



西安电子科技大学
XIDIAN UNIVERSITY

软件过程 and 开发方法



1. 何为软件过程模型

✓ 基本概念和特点

2. 有哪些软件过程模型

✓ 有什么类别，各有什么优缺点

3. 如何来选择软件过程模型

✓ 软件过程模型的选择方式和策略



1.1 软件开发的特点

□基于智力的协作过程

- ✓智力活动：基于逻辑思维来构造软件
- ✓交流协作：软件工程师、用户间的交流和讨论

□软件项目内在复杂性

- ✓介入的人多、考虑的内容多、产生的制品多
- ✓不同要素间存在关联

□循序渐进的开发过程

- ✓开展有序的开发活动，如编码、分析、设计
- ✓体现了工程的思想：按步骤、分阶段

按照什么样的过程来有序地开发软件？



1.2 软件过程

□ **过程(Process)**: 软件过程的三要素之一, 从管理的视角, 回答软件开发、运行和维护需要做哪些工作、如何管理好这些工作等问题

- ✓ **活动**: 明确要做哪些事情, 包括具体的活动
- ✓ **关系**: 活动间存在逻辑关系, 如依赖和先后次序
- ✓ **示例**: 装修房子的过程

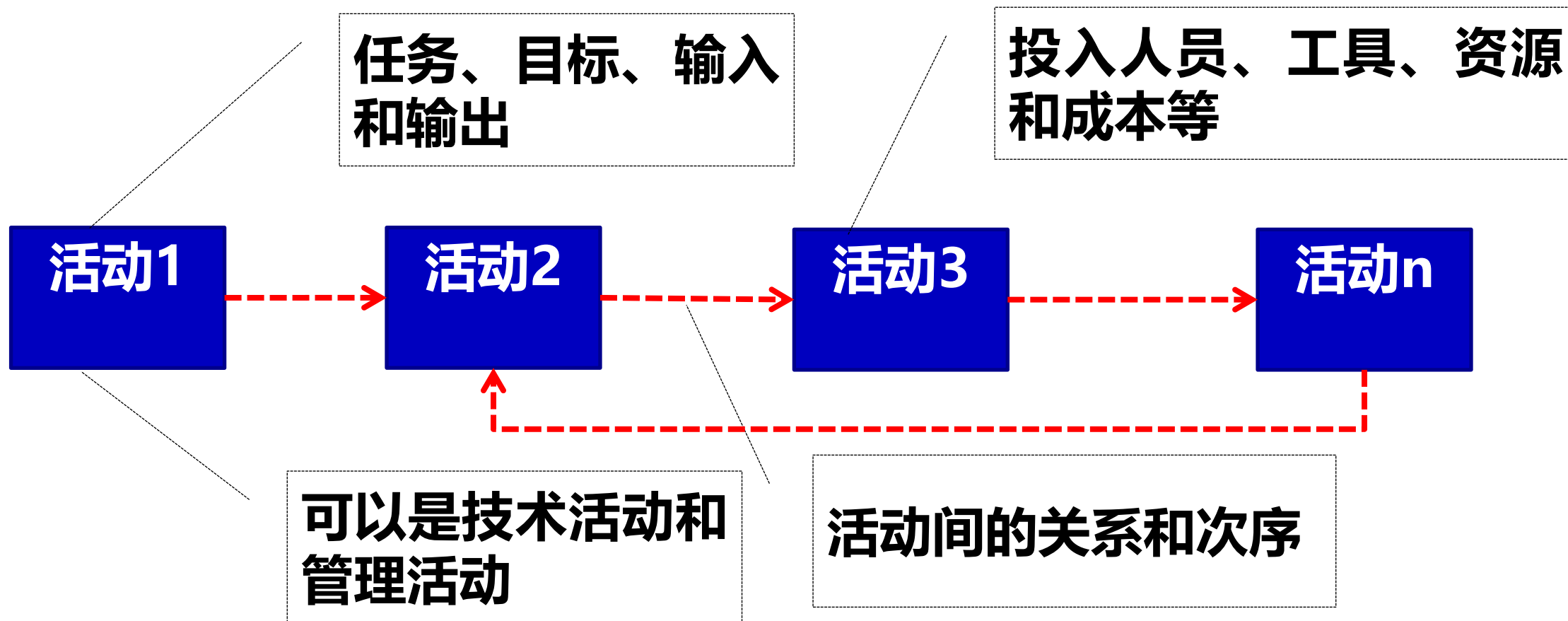
□ **软件过程(Software Process)**

- ✓ 按照项目进度、成本和质量要求, 遵循用户需求, 开发和维护软件、管理软件项目的一系列**有序软件开发活动**
- ✓ 软件开发活动: **技术活动和管理活动**

1.3 软件过程模型

□ 软件过程模型(Software Process Model)

- ✓ 定义了软件开发的具体活动以及活动间的逻辑关系



1. 何为软件过程模型

✓ 基本概念和特点

2. 有哪些软件过程模型

✓ 有什么类别，各有什么特点和优缺点

3. 如何来选择软件过程模型

✓ 软件过程模型的选择方式和策略



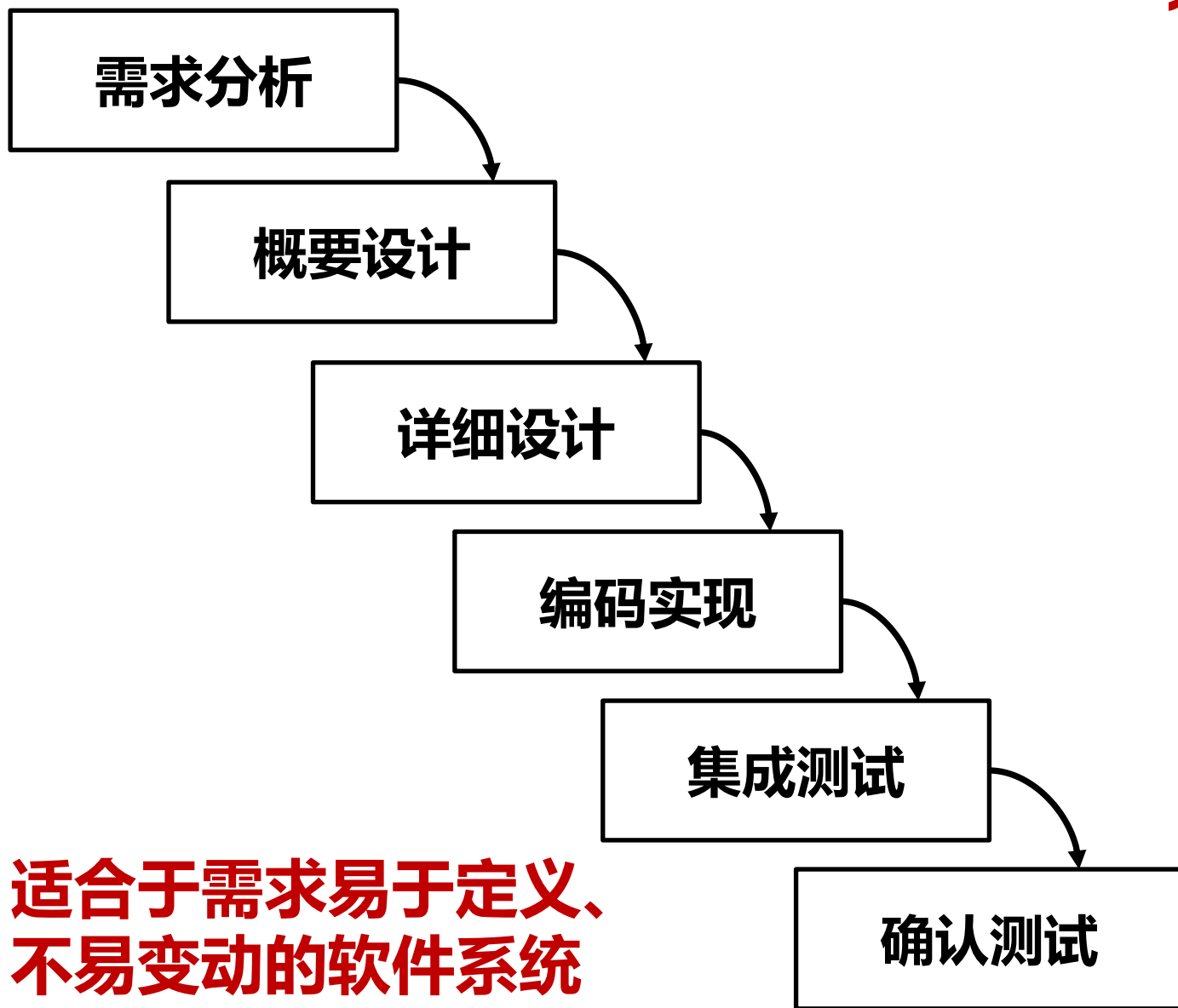
典型的软件过程模型

- 瀑布模型
- 增量模型
- 迭代模型
- 原型模型
- 螺旋模型
- 基于构件的过程模型
- UP模型

- 需要系统、规范性的软件过程模型的指导
- 每种软件过程模型有其各自的特点和适用的场所

2.2 瀑布模型(Waterfall Model)

1970提出的第一个软件过程模型



Ø特点

- 与软件生命周期相互一致
- 步骤严格按照先后次序和逻辑关系来组织实施
- 上一步骤的输出是下一步骤的输入，下一步在需等到前一步骤完成后才能实施
- 每个活动结束后需要评审

Ø优点

- 简单，一目了然，易理解、掌握、应用和管理

需求分析(Requirement Analysis)

□活动

- ✓任务：**定义软件需求**，包括功能、非功能需求
- ✓关注点：**要做什么？(What, Problem)**
- ✓层次和视角：用户角度，仅描述问题和需求

问题是什么？

□方法

- ✓依据：用户的期望和要求
- ✓不断与用户进行交流和商讨，抽象、问题分解、多视点等技术

□产出

- ✓**软件需求模型、软件需求文档、软件确认测试计划**
- ✓**文档类的软件制品**

概要设计(Architecture Design)

□活动

- ✓任务：**建立软件总体架构、制定集成测试计划**
- ✓关注点：**软件高层设计 (How, Solution)**
- ✓层次和视角：宏观、全局、整体、战略性

问题如何解决？

□方法

- ✓依据：软件需求文档
- ✓自顶向下, 逐步求精, 抽象, 模块化, 局部化, 信息隐藏

□产出

- ✓软件概要设计模型、软件概要设计文档、软件集成测试计划
- ✓文档类的软件制品

详细设计(Detailed Design)

□活动

- ✓任务：设计模块内部细节(算法、数据结构)，制订单元测试计划
- ✓关注点：详细设计？(How, Solution)
- ✓层次和视角：微观、局部、细节性

问题如何解决？

□方法

- ✓依据：概要设计文档、软件需求文档
- ✓高质量的软件设计原则

□产出

- ✓软件详细设计模型、软件详细设计文档、单元测试计划
- ✓文档类的软件制品

编程实现(Implementation)

实际解决问题

□活动

- ✓任务：编写程序代码并进行单元测试和调试
- ✓关注点：如何最终做出这个东西？(How, Code)
- ✓层次和视角：最终的实现代码

□方法

- ✓依据：软件概要和详细设计文档、单元测试计划
- ✓采用某种程序设计语言(如C、C++、Java)

□产出

- ✓经过单元测试的源程序代码
- ✓程序类的软件制品

集成测试(Integration Test)

□活动

- ✓任务：**组装软件模块并进行测试以发现问题**
- ✓关注点：**集成后软件中的缺陷 (Bug)**
- ✓层次和视角：自底向上组装、全局

问题解决如何？
软件有缺陷吗？

□方法

- ✓依据：软件概要设计文档、软件集成测试计划
- ✓软件集成测试工具

□产出

- ✓经过集成测试、修复缺陷的源程序代码，集成测试报告
- ✓数据、文档和代码类的软件制品

确认测试(Validation Test)

□ 活动

- ✓任务：测试软件是否满足用户需求
- ✓关注点：软件在满足用户需求方面是否存在缺陷
- ✓层次和视角：从用户角度，聚焦需求是否得以正确实现

问题解决如何？
软件有缺陷吗？

□ 方法

- ✓依据：软件确认测试计划、软件需求文档
- ✓软件测试支撑工具

□ 产出

- ✓经过确认测试、修复缺陷后的代码，软件确认测试报告
- ✓数据、文档和代码类的软件制品

瀑布模型的局限性?

