐增式模型(续)

- 与快速原型模型的比较：
  - 相同点：尽早向用户提供可运行的软件系统
  - 区别：优先开发什么功能或部分系统
    - 快速原型模型主要根据用户需求较为模糊的部分优先开发原型
    - 渐增式模型从功能明确、设计技术上不确定因素很少的核心功能优先开发

渐增式模型(续)

- 特点：
  - 能在短时间向用户提交可完成部分功能的产品
  - 能逐步增强产品功能，以使用户有较充裕的时间学习和适应新的软件系统

渐增式模型(续)

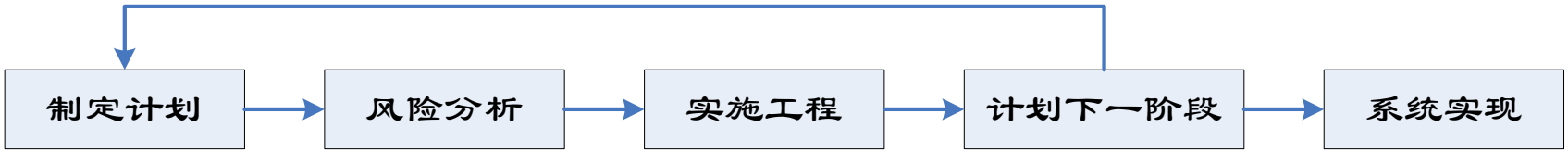
- 不足：
  - 在把每个新增的构件或功能集成到现有的软件系统中时，必须不破坏该软件系统
  - 在设计软件系统的体系结构时，要充分考虑到其开放性，而且加入新构件的过程必须简单和方便。

4. 螺旋式模型

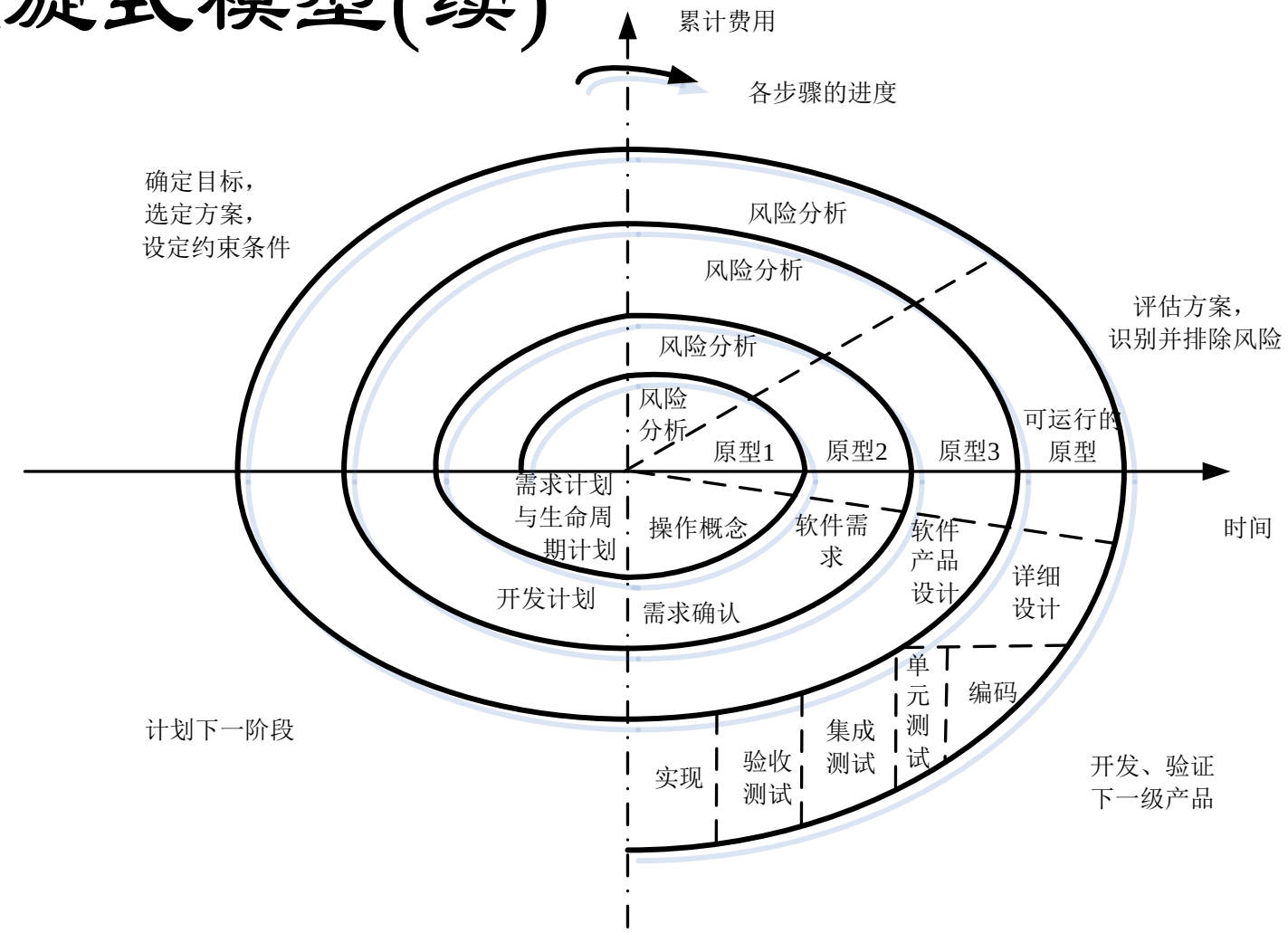
- 项目的规模越大，问题越复杂，资源、进度、成本等因素的不确定性越大，承担项目所冒的风险越大
- 在软件的开发过程中应该考虑风险问题，及时识别和分析风险，且采取适当的措施以消除或减少风险的危害