需求分析

概要设计

详细设计

编码实现

集成测试

确认测试

软件需求具有易变、多变的特点

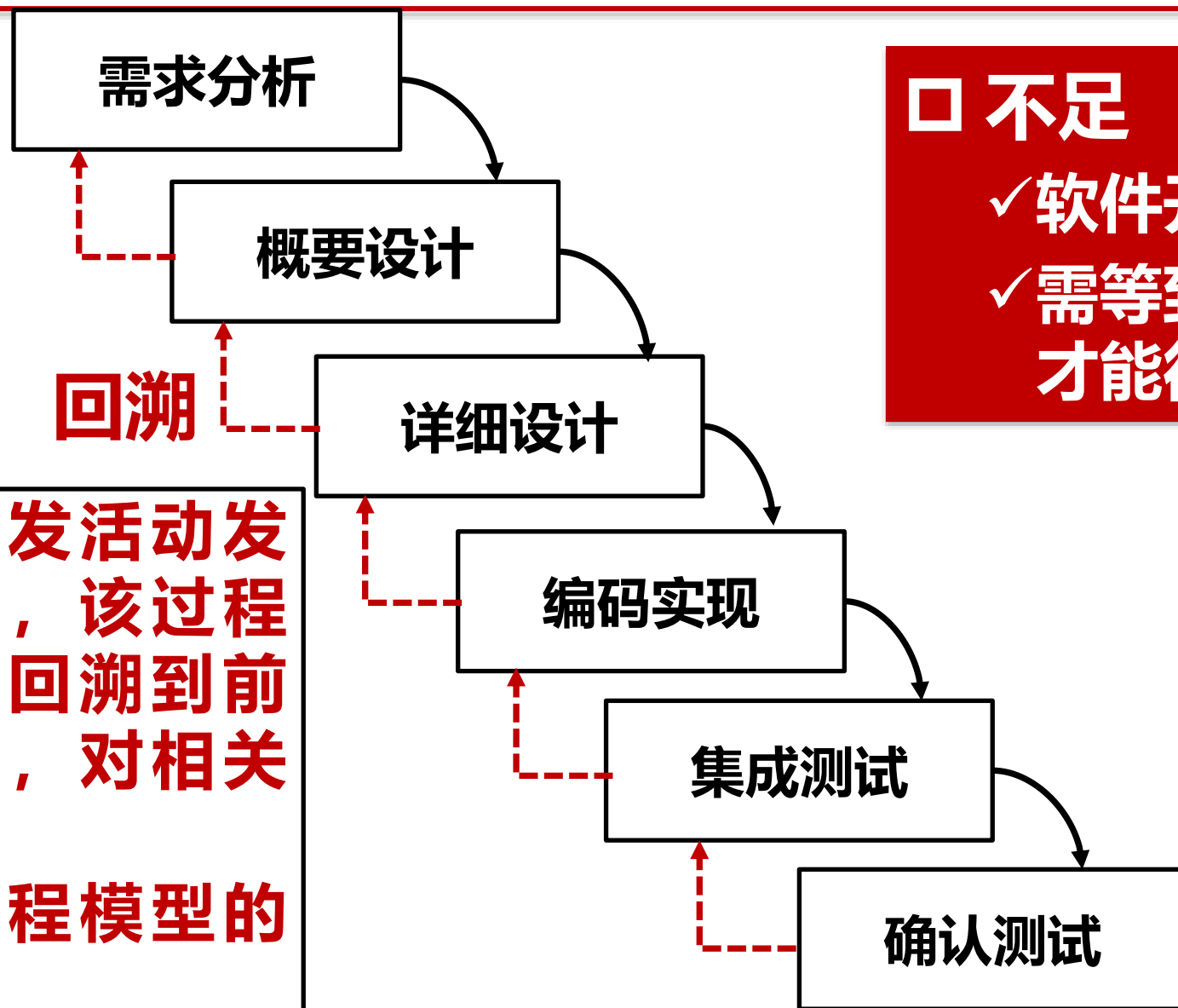
结合课程综合实践思考

□ 不足

- ✓ 需求确定，过于理想化
- ✓ 缺乏变通，难应对变化



2.3 改进的瀑布模型：带反馈和回溯



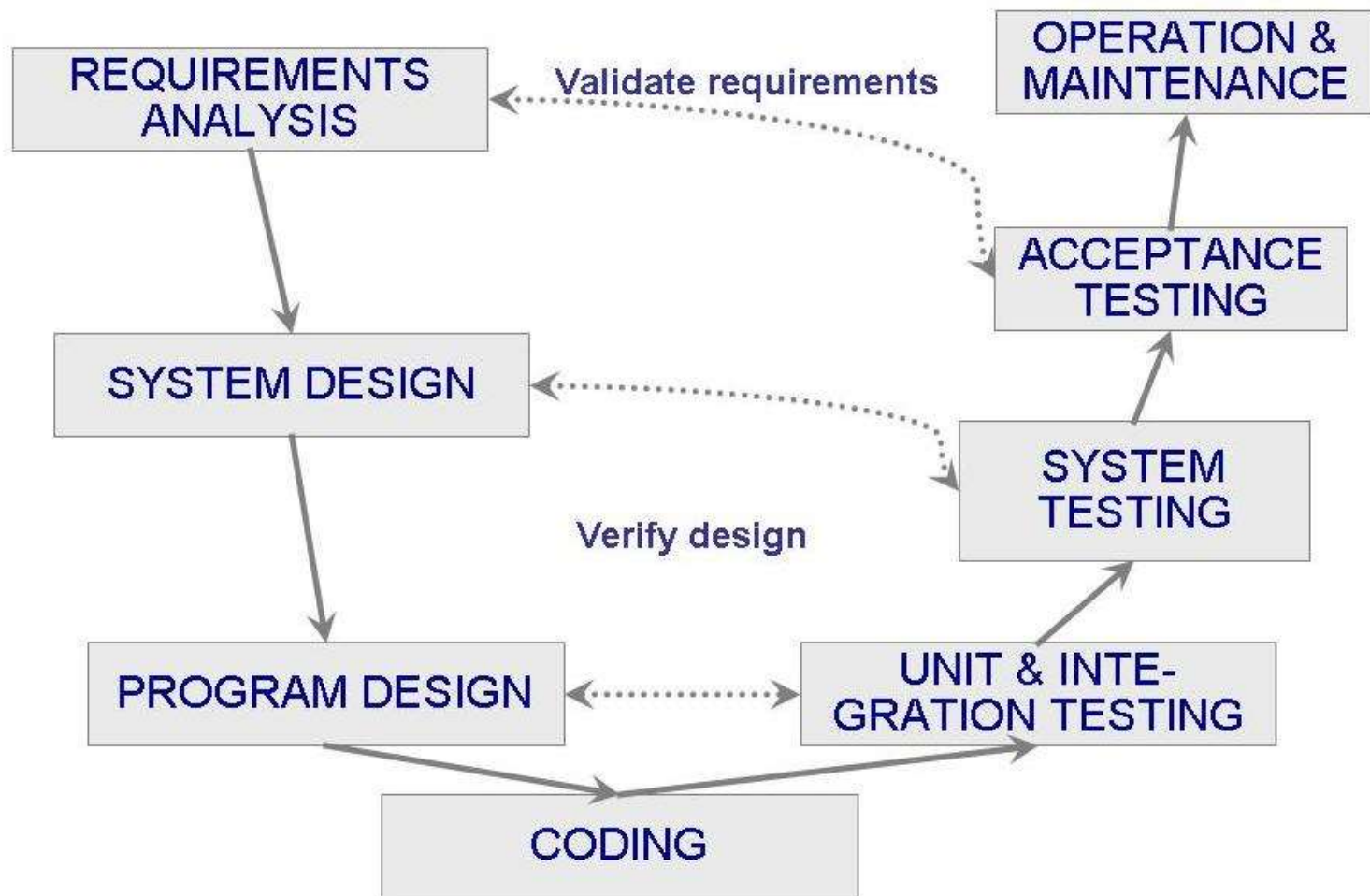
□ 不足

- ✓ 软件开发处于动荡之中
- ✓ 需等到所有功能实现后，才能得到可运行软件



- ✓ 当某个开发活动发现问题时，该过程模型允许回溯到前一项活动，对相关问题解决
- ✓ 提高了过程模型的灵活性

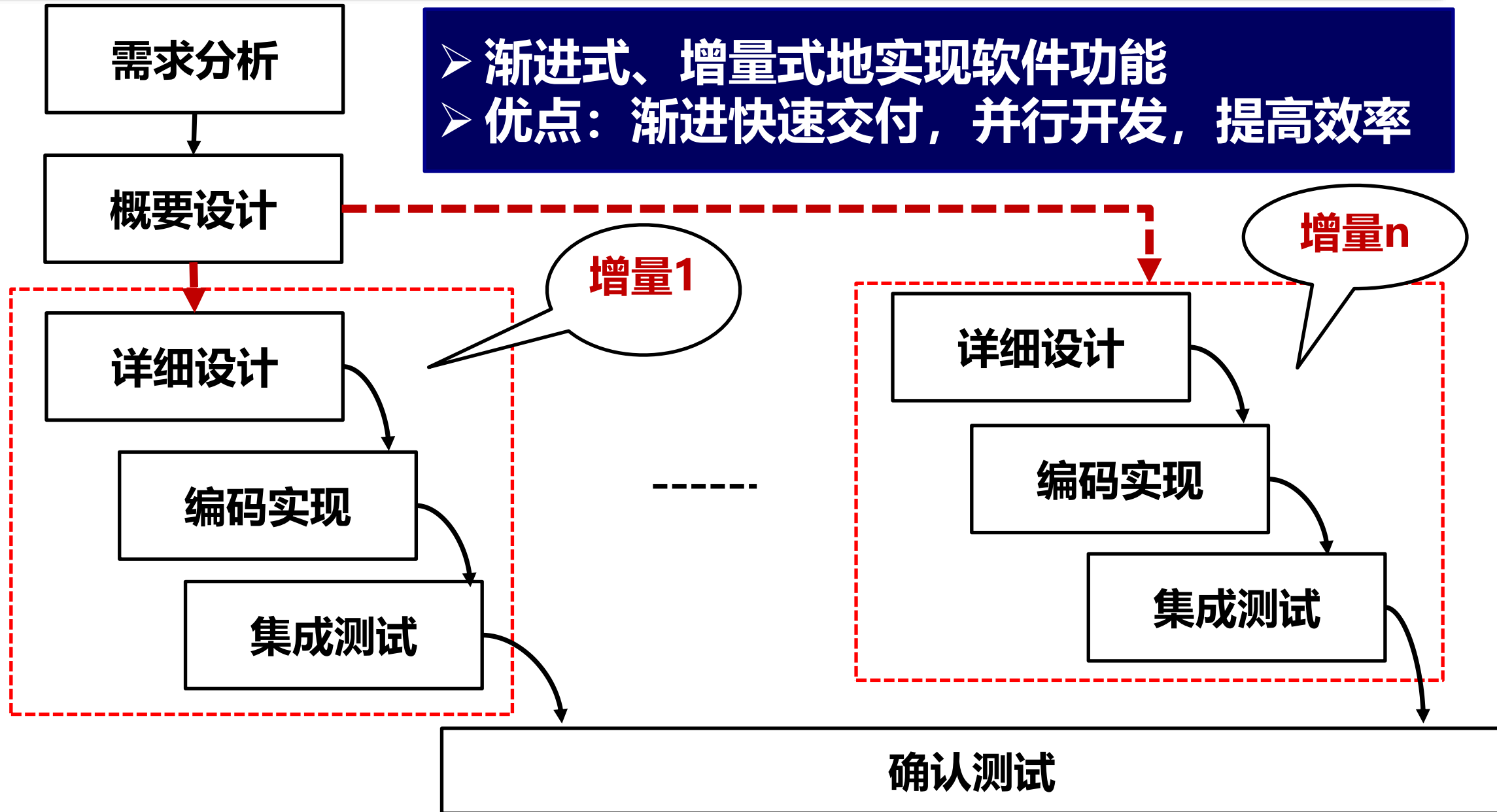
✓ 建立软件开发与软件测试活动之间的对应关系



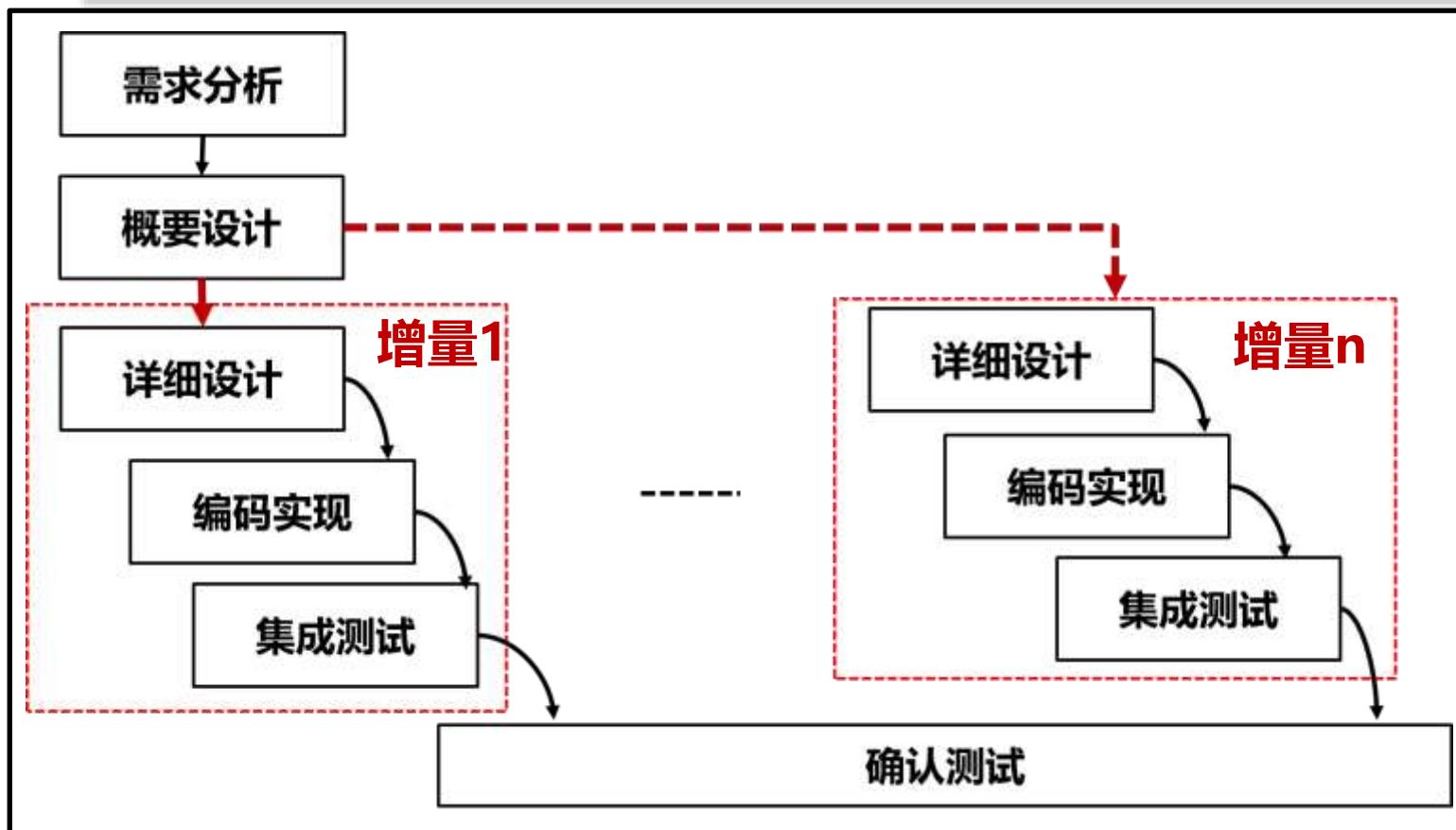
瀑布模型及其变体的局限性

- 通过需求分析给出软件系统的完整需求不现实，过于理想化
- 软件开发人员要等到后期阶段才能得到可运行的软件系统，用户才可以接触和使用软件

2.4 增量模型(Incremental Model)



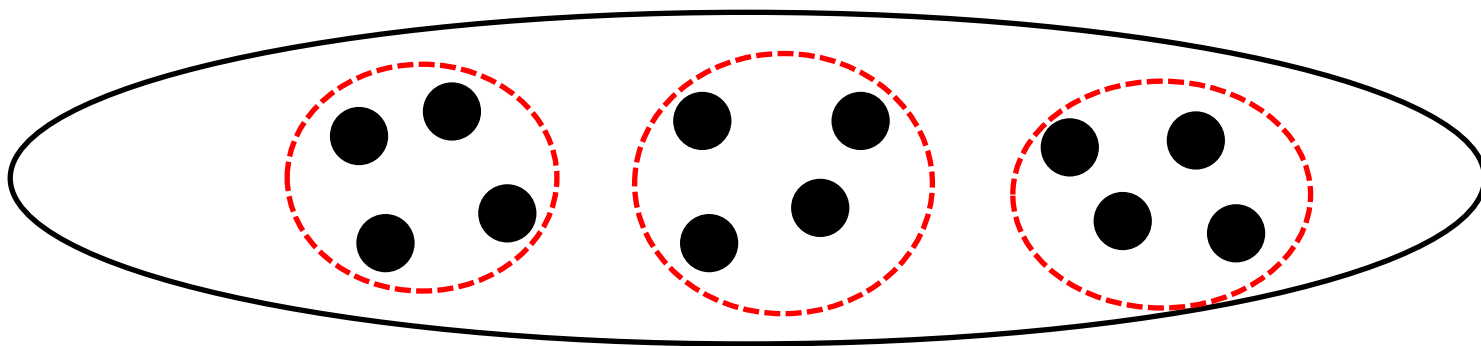
增量模型的局限性?



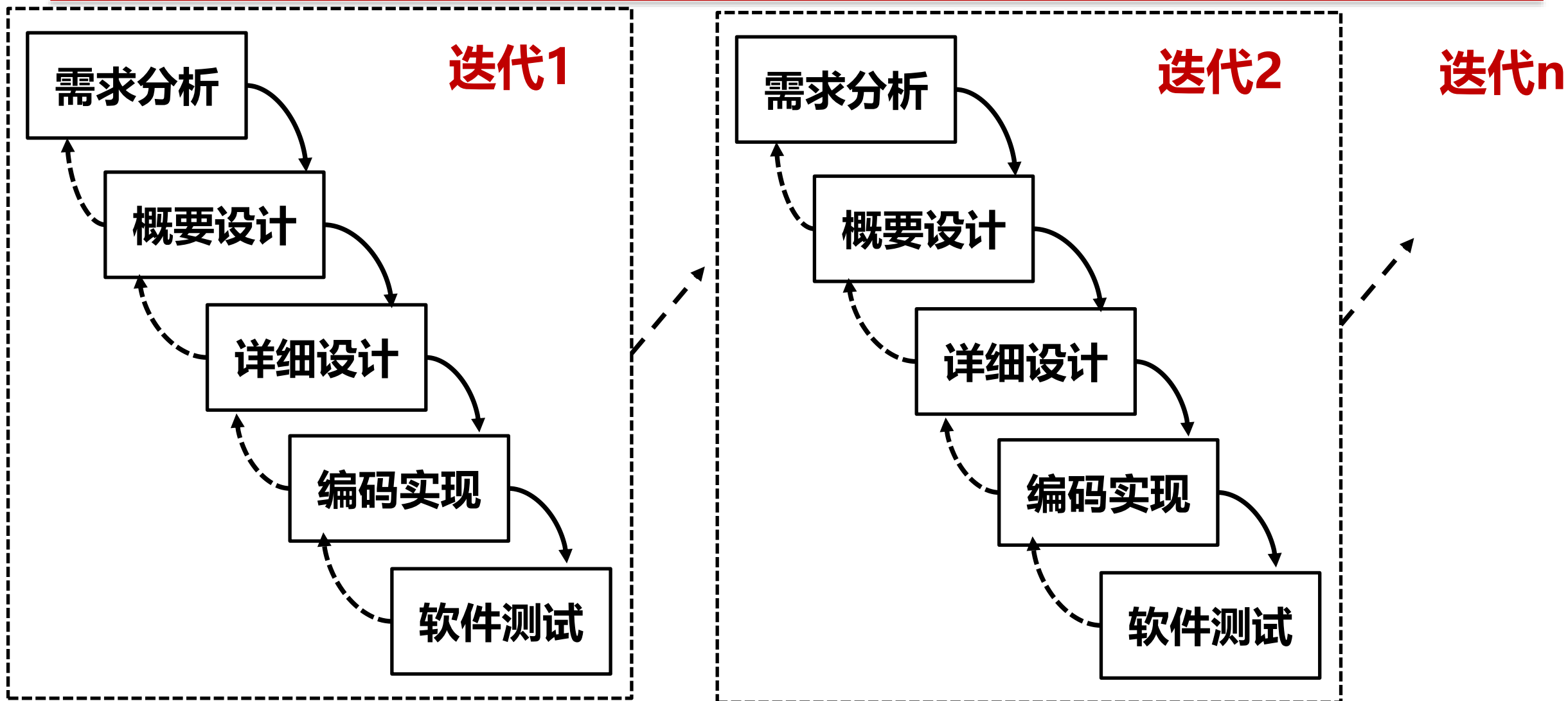
复杂软件系统的需求是不断演化的

□ 不足

✓ 软件需求可确定且不易于变化

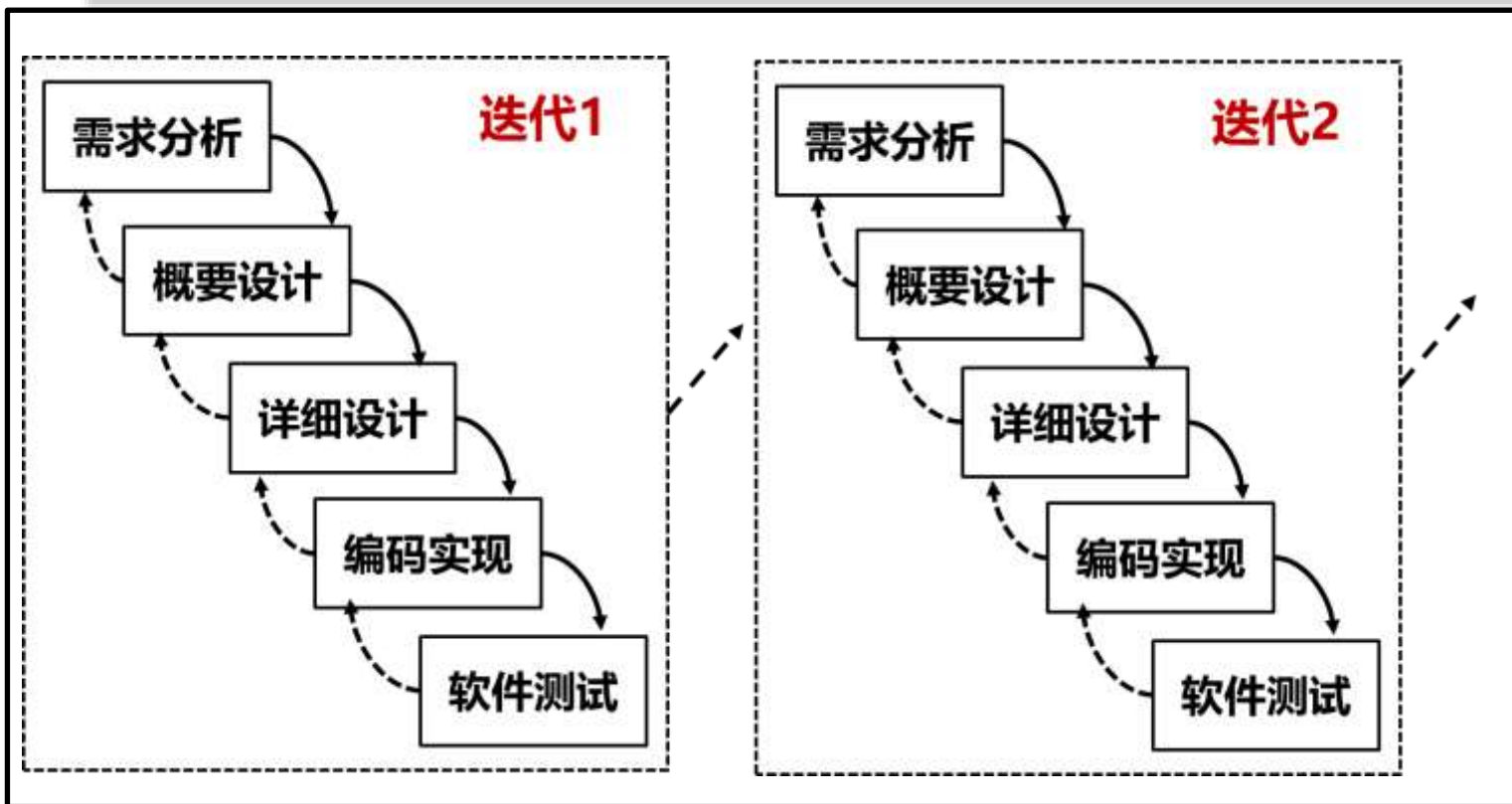


2.5 迭代模型(Iterative Model)

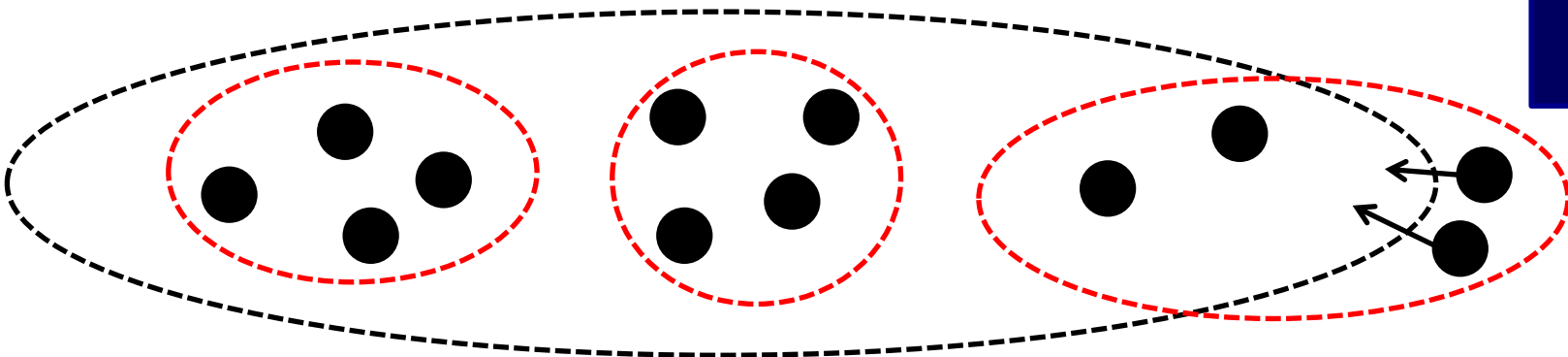


每次迭代只针对部分需求，都包括了软件开发的完整过程

迭代模型的特点

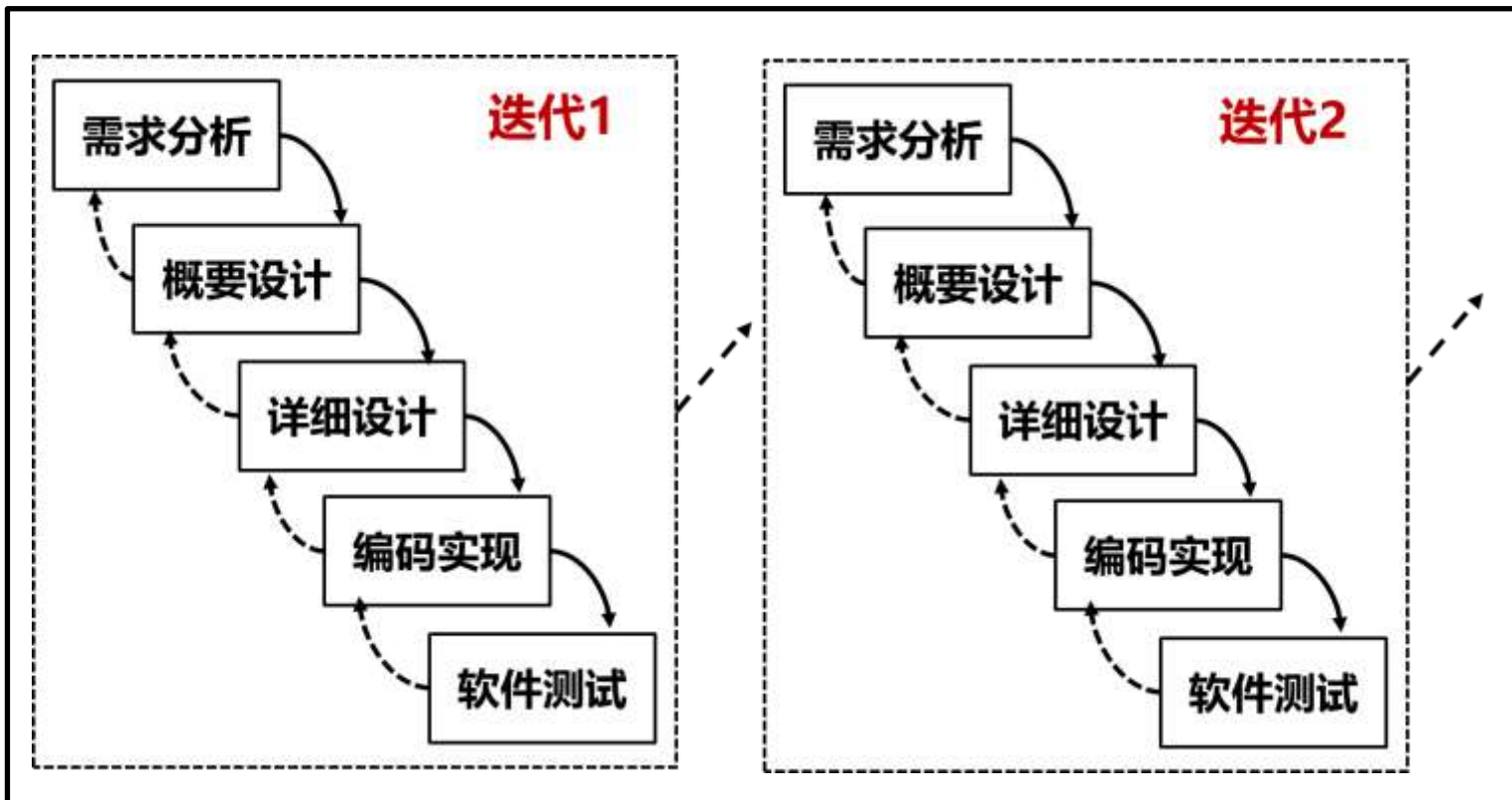


- 每次迭代是一完整过程
- 体现了小步快跑的开发理念
- 适合需求难导出、不易确定且持续变动的软件



迭代模型的局限性?

结合课程综合实践思考



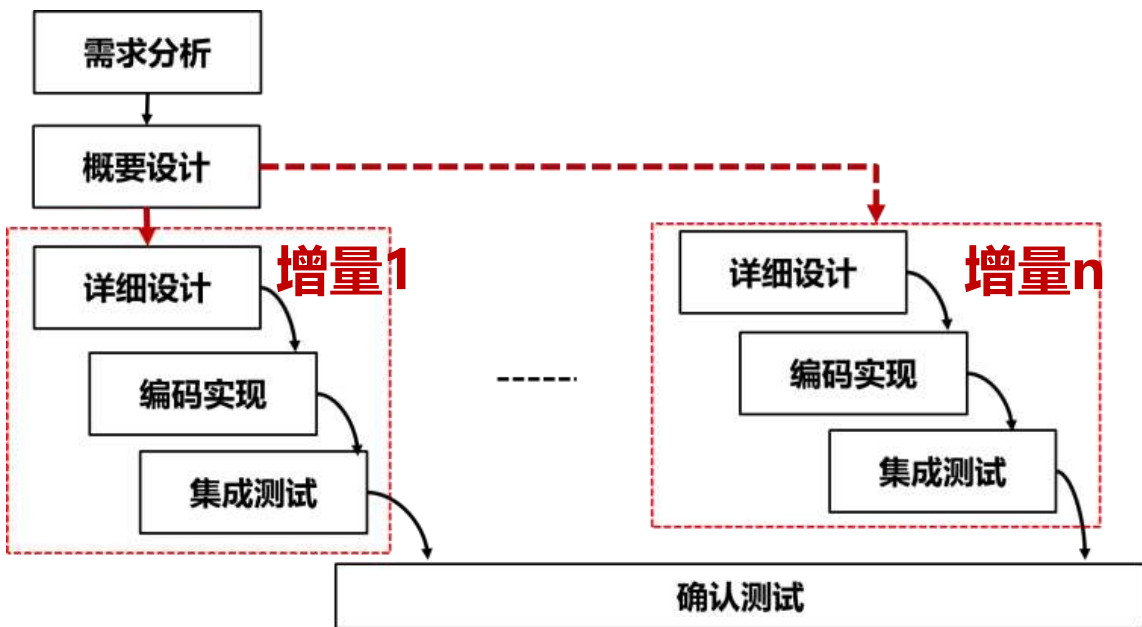
□ 不足

✓ 迭代多少次不确定

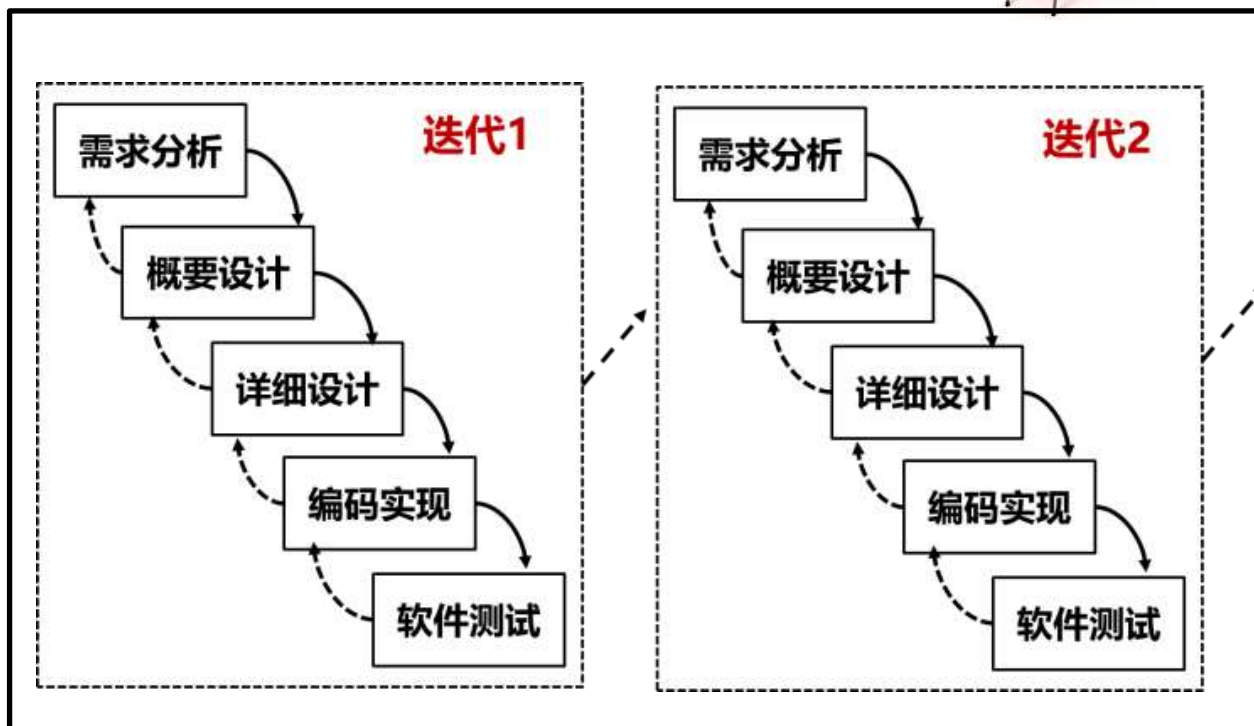
✓ 管理较为复杂



- 增量过程模型与迭代过程模型有何区别？



增量模型



迭代模型

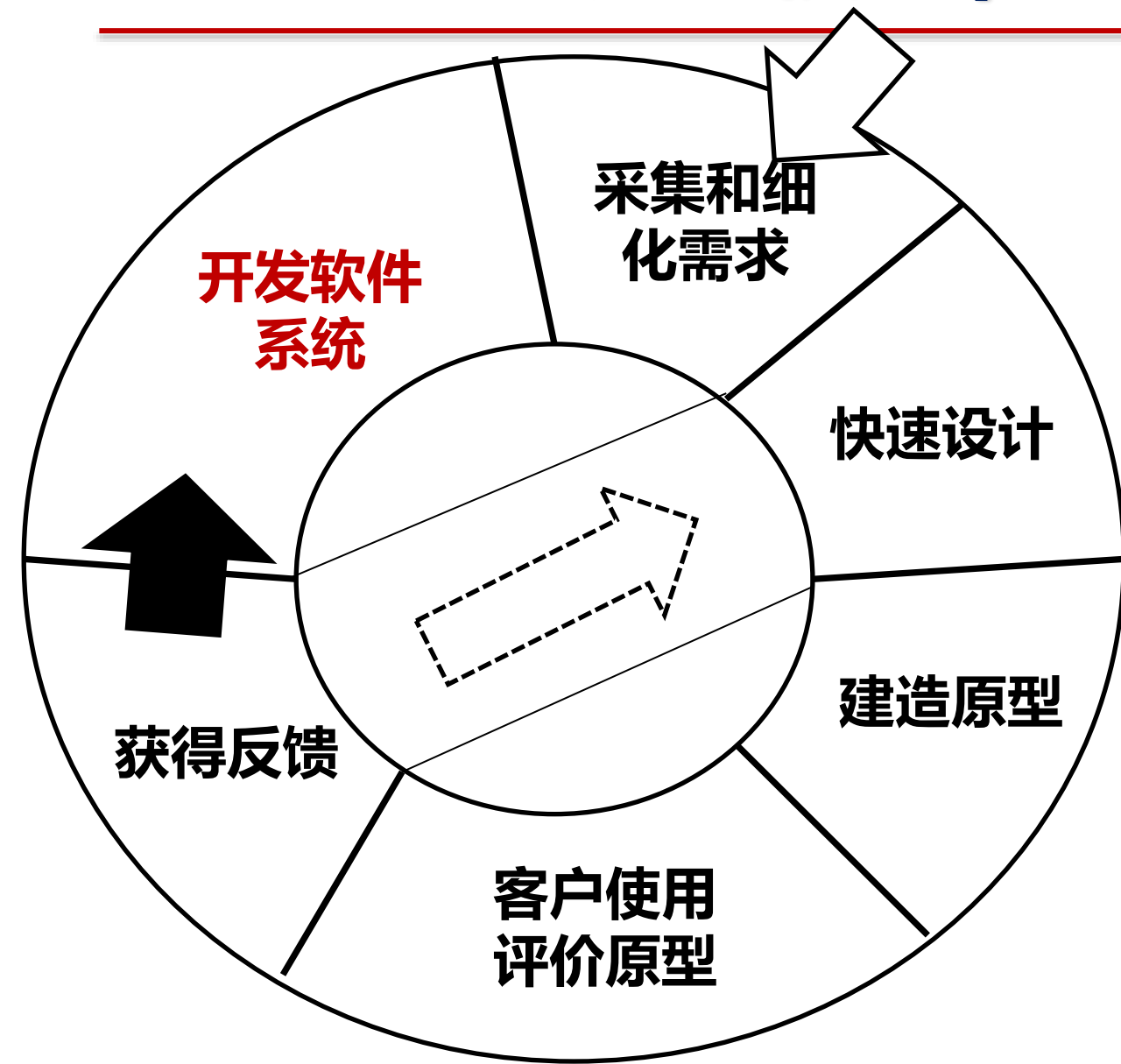
软件需求获取是一关键和瓶颈问题

- 软件需求非常关键
 - 软件开发的基础、验收的依据
- 用户讲不清楚软件需求有哪些、是什么？
 - 说不清、道不明
 - 尤其当软件较为复杂和庞大之时
- 用户与软件工程师对软件需求理解存在偏差
 - 对软件需求描述的歧义性、二义性、不准确等造成的
 - “应该是这样的”、“实际是这样的”

如果软件需求分析结果不正确、不完整、有歧义，会带来什么样的问题？



2.6 原型模型(Prototype Model)



□ 何为软件原型?