螺旋式模型(续)

- 基本思想：把瀑布式模型与快速原型模型结合到一起，加上风险分析（风险识别、风险分析、风险控制）



螺旋式模型(续)


累计费用
各步骤的进度
确定目标，
选定方案，
设定约束条件
风险分析
风险分析
风险分析
风险
分析
评估方案，
识别并排除风险
可运行的
原型
原型1
原型2
原型3
需求计划
与生命周
期计划
操作概念
软件需
求
软件
产品
设计
详细
设计
时间
开发计划
需求确认
单
元
测
试
编码
集成
测试
验收
测试
实现
计划下一阶段
开发、验证
下一级产品

螺旋式模型(续)

- 特点
  - 适用于软件开发机构内部开发大规模软件项目
  - 对于可选方案和约束条件的强调有利于已有软件的重用，也有助于把软件质量作为软件开发的一个重要目标
  - 减少过多测试或测试不足所带来的风险

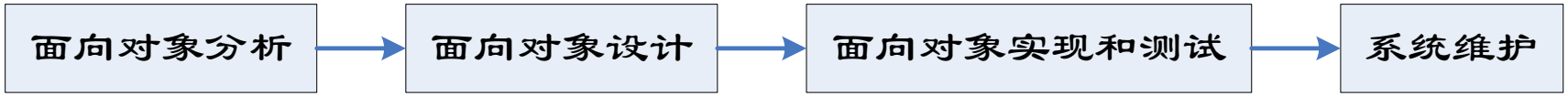
螺旋式模型(续)

- 不足
  - 要求软件开发人员具有丰富的风险评估经验和专门知识
  - 若执行风险分析大大影响项目的利润，那进行风险分析毫无意义

5. 面向对象的开发模型

- 面向对象：应用面向对象的概念对问题进行分析和求解的软件开发技术，或者说，是以对象（类）为数据中心、对象之间的动态行为模式作为运行机制的一种问题求解方法。

面向对象的开发模型(续)



- 面向对象分析：构造可理解的现实世界的问题分析模型
- 面向对象设计：确定对象的内部细节
- 面向对象实现和测试：用面向对象的程序设计语言和工具进行实现，包括面向对象程序设计和组装测试

面向对象的开发模型(续)