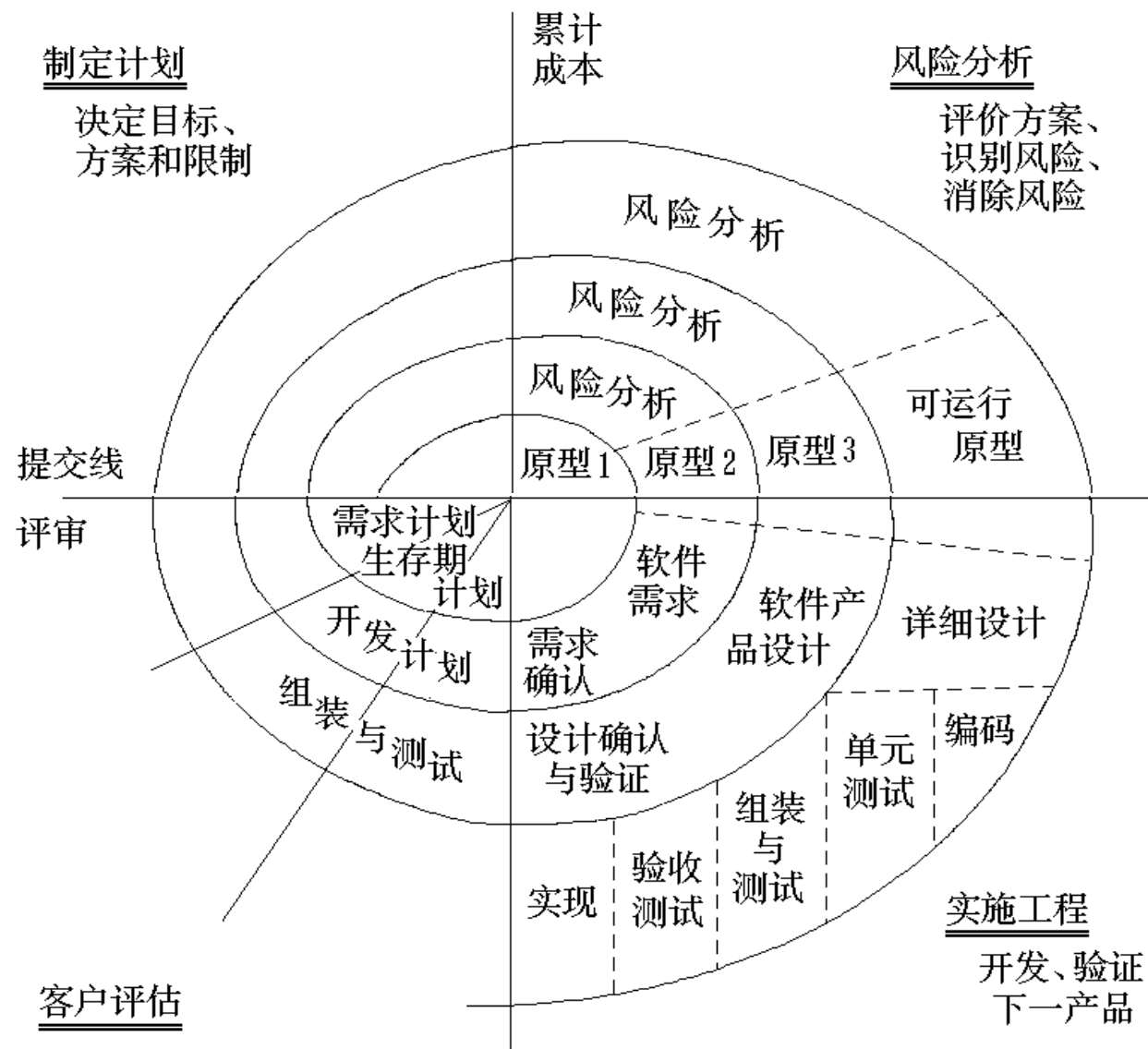
- ✓ 用户界面
- ✓ 执行流程
- ✓ 抛弃型原型
- ✓ 开发型原型



➤ 特点

- ✓ 软件原型作为交流载体和媒介
- ✓ 支持用户参与到软件开发中
- ✓ 持续、渐进地导出用户要求
- 适合于需求难导出、不易确定且持续变动的软件

2.7 螺旋模型(Spiral Model)



□ 软件风险

✓ 使软件开发受到影响和损失、甚至导致失败的、可能会发生的事件

- 集成迭代模型和原型模型
- 引入风险分析, 风险驱动的过程模型
- 每个迭代四个阶段, 若干活动
- 适合于需求不明确、开发风险高、开发过程中需求变更大的软件项目
- 不足: 管理复杂

不同软件过程模型的特点

模型名称	指导思想	关注点	适合软件	管理难度
瀑布模型	提供系统性指导	与软件生命周期相一致	需求变动不大、较为明确、可预先定义的应用	易
原型模型	以原型为媒介指导用户的需求导出和评价	需求获取、导出和确认	理解需求难以表述清楚、不易导出和获取的应用	易
增量模型	快速交付和并行开发	软件详细设计、编码和测试的增量式完成	需求变动不大、较为明确、可预先定义的应用	易
迭代模型	多次迭代，每次仅针对部分明确软件需求	分多次迭代来开发软件，每次仅关注部分需求	需求变动大、难以一次性说清楚的应用	中等
螺旋模型	集成迭代模型和原型模型，引入风险分析	软件计划制定和实施，软件风险管理，基于原型的迭代式开发	开发风险大，需求难以确定的应用	难

1. 何为软件过程模型

✓ 基本概念和特点

2. 有哪些软件过程模型

✓ 有什么类别，各有什么特点和优缺点

3. 如何来选择软件过程模型

✓ 软件过程模型的选择方式和策略



3.1 软件过程模型的选择

□考虑软件项目的特点

- ✓尤其是所开发软件的业务特点，如业务领域是否明确、软件需求是否易于确定、用户需求是否会经常性变化等等
- ✓是否可以预估到潜在的软件开发风险

□软件开发团队的水平

- ✓需要结合软件开发团队的能力和水平来选择过程模型，以防开发团队和管理人员无法掌控和驾驭过程模型

□分析软件过程模型特点

- ✓优缺点以及适合的场所

示例：如何选择合适的过程

□互联网应用软件开发过程模型

- ✓特点：软件需求不确定且快速变化
- ✓如：12306 APP软件，微信软件，淘宝软件
- ✓选用瀑布模型不合适，迭代模型较为合适



□装备软件开发过程模型

- ✓特点：软件需求确定且较为稳定
- ✓如：飞行控制软件
- ✓可考虑选用瀑布模型，用迭代模型不是很合适



3.2 传统软件过程模型的特点和不足

- 软件开发和运维的大量工作用于**撰写软件文档**，而非去编写程序代码
- 软件开发过程中会花费大量时间和精力用于**软件文档的评审**，以确保软件质量
- 一旦软件**需求发生变化**，开发人员需要修改软件需求文档，并据此来调整其他的一系列文档，最后再修改程序代码
- 等**较长时间**才能得到**可运行软件系统**

- 瀑布模型
- 增量模型
- 迭代模型
- 原型模型
- 螺旋模型
-

以文档为中心的重型软件开发方法，非常笨重

1. 何为敏捷开发方法

- ✓ 基本思想和原则
- ✓ 特点和应用

2. 具体敏捷开发方法

- ✓ 极限编程
- ✓ 测试驱动开发方法
- ✓ Scrum方法



1.2 什么是敏捷 (Agile) 方法?

□ 一种轻量级软件开发方法

✓ 相对于重量级的软件开发方法而言

□ 主张软件开发要以代码为中心，快速、轻巧和主动应对需求变化，持续、及时交付可运行的软件系统

✓ 轻便、轻巧

□ 提供了一组思想和策略，指导快速响应用户需求的变化，快速交付可运行的软件制品

1.3 敏捷开发方法的基本观点

- 较之于过程和工具，应更加重视**人和交互**的价值
- 较之于面面俱到文档，应更加重视**可运行软件系统**的价值
- 较之于合同谈判，应更加重视**客户合作**的价值
- 较之于遵循计划，应更加重视**响应用户需求变化**的价值

敏捷方法体现的思想

□强化可运行的软件，弱化文档

- ✓以可运行软件为中心来开展软件开发

□以适应变化为目的来推进开发

- ✓针对变化不断进行优化和调整任务、产品和计划等

□以人为本

- ✓敏捷软件开发是面向人的而不是面向过程的，让方法、技术、工具、过程等来适应人，而不是让人来适应它们

敏捷意味着：轻盈、灵巧；无过多的负担；迅速响应变化

1.4 敏捷准则(1/2)

- 尽早和持续地交付有价值的软件，以使用户满意
- 即使到了软件开发后期，也欢迎用户需求的变化
- 不断交付可运行的软件系统，交付周期可以从几周到几个月
- 在整个软件项目开发期间，用户和开发人员最好能每天一起工作
- 由积极主动的人来承担项目开发，给他们提供所需环境和支持，信任他们的能力
- 团队内部最有效的信息传递方式是面对面的交谈

敏捷准则(2/2)

- 将**可运行软件**作为衡量软件开发进度的首要标准
- 可持续性的开发，出资方、开发方和用户方应当保持**长期、恒定的开发速度**
- 关注**优秀的技能和良好的设计**会增强敏捷性
- 简单化**，着眼于当前的问题，不把问题复杂化
- 最好的架构、需求和设计出自于**自组织的团队**
- 软件开发团队应定期就如何提高工作效率的问题进行**反思**，并进行相应的**调整**

1. 何为敏捷开发方法

- ✓ 基本思想和原则
- ✓ 特点和应用

2. 具体敏捷开发方法

- ✓ 极限编程
- ✓ 测试驱动开发方法
- ✓ Scrum方法



2.1 极限编程的基本思想

□由Kent Beck提出的一种特殊的敏捷软件开发方法

□四条核心思想

- ✓**交流**，强调基于口头（而非文档、报表和计划）的交流
- ✓**反馈**，通过持续、明确反馈来获得软件状态
- ✓**简单**，用最简单的技术来解决问题
- ✓**勇气**，快速开发并在必要时具有重新进行开发的信心

□将经过数十年检验的准则结合在一起，定义了五条指导性原则和十二条须遵循的核心准则

2.2 极限编程的5条指导原则

□ 反馈速快

- ✓ 从用户处迅速得到有关软件的反馈，确认开发是否满足用户需求，通过自动化测试迅速了解软件运行状况

□ 设假性单简

- ✓ 开发人员只考虑当前迭代所面临问题，无需考虑下一次迭代的问题，用简单方法和技术来解决问题。

□ 改更步逐

- ✓ 通过一系列修改来逐步解决问题和完善系统，不要期望一次迭代就开发出完整的软件系统。

□ 支持化变

- ✓ 欢迎用户改变需求，支持用户需求动态变化。

□ 作工的量高

- ✓ 采用诸如测试驱动开发等技术高质量地开展工作，确保软件质量。

2.3 极限编程的12条核心准则(1/3)

□计划游戏

- ✓ 软件开发团队快速制定下一次迭代的软件开发计划

□隐喻(Metaphor)

- ✓ 使用业务相关术语（而不是技术术语）来描述需求，促使开发人员和业务人员对系统达成共同和一致的理解

□小型发布

- ✓ 经常性发布可运行软件系统，每次发布的软件系统仅提供少量功能

□简单设计

- ✓ 只为当前的需求做设计，程序能运行所有测试、没有重复逻辑、包含尽可能少的类和方法

极限编程的12条核心准则(2/3)

□测试驱动开发

- ✓首先编写测试程序和设计测试用例，再正式编写目标软件的程序

□重构

- ✓在不改变程序代码功能的前提下，改进程序代码的设计，使程序代码更加简单，更易于扩展

□结对编程

- ✓两名程序员同时在一台计算机上共同开展编程工作

□代码集体拥有

- ✓开发小组的任何成员都可以查看并修改任何部分的代码

极限编程的12条核心准则(3/3)

□持续集成

- ✓ 经常性地集成，周期尽可能短

□每周工作40小时

- ✓ 倡导质量优先，不为了追求速度片面延长工作时间

□现场用户

- ✓ 用户代表在现场办公，参与开发全过程，确保能及时得到反馈

□编码标准

- ✓ 遵循统一编码标准，以提高软件系统的可理解性和可维护性

2.4 传统软件开发的局限

□程序员先编写程序代码，然后再对程序代码进行测试

- ✓基于已有的程序代码进行测试

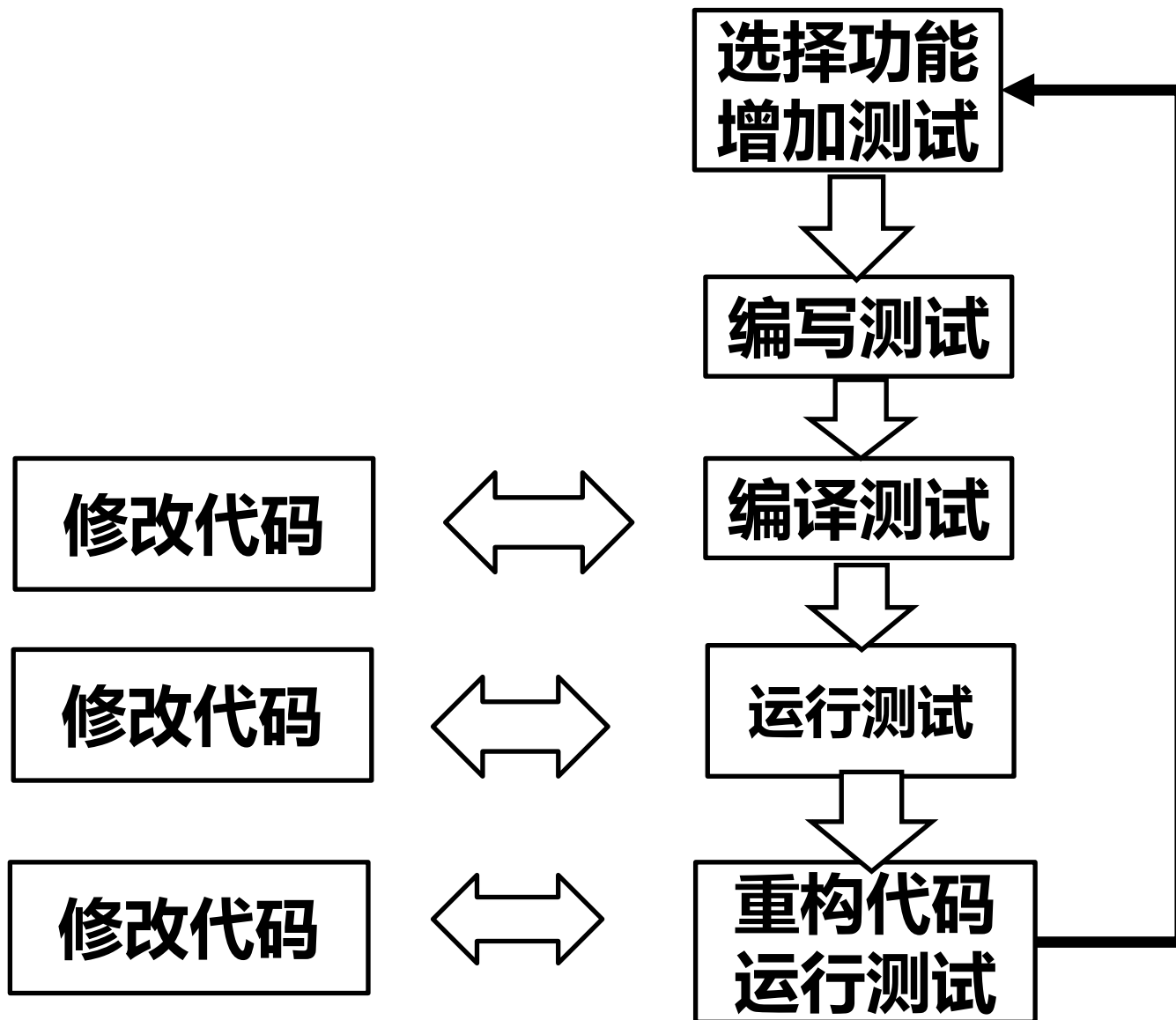
□局限性

- ✓**测试常被视为是附加工作**，由于进度压力，常被忽视，导致没有足够时间对代码进行详尽和充分的测试
- ✓当文档与程序代码不一致时，对程序代码进行测试就会存在许多问题
- ✓测试通常是在程序代码编写完成之后才进行的，因而**无法保证编写程序和测试同步**
- ✓测试被视为是乏味的工作，人员缺乏积极性和成就感

2.5 测试驱动开发的思想

- 在开发程序代码之前，先确定和编写测试
- 程序员首先要思考如何对某个功能进行测试，设计好相应的测试用例，编写好相关的测试代码，然后编写相应的程序代码以通过软件测试

测试驱动开发的过程



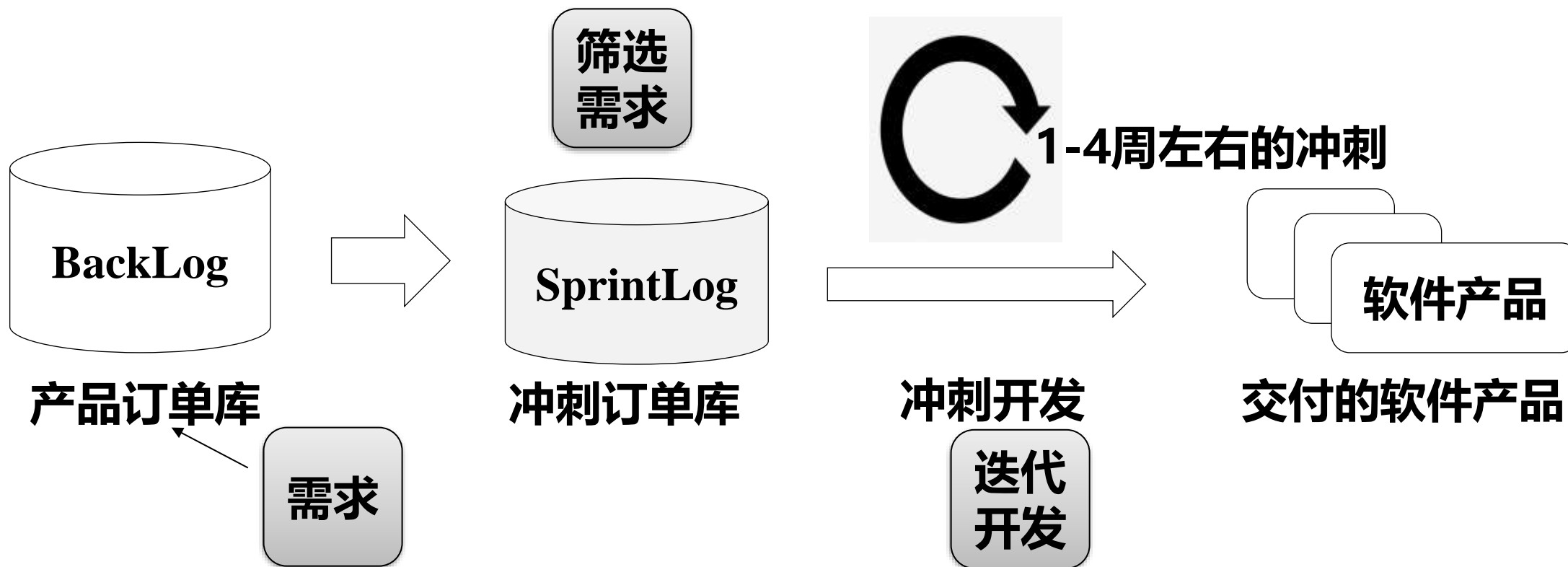
测试驱动开发的特点

- 根据测试来编写代码
- 编写测试的目的不仅是为了测试程序代码能够正常工作，而且被用于定义程序代码的内涵
- 确保任何程序代码都是可测试的
- 编码完成后即完工
- 易于维护、质量保证



2.6 Scrum方法

- 旨在通过增量或迭代的方式加强软件项目的管理
- 开发团队应如同橄榄球队一样，每个人都有明确的角色和分工，目标一致，高效协作完成软件开发任务



案例一：某电商网站的敏捷开发

项目背景

该电商网站面临激烈的市场竞争，用户需求变化频繁。
项目团队采用敏捷开发方法，以快速响应用户需求，提高网站的竞争力。

开发过程

项目团队采用Scrum框架，每个Sprint持续两周。
在开发过程中，团队通过每日站会及时沟通进展和问题，通过Sprint评审会议向客户展示可工作的软件，根据客户反馈进行调整。



项目成果

通过敏捷开发，该电商网站在短时间内完成了多项功能优化和新功能上线。
敏捷开发使项目团队能够快速响应用户需求，提高了项目的成功率和市场竞争力，同时也增强了客户对团队的信任和合作意愿。

案例二：某移动应用的敏捷开发



项目背景

该移动应用是一款社交类应用，用户需求变化迅速，市场竞争激烈。项目团队采用敏捷开发方法，以快速迭代和持续交付为目标，提高应用的市场竞争力。



开发过程

项目团队采用XP方法，结合Scrum框架。团队采用结对编程和测试驱动开发，确保代码质量和开发效率；同时通过Scrum的会议机制，及时沟通和调整开发计划。在每个迭代周期中，发布新的版本，获取用户反馈，根据反馈进行优化和调整。



项目成果

通过敏捷开发，该移动应用在短时间内完成了多项功能更新和优化。例如，视频分享功能的优化使用户活跃度提高了20%，应用的下载量也显著增加。快速响应市场变化和用户需求，提高了应用的市场竞争力和用户满意度。

2.7 敏捷方法的特点

□小

- ✓ 生成少量软件文档，每个文档规模要小
- ✓ 每次迭代要实现软件功能的数量和规模要小，迭代周期要小

□简

- ✓ 技术、工具以及每次迭代要解决的问题尽可能简单
- ✓ 只关注当前欲实现的功能需求，而不要考虑将来的问题

□快

- ✓ 快速响应变化、从用户处获得反馈，给用户提交有价值软件，对软件产品进行迭代和更新

□变

- ✓ 允许需求动态变化，要以变应变，开发团队应是自组织的

□敏捷开发方法的本质

- ✓ 应对软件**需求变化**，解决传统过程模型的不足

□敏捷开发方法的特点

- ✓ 小、简、快、变、体，**轻量级方法**，以**代码为中心**的方法

□敏捷开发方法的构成

- ✓ 由许多具体的方法组成，如Scrum方法、极限编程等

1. 何为开源软件

✓特点和开发要求

2. 何为群体化软件开发方法

✓基本理念和思想

3. 如何实现群体化软件开发

✓关键软件工程技术



1. 何为开源软件

✓特点和软件开发要求

2. 何为群体化软件开发

✓基本理念和思想

3. 如何实现群体化软件开发

✓关键软件工程技术





2.1 软件开发是创作和生产的过程

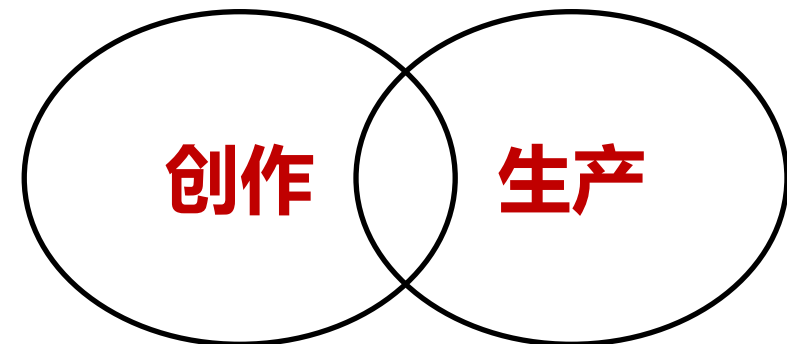
□软件创作

- ✓发挥软件工程师的智慧，结合创作者的爱好和兴趣，开展软件创作
- ✓如构思需求、开展设计、精雕代码等

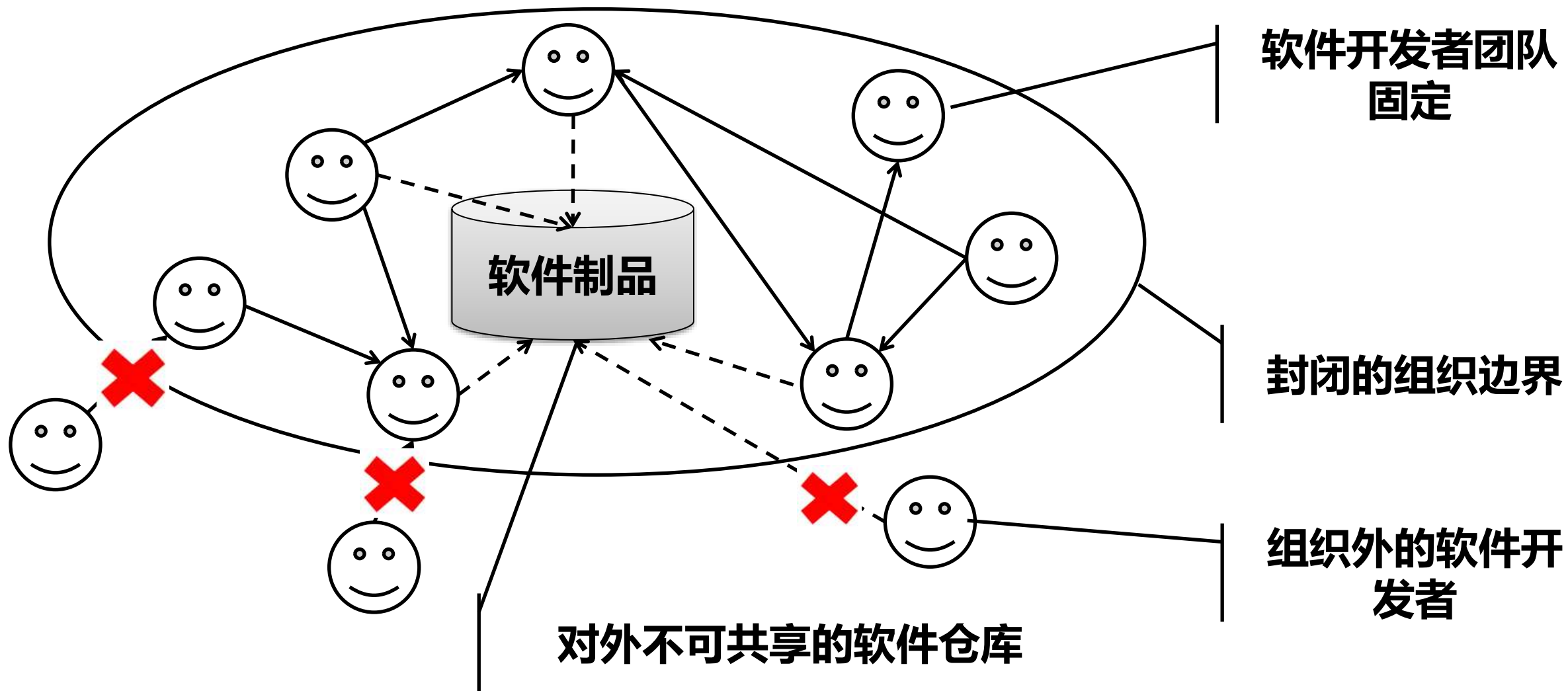


□软件生产

- ✓任务分工，集中管理
- ✓计划驱动，有序开发
- ✓评审测试，保证质量
- ✓交流沟通，促进合作



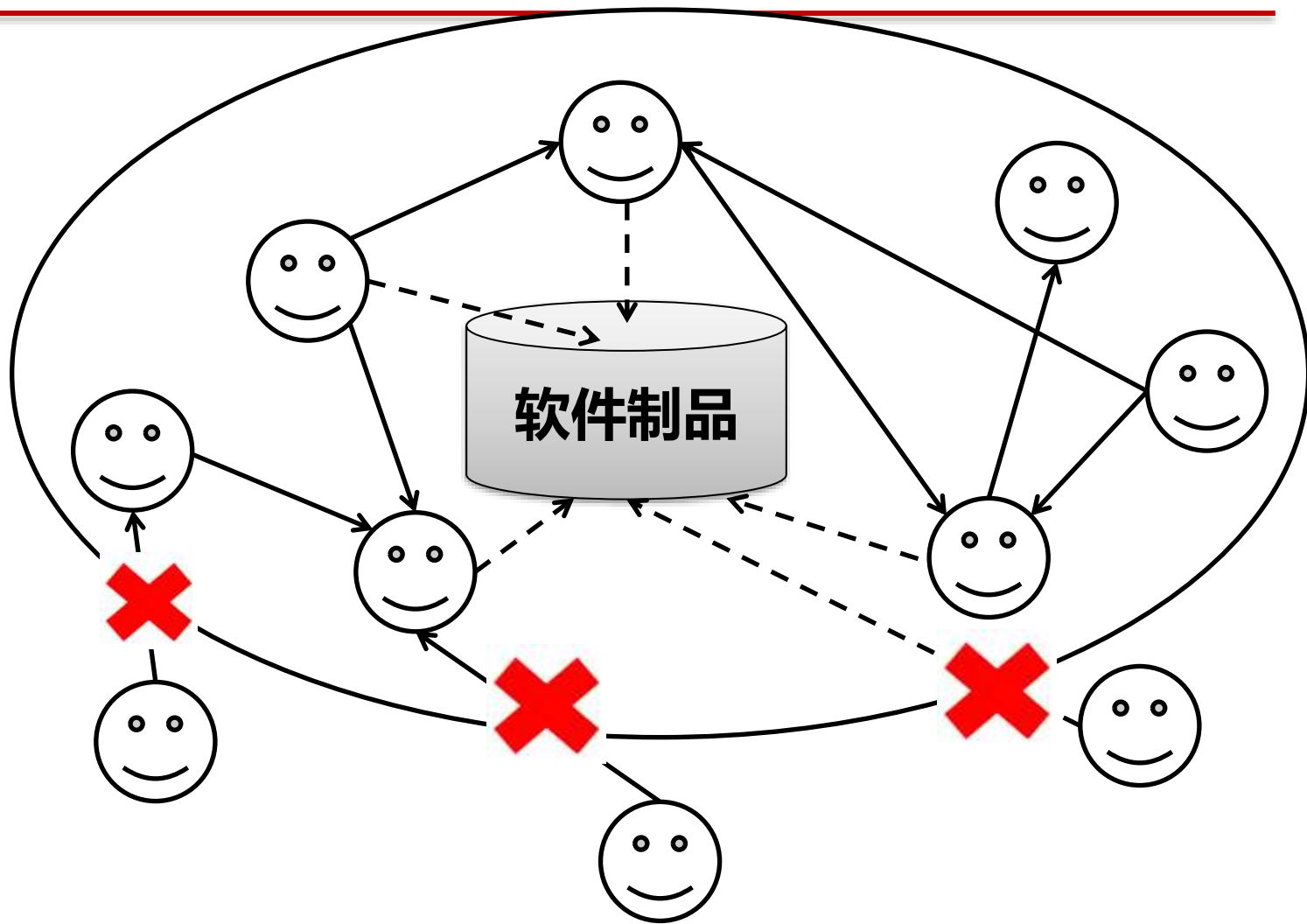
基于团队的软件开发方法及其组织模式



特定组织（如微软、IBM、华为）的人员所组成软件项目团队

基于团队软件开发方法的特点

- 开发团队**边界封闭**
- 域外人员**无法参与**
- 人员确定**资源有限**
- 项目的成果**不共享**
- **集中化**的管理模式
- **关注生产**而非创作

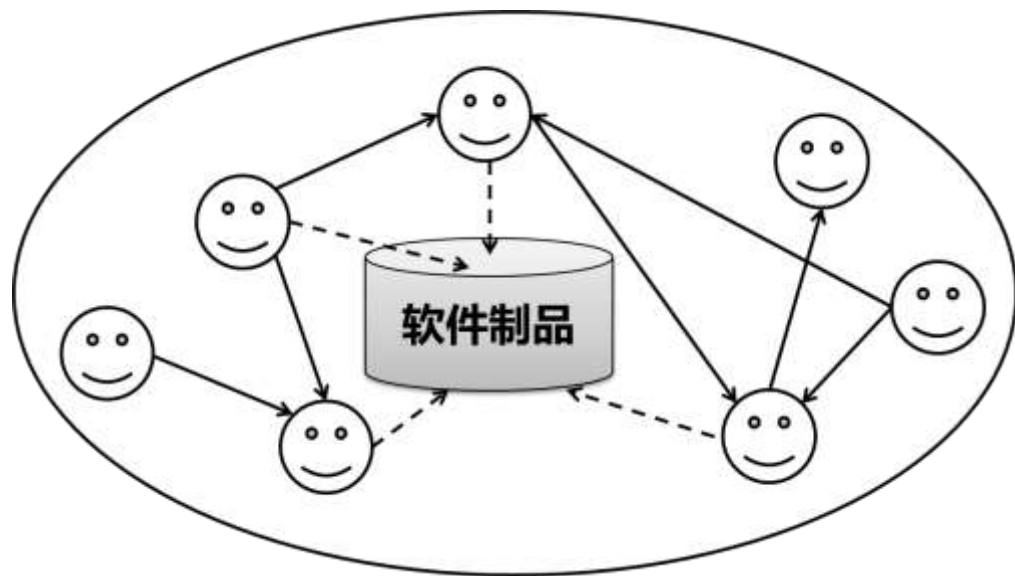


无法充分利用团队之外的力量，阻碍大众参与软件创作和生产

示例：Windows软件开发团队组成

□ Windows 7软件项目的团队组织

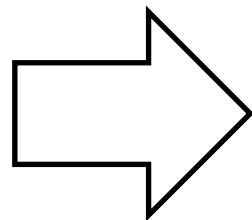
- ✓ **核心开发团队**大约有1000人
- ✓ 25个**功能小组**，每个小组大约有40个人
- ✓ 每个小组包括**三类人员**：程序经理，开发工程师，测试工程师
- ✓ **全部是Microsoft员工**



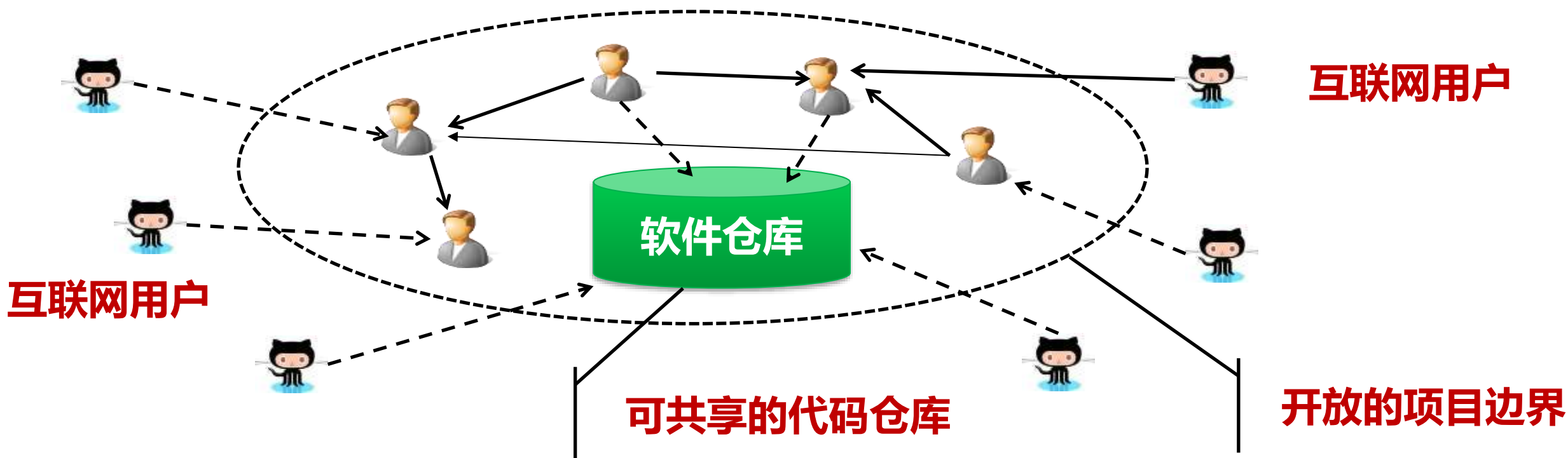
Windows 7源代码是微软商业机密，不对外共享和开放

2.2 开源软件项目的开发和组织模式

- 开放项目边界
- 大众参与开发
- 软件代码共享

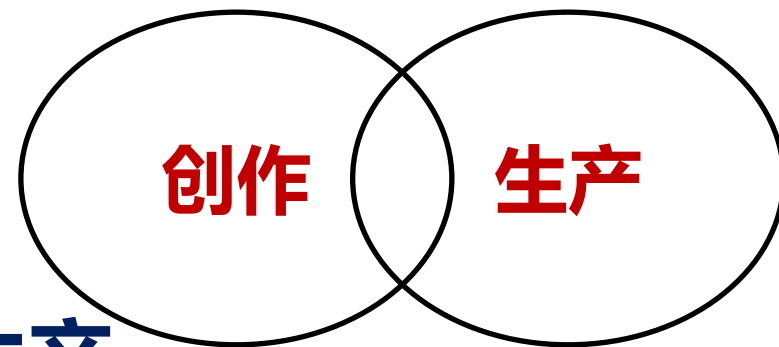


由少数核心开发人员和大量互联网大众（外围）所组成的群体化开发

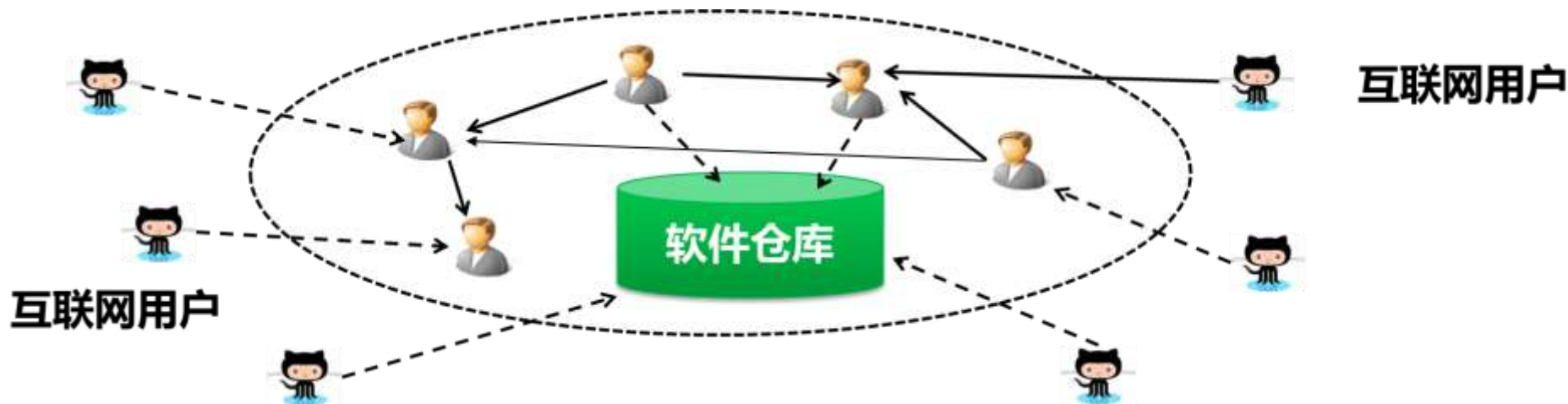


互联网大众能干些什么？

- 构思软件需求，增强软件功能-创作
- 发现软件问题，指出软件缺陷-创作
- 编写程序代码，提交开发成果-创作+生产
- 参与代码评审，评价贡献质量-生产
-



高手在民间，高手在互联网上



将软件创作和生产融合在一起，充分发挥大众的智慧 and 力量

OpenHarmony开源软件中的群体化贡献

Gitee 开源软件 企业版 高校版 私有云 博客 搜开源 登录 注册

OpenHarmony    关注 31K

概览 仓库 281 任务 1189 Pull Requests 676 动态 成员 128

快速搜索...