- 特点
  - 各个阶段之间呈现一种过程交替、工作循环和信息反馈的复杂关系，呈现出非线性的工作方式
  - 软件系统的表达形式在整个开发模型中都是相同的，即面向对象方法中把类及其结构作为系统的表达单元，无论哪一个阶段都以渐增的方式不断地进化或细化这些表达单元
  - 开发模型支持软件的重用。

2.3 需求工程与软件开发

1. 需求工程对软件开发的影响
2. 需求工程面临的困难

1. 需求工程对软件开发的影响

- 需求是制定项目计划的基础。
- 需求工程所产生的最终产物（需求规格说明）是软件设计和软件实现的基础。
- 需求规格说明也是测试工作和用户验收软件系统的依据。
- 需求规格说明也是软件维护工作的依据。

需求工程贯穿于软件系统的整个开发工作中

2. 需求工程面临的困难

- 需求获取与需求分析的困难性
- 需求描述语言和规范化的困难性
- 需求验证的困难性
- 需求管理的困难性

需求工程面临的困难(续)

- 需求获取与需求分析的困难性
- 需求描[illegible]规范化的困难性
- [illegible]
- [illegible]

  - 有些需求可能用户也不是很清楚
  - 需要用户与开发人间进行充分的交流和协商
  - 需求间的冲突和矛盾的检查以及解决
  - 需求是否完整和确定
  - 合适的需求建模的方法和技术

需求工程面临的困难(续)

- 需求获取与需求分析的困难性
- 需求描述语言和规范化的困难性
-
-
  - 怎样规范化用户需求
  - 规范化哪些用户需求
  - 非形式化和形式化描述语言的使用

需求工程面临的困难(续)

- 需求获取与需求分析的困难性
- 需求描述语言和规范化的困难性
- 需求验证的困难性
-

  - 需求规格说明正确性的确认和验证
  - 验证的方法和技术
  - 如何进行自动验证

需求工程面临的困难(续)

- 需求获取与需求分析的困难性
- 需求描述语言和规范化的困难性
- 需求验证的困难性
- 需求管理的困难性

  - 需求规格说明书的质量保证
  - 需求规格说明书的版本管理
  - 需求变更的控制。

2.4 需求工程过程

- 需求工程的目标：给出待开发或待完善的软件系统的一个清晰的、完整的、无二义性的和精确的描述，并最终产生高质量的软件需求规格说明
- 需求工程过程是由导出、确认和维护软件系统需求规格说明的一系列活动组成的。