状态: 正常 类型: 全部 语言: 全部 排序: 最近更新

提出问题和需求 **贡献和提交代码**

security_permission
Application permission management and IPC
最近更新: 4分钟前

aafwk_standard
Ability management framework | 元能力框架
最近更新: 7分钟前

test_developertest
Development self-test framework | 开发者自测试框架

仓库语言

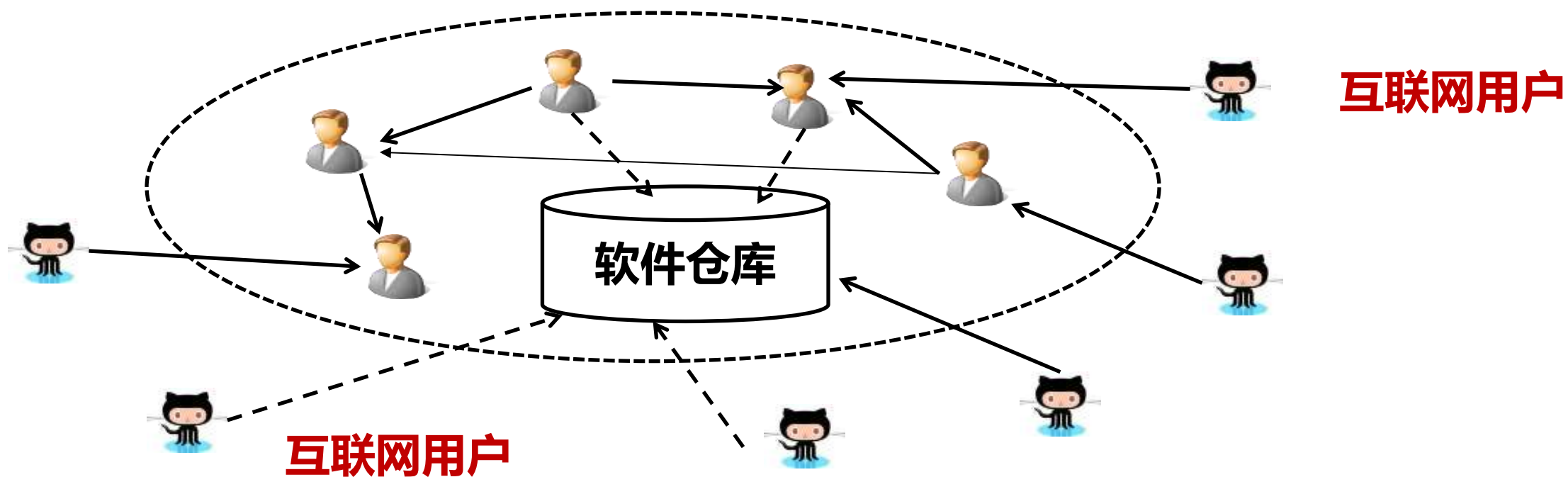
C	42%
C++	37%
Python	4%
Shell	4%
JavaScript	?
Makefile	2%
Java	
TypeScript	1%
HTML	1%
其他	4%

开源鸿蒙充分利用群体的智慧来推动软件的创作和生产



2.3 何为群体化开发方法

□ 依托**互联网平台**来**吸引、汇聚、组织和管理**互联网上的大规模软件开发人员，通过**竞争、合作、协商**等多种自主协同方式，让他们**参与软件开发、分享软件开发知识和成果、贡献智慧和力量**的一种新颖软件开发方法



支持群体化开发的互联网平台

□ Github（国际）和 Gitee（国内）



Why GitHub? Team Enterprise Explore Marketplace Pricing

Where the world builds software

Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world.

www.github.com



gitee 开源软件 企业版 高校版 私有云 博客

企业级 DevOps 研发管理平台

帮助开发者/团队/企业更好地管理代码，让软件研发更高效

[注册 Gitee](#) [免费开通企业版](#)

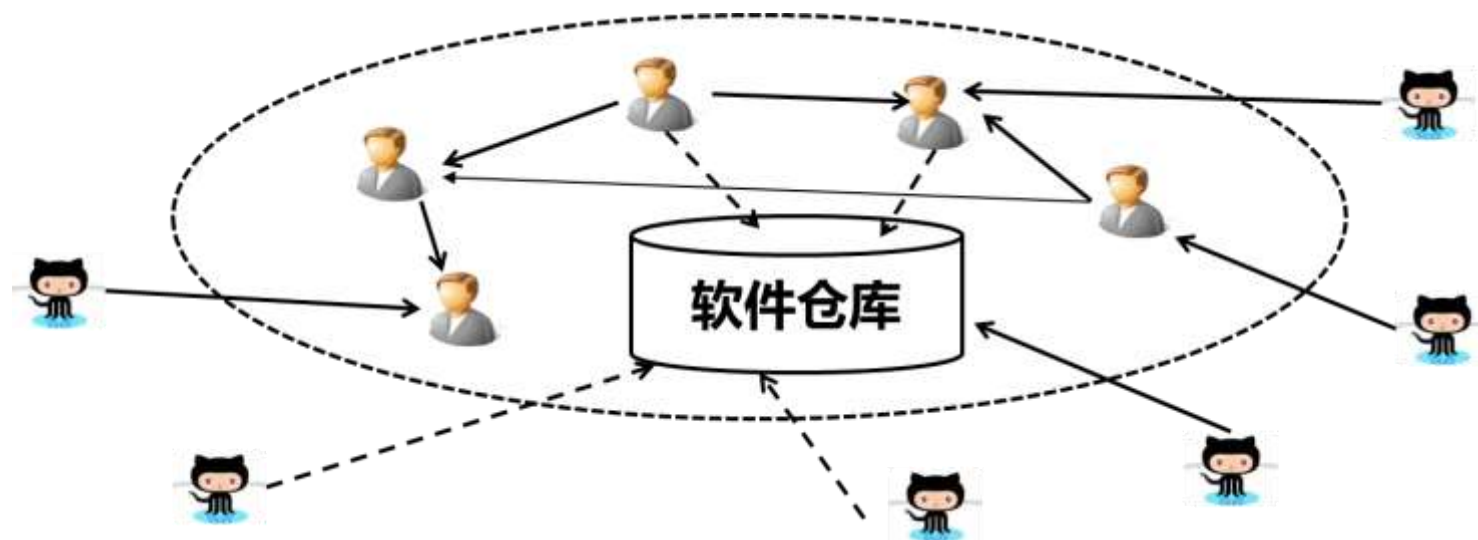
开发者	代码仓库	企业客户	高校
600万+	×1500 万+	18 万+	2500 +

1024

www.gitee.com

2.4 群体化软件开发方法的特点

- 软件开发边界开放
- 互联网大众自由参与
- 利用海量的大众资源
- 共享源程序代码
- 兼顾软件创作和生产
- 依托互联网平台



开发社区而非团队：社区和团队的区别

群体化软件开发是一种基于社区的软件开发模式

1. 何为开源软件

✓特点和开发要求

2. 何为群体化软件开发

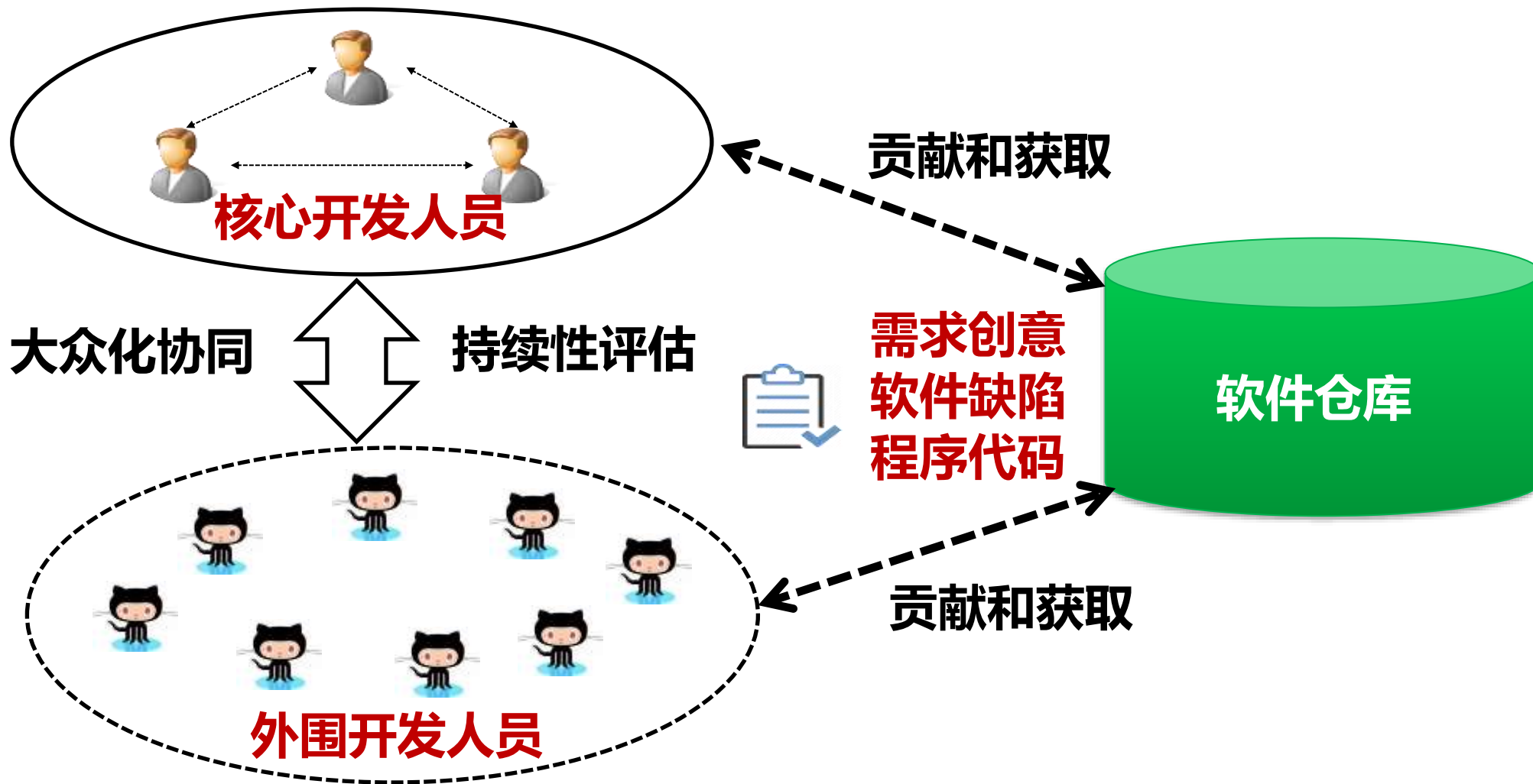
✓基本理念和思想

3. 如何实现群体化软件开发

✓关键软件工程技术



3.1 基于社区的群体化软件项目组织

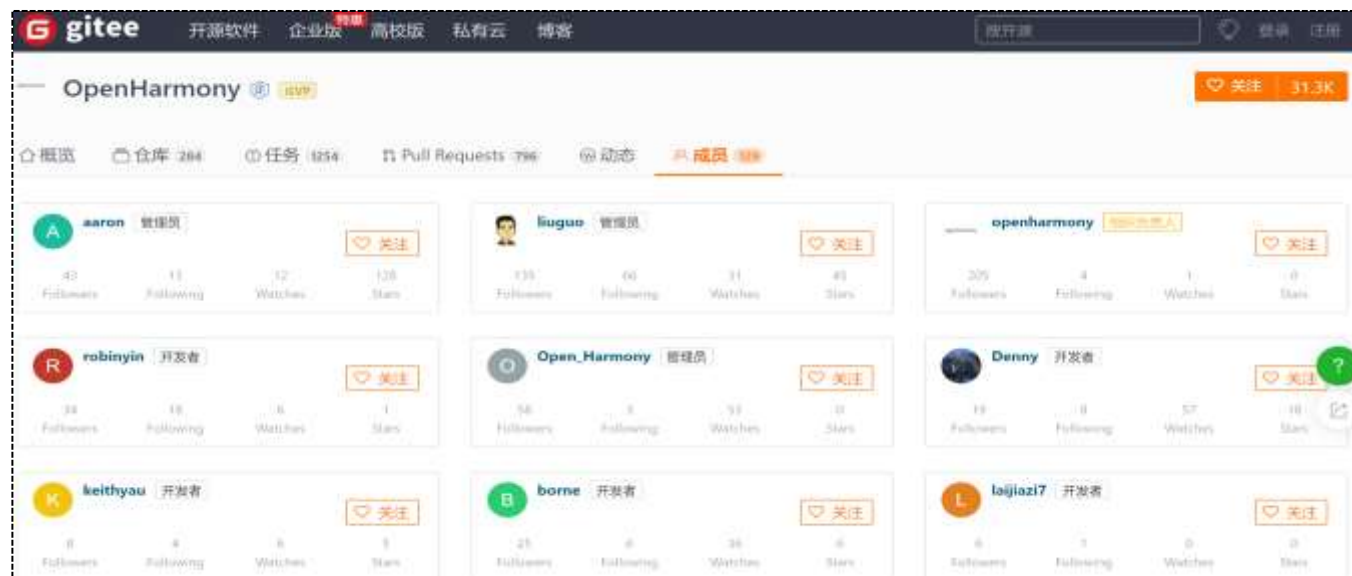


□ 开源软件项目的**主要贡献者**，**主要职责**

- ✓ 开源软件的**核心贡献**人员（如提供了最初的开源代码）
- ✓ **分析和评估**外围开放大众所发现的软件缺陷和提出的软件需求，将其转化为生产性的软件开发计划
- ✓ **评审和分析**开放大众所提交的程序代码，并将通过评审的代码融入到软件仓库之中

□ **人数不多**

- ✓ 开源鸿蒙有129人

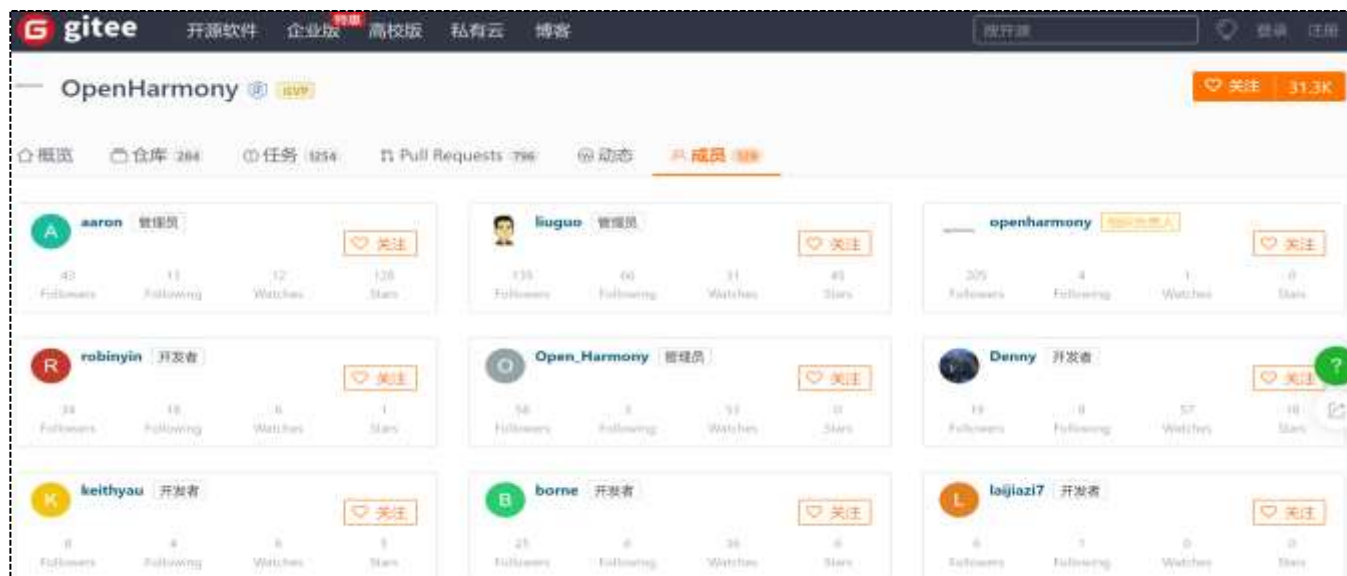


□ 开源软件项目的外围贡献者，主要职责

- ✓ 借助于互联网平台获得软件项目信息，结合个人的兴趣爱好、经验和技能，自发地为软件项目**贡献自己的力量**
- ✓ 提出需求建议、汇报代码缺陷、提交程序代码、评审代码质量等

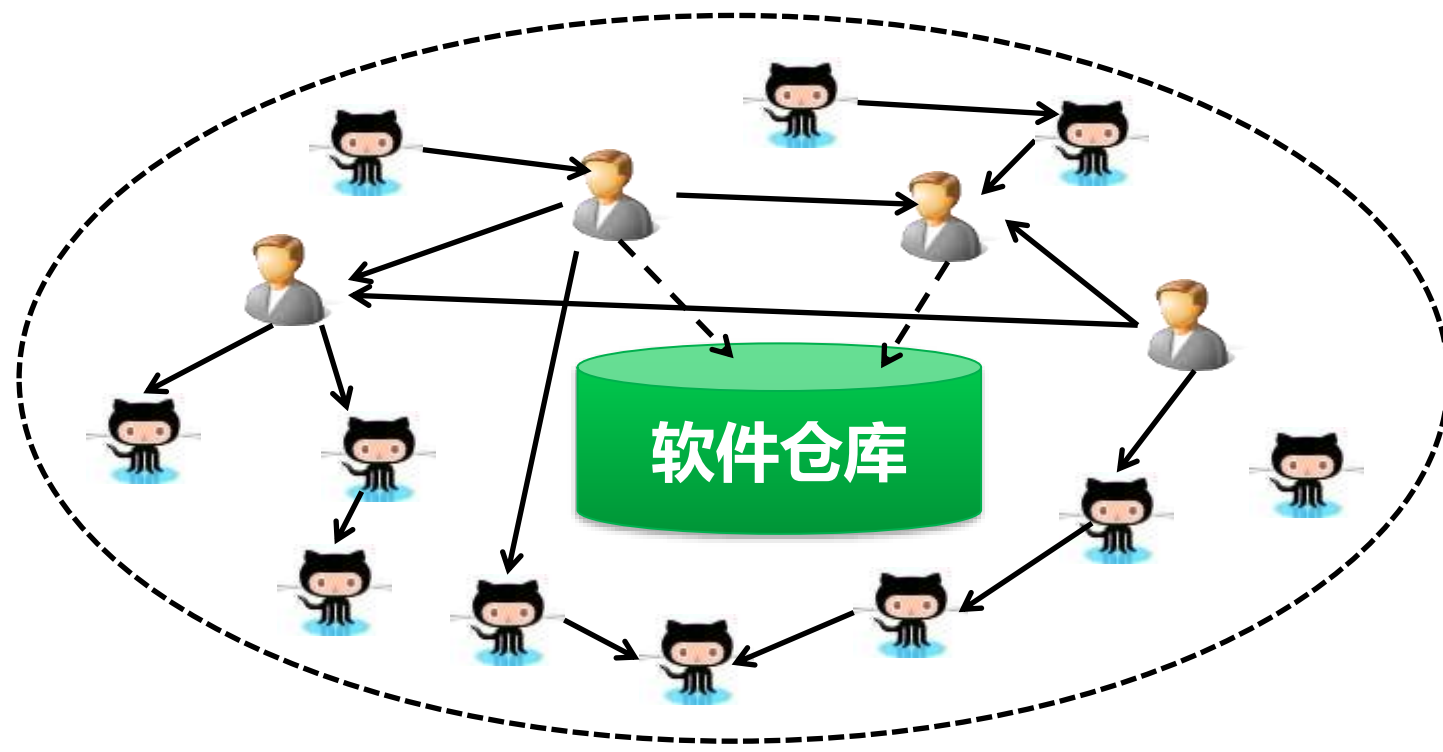
□ 这类人群的数量会非常大，成百上千，甚至有几万人

- ✓ 开源鸿蒙有3万多人



依托开源社区来组织不同人员

□ **开源软件社区**将核心开发与外围开发人员有机地结合在一起，依托软件仓库进行分布式协同开发



开源软件社区

- 自由进出社区
- 遵循社区规定
- 自愿开展工作
- 开放分享代码

示例：依托社区自由获取开源代码

□克隆、下载开源代码

开源项目 > 鸿蒙开源项目 > 硬件驱动(Drivers)

OpenHarmony / drivers_framework

Watch 27 Star 165 Fork 163

代码

Issues 33

Pull Requests 9

Wiki

统计

DevOps

服务

加入 Gitee

与超过 600 万 开发者一起发现、参与优秀开源项目，私有仓库也完全免费：)

免费加入

已有帐号？立即登录

自由地克隆和下载代码

master 分支 6 标签 9

文件

Web IDE

克隆/下载

openharmony_ci !359 Platform dir adjustment 27f5782 12小时前

601 次提交

.gitee add issue and pr template 7个月前

core solve the warings of C90 1个月前

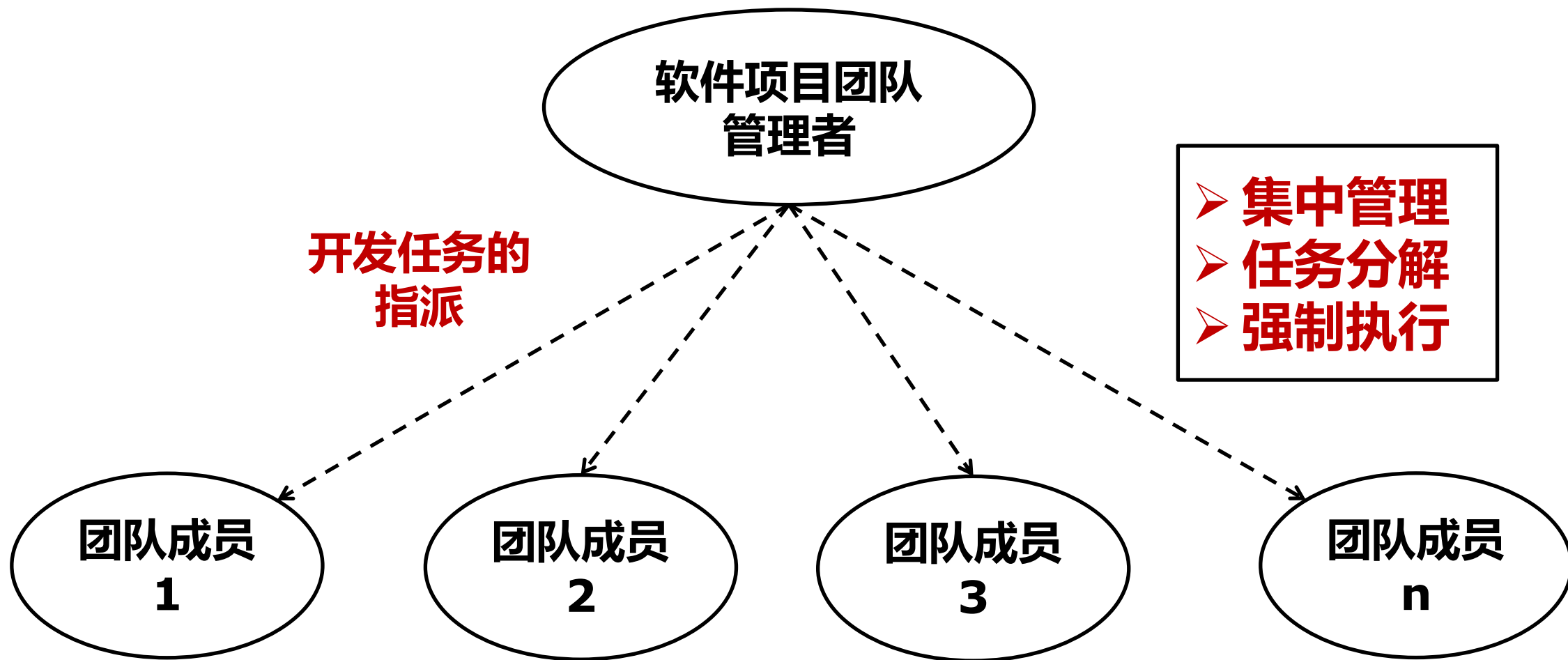
简介

Framework of the Hardware Driver Foundation (HDF) | HDF驱动框架

暂无标签

C 等 6 种语言

传统软件开发的任务管理



这种方法适用于群体化软件开发方法吗？为什么？



3.2 基于Issue的任务管理

□ 开源软件的开发任务

- ✓ 不同于传统的开发任务分解方式
- ✓ 开发者群体自主提出，体现了群体创作思想
- ✓ 每个任务对应于一个 **Issue**
- ✓ **开发任务的二类形式**：修复软件缺陷、功能实现需求

- **自主提出任务**
- **自发完成任务**

□ 基于Issue的软件开发任务管理

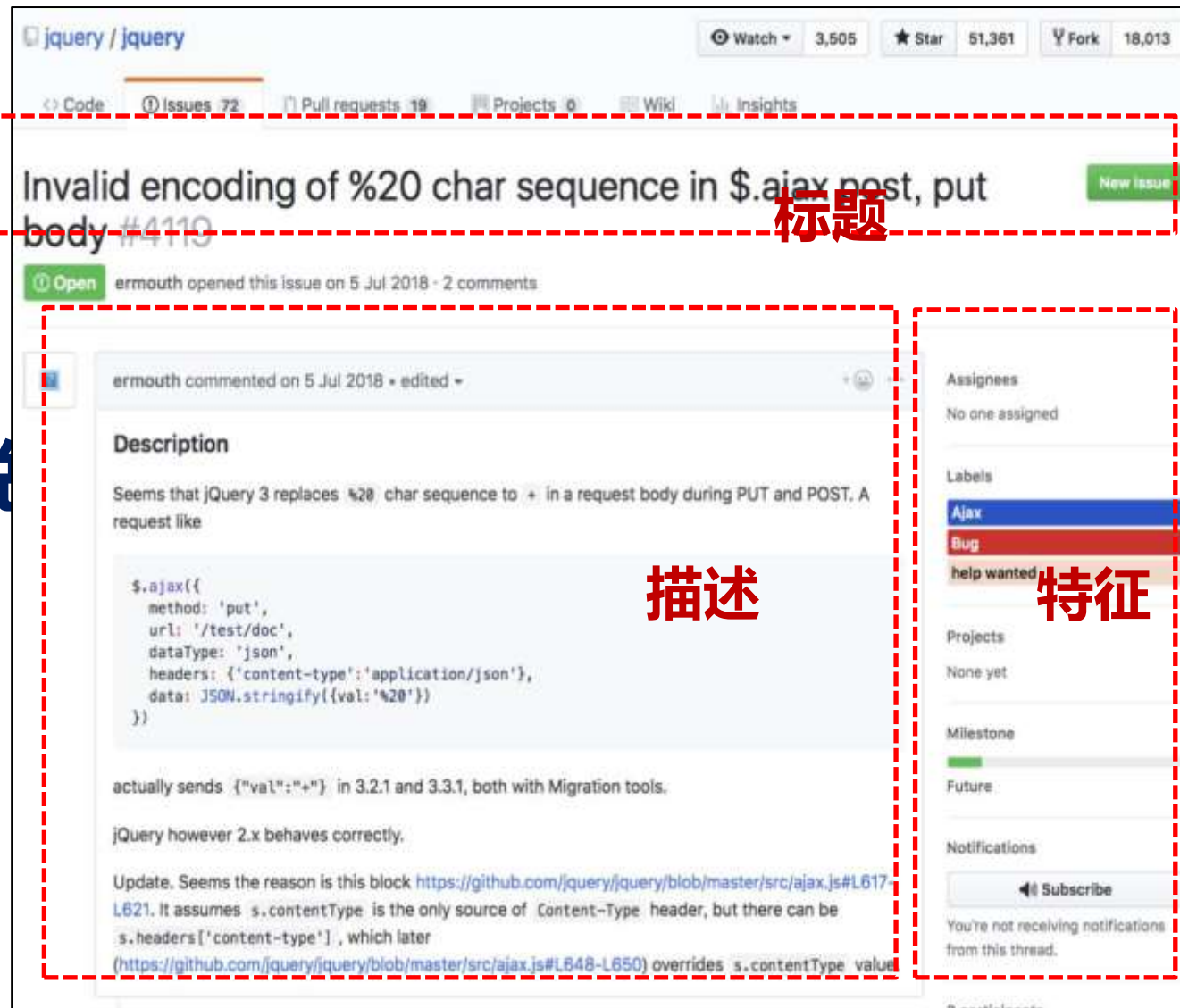
- ✓ **创建Issue**，提出软件开发任务
- ✓ **讨论Issue**，分析开发任务的意义和价值
- ✓ **指派Issue**，安排人员来完成Issue
- ✓ **掌控Issue**，掌握Issue解决的进展状况

□清晰地描述任务

- ✓标题
- ✓内容
- ✓特征

□提供必要的标签以补充说明

- ✓缺陷 or 需求
- ✓程序语言



The screenshot shows a GitHub issue page for the jQuery repository. The issue title is "Invalid encoding of %20 char sequence in \$.ajax post, put body #4110". The issue is labeled as "Open" and was opened by "ermouth" on July 5, 2018. The description explains that jQuery 3 replaces the %20 character sequence with a space character in a request body during PUT and POST requests. A code block shows the \$.ajax configuration. The issue is labeled with "Ajax", "Bug", and "help wanted". The right sidebar shows the "Assignees" section with "No one assigned" and the "Labels" section with the selected labels. The "Subscribe" button is at the bottom right.

Invalid encoding of %20 char sequence in \$.ajax post, put body #4110

ermouth opened this issue on 5 Jul 2018 - 2 comments

ermouth commented on 5 Jul 2018 • edited •

Description

Seems that jQuery 3 replaces %20 char sequence to + in a request body during PUT and POST. A request like

```
$.ajax({
  method: 'put',
  url: '/test/doc',
  dataType: 'json',
  headers: {'content-type': 'application/json'},
  data: JSON.stringify({val: '%20'})
})
```

actually sends {"val": "+"} in 3.2.1 and 3.3.1, both with Migration tools.

jQuery however 2.x behaves correctly.

Update. Seems the reason is this block <https://github.com/jquery/jquery/blob/master/src/ajax.js#L617-L621>. It assumes s.contentType is the only source of Content-Type header, but there can be s.headers['content-type'], which later (<https://github.com/jquery/jquery/blob/master/src/ajax.js#L648-L650>) overrides s.contentType value

Assignees

No one assigned

Labels

Ajax

Bug

help wanted

Projects

None yet

Milestone

Future

Notifications

Subscribe

You're not receiving notifications from this thread.

示例：提出Issue

- 任何人都可提出软件开发任务：代码缺陷 和 软件需求
- “新建Issue”