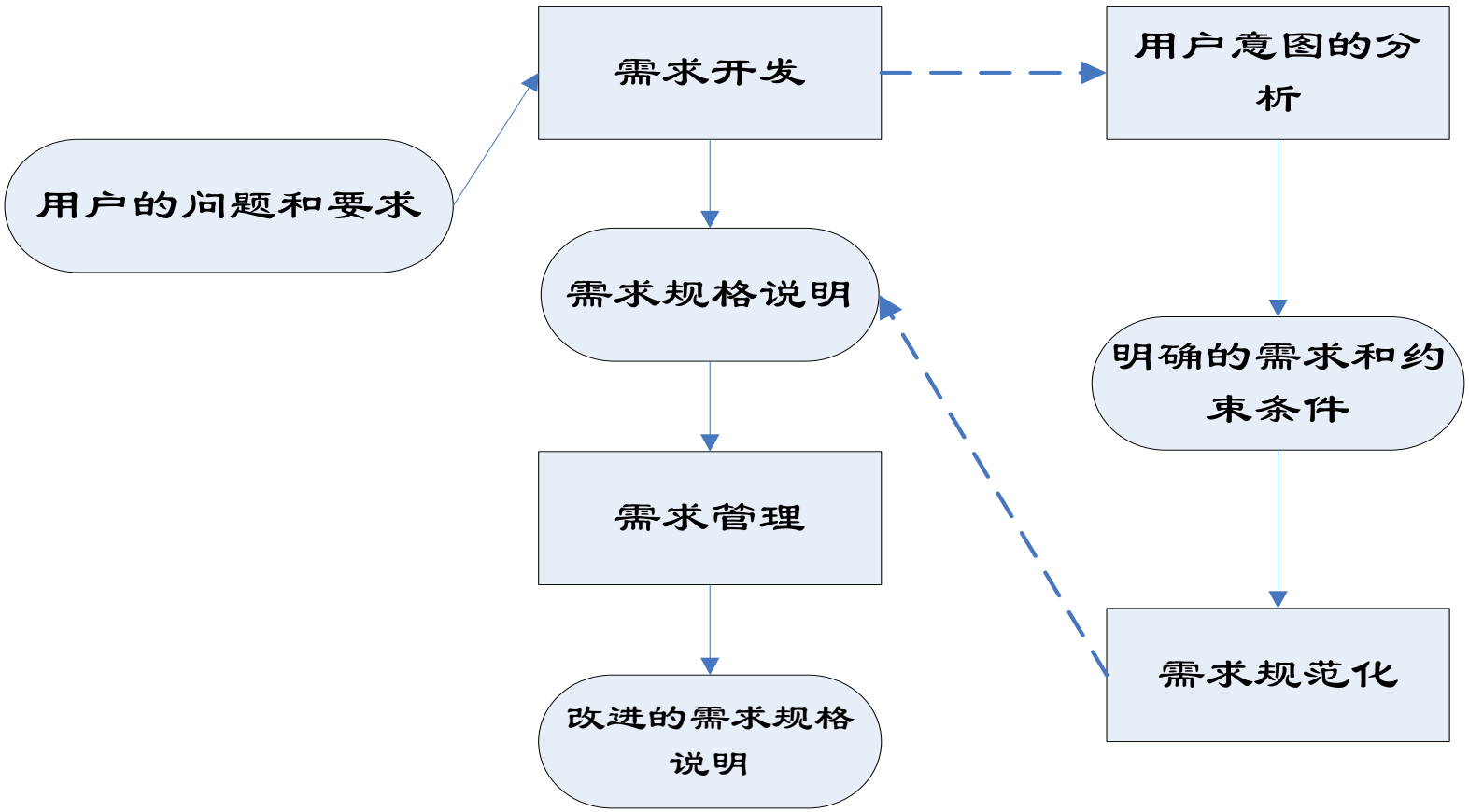
需求工程过程(续)

- 两个阶段：
  - 需求开发：收集、分析、编写、验证需求的全过程

    重点在于开发出高质量的需求规格说明书
  - 需求管理：对需求的变化全过程进行跟踪

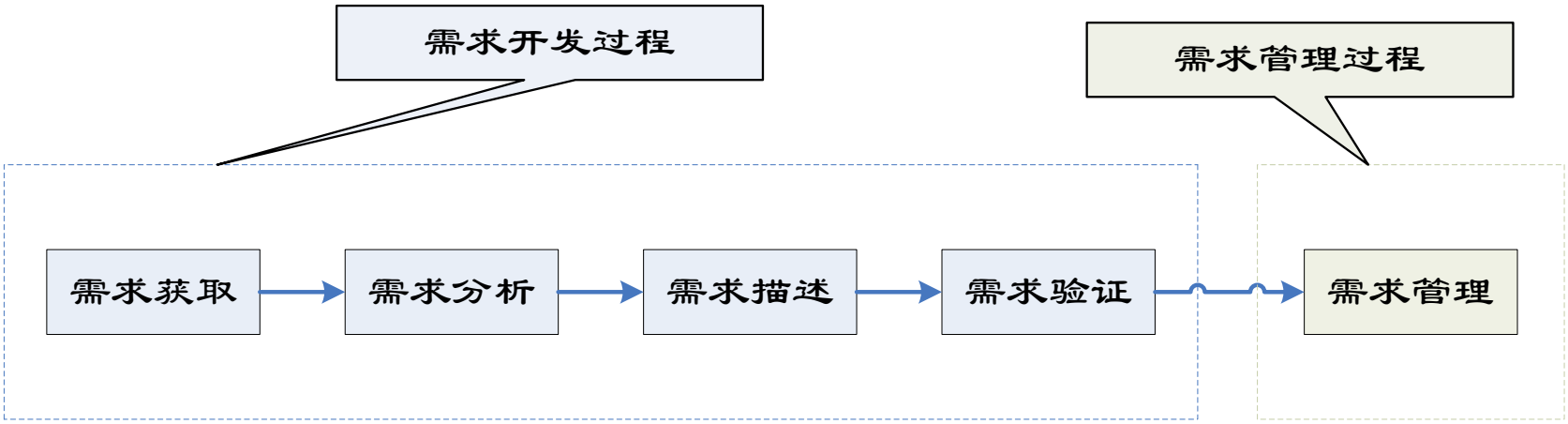
    重点在于确保开发的软件满足需求的定义

需求工程过程(续)


用户的问题和要求
需求开发
用户意图的分析
需求规格说明
明确的需求和约束条件
需求管理
需求规范化
改进的需求规格说明

需求工程过程(续)

- 需求工程过程可进一步划分为下图的若干阶段



需求工程过程(续)

需求获取：确定和收集与软件系统相关的、来自不同来源和对象的用户需求信息

- 需求获取在实际应用中常存在的问题：

  (1) 捕获范围不足：许多需求分析人员认为需求就是用户要实现什么功能，以为业务知识不是系统要做的，因此在需求获取时往往不注重对业务知识的捕获。表面上节省了一些时间，代价是后续工作往往要不断变更

当前，许多大型的应用软件系统开发都需要行业专家的参与。行业专家的主要特点是了解相关行业的业务知识，对业务过程、业务流程等相当熟悉。在系统开发初期，他们比技术人员起到的作用更大。

需求工程过程(续)

需求获取

(2) 缺乏计划性、科学性：捕获过程随意，走过场，预先没有对问题、时间、访谈人员进行计划，造成需求获取的时间利用率不高；获取过程相对比较分散，没有做到定向、聚焦，且经常将宏观问题和微观问题混在一起。例如，了解流程时可能更关注数据细节问题

当前，许多大型的应用软件系统开发都需要根据行业专家的意见，科学合理地安排获取流程和获取程序。

需求工程过程(续)

需求获取

(3) 捕获对象不明确，获取手段不足：很少主动寻找被访谈者，经常是用户方占据主动。不同场景下，不同的用户层次需要不同的需求获取手段。

主动开展需求捕获工作是需求获取阶段的重要策略。

需求工程过程(续)

需求分析：对获得的用户需求信息进行分析和综合，并找出其中的错误、遗漏或其它不足的地方，以获得用户对软件系统的真正需求，建立软件系统的逻辑模型（或需求模型）

需求分析就是通过对问题域的研究，获得对该领域特性及其存在于其中的（需要解决）的问题的透彻说明

需求工程过程(续)

需求分析

- 需求分析是业务分析，其任务是对问题进行研究，因此将从业务线索入手，而非系统结构入手
- 需求分析是分解过程，将待开发的系统按照职责划分为不同的主题域（子系统），然后分解组成该主题域的所有业务流程，再分解到业务活动、业务步骤
- 需求分析以一种综合的过程，需要将用户的原始需求合并到业务活动中
- 需求分析是一个规范化活动，要找到冲突、矛盾，并通过访谈等手段解决这些问题

需求工程过程(续)

需求定义：使用适当的描述语言，按标准的格式描述软件系统的需求，并产生需求规格说明及其相应文档

将需求分析结果文档化的过程

需求验证：审查和验证需求规格说明是否正确和完整地表达了用户对软件系统的需求

没有经过需求验证的需求往往存在大量的业务和技术隐患，结果将会导致大量的需求变更

需求工程过程(续)

- 需求管理主要体现的思想：
  - 软件需求规格说明书是用来完成信息传递和沟通的，因此必须实现共享
  - 软件需求规格说明书在整个开发过程中是不断演化的，需要建立良好的更新机制

需求工程过程(续)

- 需求管理的主要任务
  - 有效地管理软件系统的需求规格说明及其相应文档
  - 管理需求规格说明的版本
  - 评估需求变更带来的潜在影响及可能的成本费用
  - 跟踪软件需求的状态
  - ……