□辨别Issue的类别

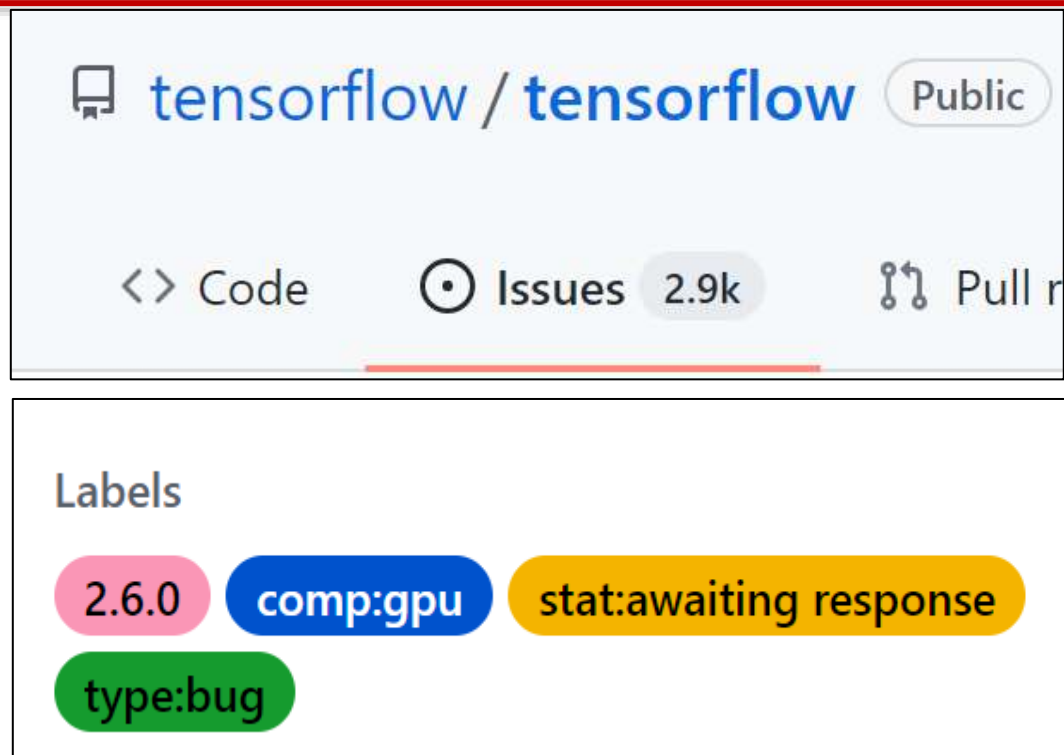
- ✓ 代码缺陷，新增功能需求

□确认Issue的有效性

- ✓ 缺陷是否客观存在
- ✓ 软件需求是否有意义

□为Issue贴上适当标签

- ✓ 直观地展示其特征信息
- ✓ “Bug” 表示缺陷修复任务，“duplicate” 标签表示这是一个重复的Issue




示例：管理和讨论Issue

  OpenHarmony / kernel_liteos_a 

 代码

 Issues 65

 Pull Requests 22

 Wiki

 统计

Issues / 详情

3516DV300开关机压测出现系统挂死

 Zhaotianyu  3个月前

请补充一下log截图

  Ieland11 将 优先级 设置为 严重 3个月前

  Ieland11 将 负责人 从 zhangfanfan2 修改为 zhangg05283106 3个月前

- 设置Issue优先级
- 补充log截图

□何为指派Issue

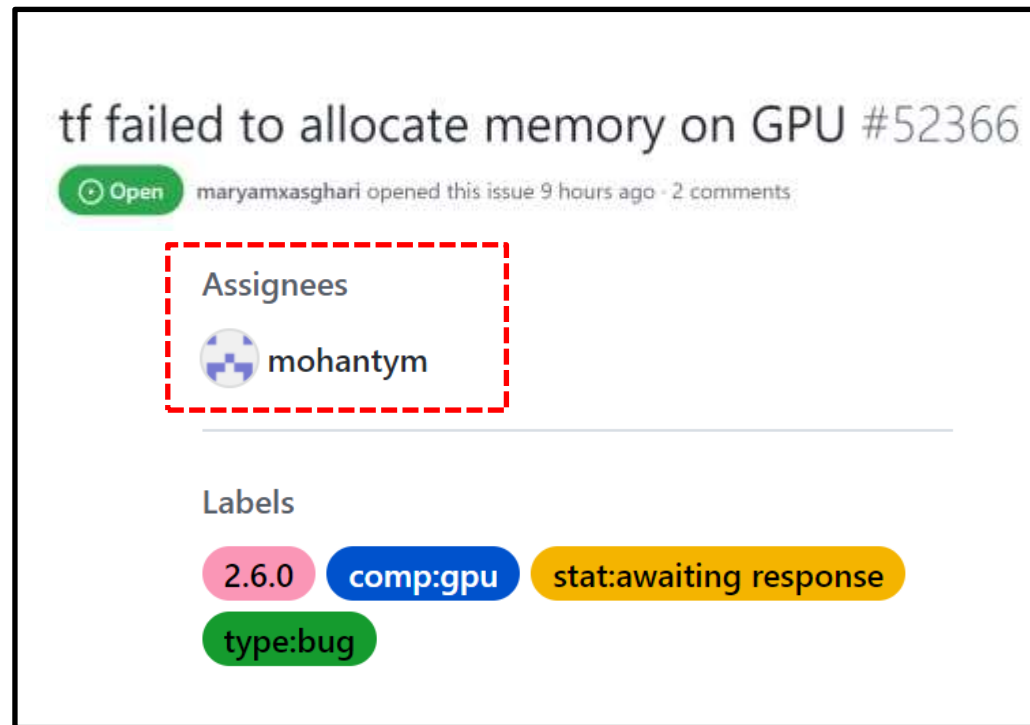
- ✓将Issue分配给相应人员加以解决

□要考虑的因素

- ✓群体基于兴趣和特长来认领相关任务
- ✓结合开发者的开发技术和经验
- ✓考虑开发人员的能力和精力

□Issue指派并非强制性

- ✓具有**通知和推荐**的性质
- ✓接受到任务指派的人员有权决定是否接受该指派，可**自愿参加或拒绝指派**



示例：指派Issue的负责人

开源项目 > 鸿蒙开源项目 && 其他开源 > 操作系统

OpenHarmony / kernel_liteos_a

Watch

281

Star

1.8K

代码

Issues 65

Pull Requests 22

Wiki

统计

DevOps

服务

Issues / 详情

3516DV300开关机压测出现系统挂死

已确认

#I41GQA

缺陷

Ieland11

创建于 2021-07-20 09:58

**【模块名_概率】简要描述：3516开发板反复开关机出现系统挂死

【环境信息】：

- 网络环境：无网络
- 硬件开发板型号：3516DV300

状态

已确认

负责人

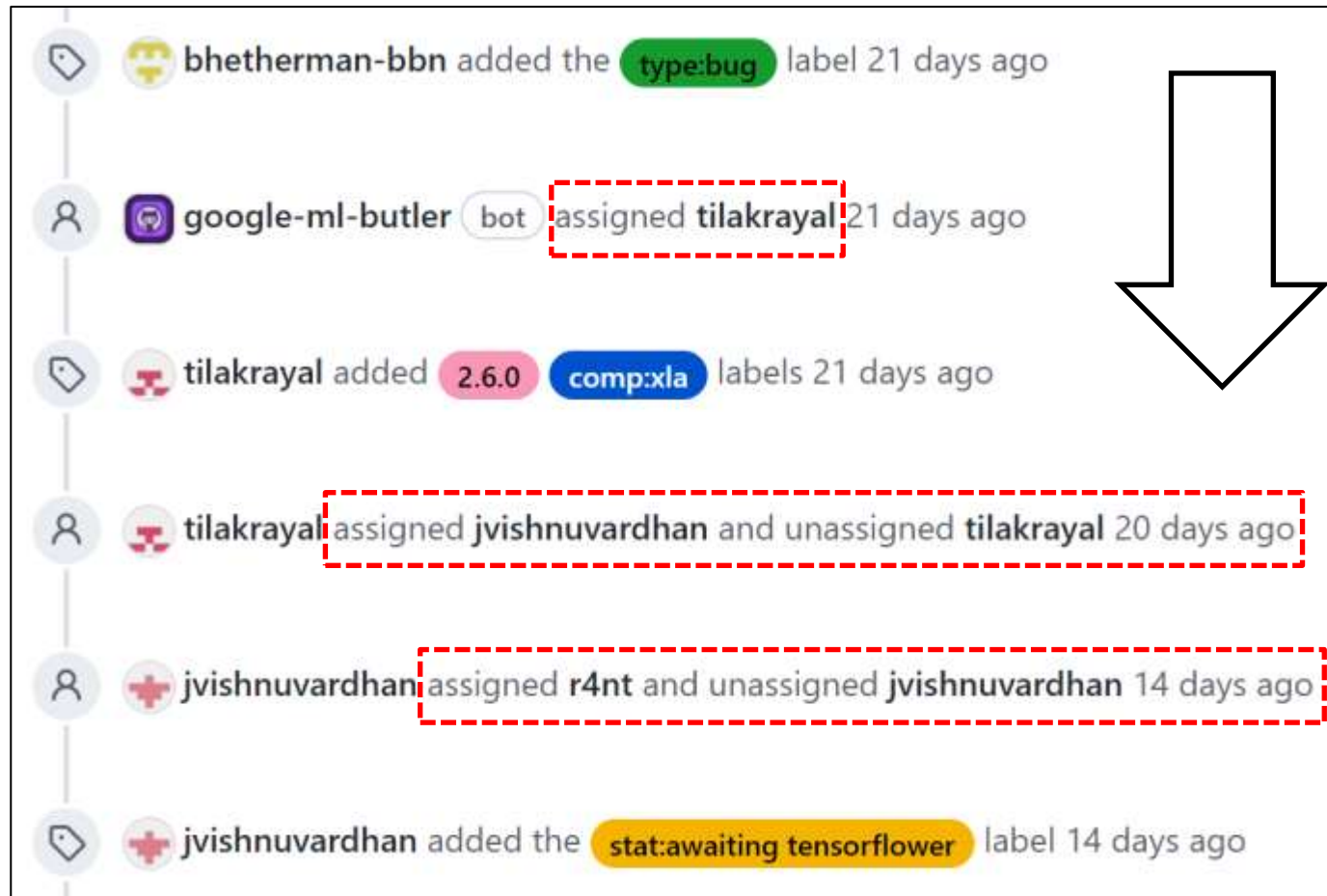
zhangg05283106

标签

Integration_Test

□跟踪和记录Issue的解决过程，掌握Issue解决状况

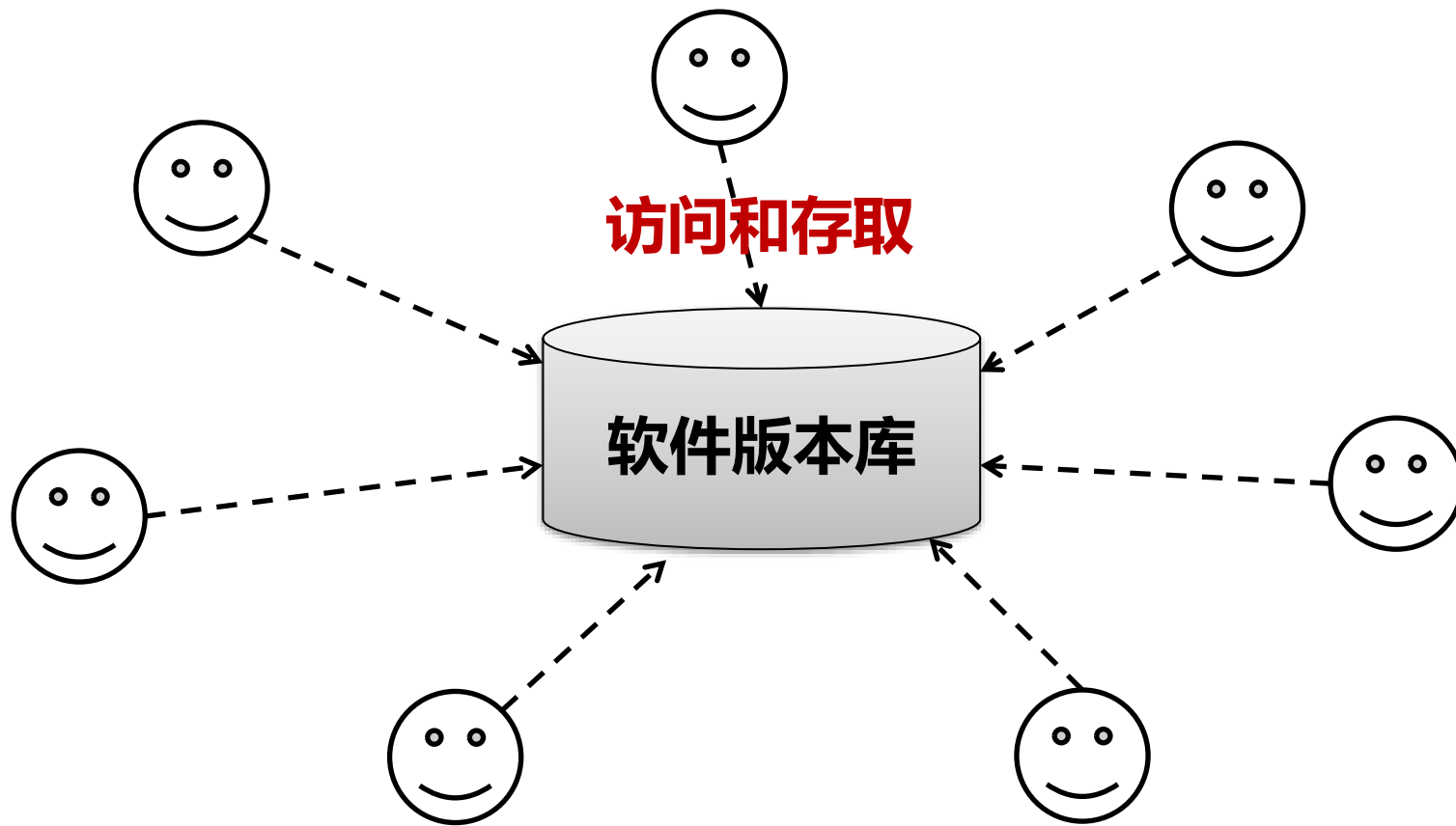
- ✓事件的发起者
- ✓发生的时间点
- ✓事件的内容等



Issue history timeline:

- bhetherman-bbn added the **type:bug** label 21 days ago
- google-ml-butler bot assigned **tilakrayal** 21 days ago
- tilakrayal added **2.6.0** **comp:xla** labels 21 days ago
- tilakrayal assigned **jvishnuvardhan** and unassigned **tilakrayal** 20 days ago
- jvishnuvardhan assigned **r4nt** and unassigned **jvishnuvardhan** 14 days ago
- jvishnuvardhan added the **stat:awaiting tensorflow** label 14 days ago

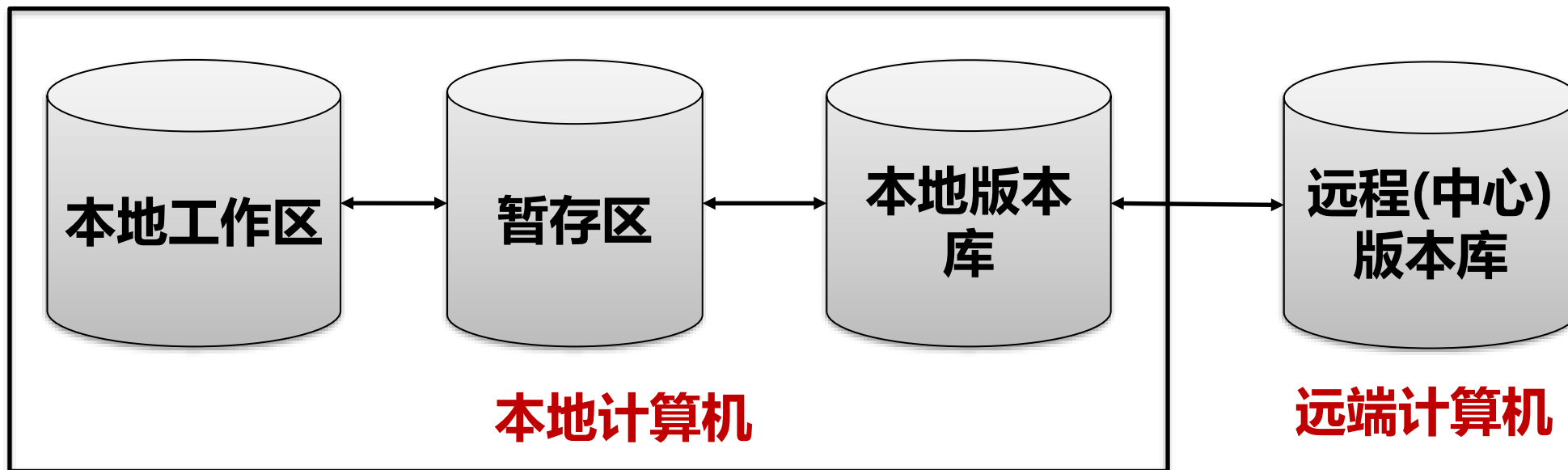
传统软件开发的集中式版本管理



多人同时需要该代码时会存在什么样的问题？



3.3 分布式版本管理思想



采用分级处理方式来区分软件项目文件的不同状态

- ✓ **工作区、暂存区、本地版本库**建在开发人员自己的计算机之中，**远程版本库**建在远程的中心服务器上
- ✓ 本地工作完成后，可把本地版本库数据（如代码）**同步推送**到远程版本库中，也可从远程版本库**拉取**（Pull）最新代码到本地版本库

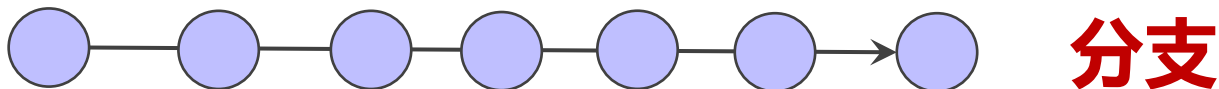
创建开发分支

□分支相当于一串开发线路，是一串代码提交历史

- ✓ 软件开发者可利用分支实现不同开发任务的并行执行与变更合并

□Git版本库中有二个常见的分支

- ✓ **master分支**：软件项目**主分支**，负责存储对外发布的项目版本，软件开发者应该确保master分支的稳定性，一般不轻易直接修改master分支中的代码，通常只有一个
- ✓ **develop分支**，软件项目的**开发分支**，通常是各分支代码的汇总分支，始终保持最新完成功能以及修复缺陷后的代码
- ✓ **开发者分支**，软件开发者自己的独立分支





□ 创建 “master” 分支

- ✓ “master” 分支仓库处于稳定、可运行和可部署的状态
- ✓ 只有管理人员才有权限把 “develop” 分支代码合并到 “master” 分支中

□ 创建 “develop” 分支

- ✓ 属于开发人员的分支仓库，只有经过审查和测试的任务分支代码才可合并到 “develop” 分支仓库中
- ✓ 只有管理人员才有权限把任务分支合并到 “develop” 分支上

□ 创建开发人员自己的任务分支

- ✓ 每个开发人员可根据自身的开发任务创建自己的任务分支

□ 代码合并 (Merge)

- ✓ 开发人员可通过项目管理人员，将自己编写代码同步合并到 “develop” 分支中，解决冲突



3.4 基于Pull/Request (P/R) 的分布式协同开发

- 每个开发人员在本地完成编程工作后，不是直接向中心仓库推送代码，而是通过发送一个**P/R合并请求**，将原始代码库的克隆库推荐合并到中心仓库之中
 - ✓ 一个“P/R”顾名思义是指一次**合并请求**，合并的内容对应于一个**代码修改补丁**，它包含了软件开发人员对软件的一次代码修改
- 接收到合并请求后，软件项目管理团队和开发人员群体需要对**P/R进行审查**
 - ✓ 评估该P/R所贡献代码的质量，将符合质量要求的代码集成到中心代码库合适的分支中

分布式协同开发的操作 (1/2)

□克隆 (Clone) / 派生 (Fork)

- ✓通过**克隆或派生**,将软件项目的仓库复制到自己的个人空间中,从而获取软件仓库中的软件代码

□本地修改

- ✓开发人员基于克隆后的软件仓库,在**本地进行软件开发活动**,如修复缺陷、开发新的功能等,所产生的代码变更将只影响其本地的克隆库,而不会影响原始的软件仓库

□提交合并 (P/R)

- ✓开发人员可将**变更的代码**以P/R的形式发送到原始的软件仓库,提供关于代码合并的相关信息,包括合并的概要性标题、合并内容,说明该PR完成了哪些工作、代码测试结果等情况

分布式协同开发的操作 (2/2)

□代码质量保证

- ✓平台采用**人工审查和自动测试**相结合的方式，检查P/R代码质量
- ✓开发人员可自发参与贡献的审查过程，以评论方式发表意见
- ✓一些软件项目还会通过持续集成工具，自动编译并测试所收到的P/R代码，并向开发人员反馈测试结果
- ✓PR的提出者接收到反馈后，可根据评论意见和测试结果，更新原始 P/R中的相关代码

□合并决策

- ✓软件项目核心开发团队综合评估上述所有因素，决定接受还是拒绝某项贡献的合并请求。如果PR被接受，则开发人员所贡献的代码和提交的历史都将被合并到项目的中心仓库中

基于P/R分布式协同开发技术的优势

□简单

- ✓使用门槛低，开发人员可方便地贡献代码、评论他人贡献，极大提高了开发人员参与软件项目开发的积极性

□规范

- ✓PR机制提供了规范化的协同开发流程，促进互联网上大众群体围绕代码贡献的交流与合作，并与大众评论、软件测试、代码审查等环节结合在一起，确保了软件开发质量

□透明

- ✓所有软件开发历史信息 and 社交活动信息都会保留下来，在开发人员主页或软件项目主页中展现



示例：mysql项目基于P/R的分布式开发协同

P/R标题

feat: pool adds gracefulExit to end gracefully #1810

Open

HQidea wants to merge 6 commits into mysqljs:master from HQidea:graceful-exit

合并6次commit
到Master分支中

开发者对P/R的评论

HQidea commented on 28 Aug 2017

As we discussed at #1803, this pull request adds an option to pool's creation,

```
var pool = common.createPool({
  // ...
  gracefulExit: true
});
```

If gracefulExit is true, every pool.getConnection or pool.query called before pool.end will success. If false, only commands / queries already in progress will complete, others will throw an error.

feat: pool adds gracefulExit to end gracefully

5df4925

dougwilson requested changes on 31 Aug 2017

dougwilson left a comment

Member

A couple comments and looks like the coverage check failed because there are uncovered cose paths introduced in this change. If you need help coming up with a test at all, let me know and I can lend a hand :)

.gitignore Outdated

Show resolved

Readme.md Outdated

Show resolved

dougwilson requested changes on 31 Aug 2017

View changes

Assignees

No one assigned

Labels

feature

Milestone

No milestone

Linked issues

Successfully merging this pull request may close these issues.

None yet

Notifications

Customize

Subscribe

You're not receiving notifications from this thread.

5 participants





3.5 基于群体的知识分享

□软件开发实践中遇到问题怎么办？

- ✓群体采用提问、回答、评论等多种方式进行在线知识分享

□基于群体的知识分享

- ✓参与分享的对象是开放的互联网大众
- ✓分享的内容表现为从问题到解答等多种知识样式

群智知识分享的分布式协同

□提问 (**Ask**)

- ✓在社区中进行提问，以获得其他人员的讨论和解答

□回答 (**Answer**) 和讨论 (**Comment**)

- ✓对提问进行回答，或者对提问和回答进行评论
- ✓必要时可以附上相关的细节，如问题解决的代码片段

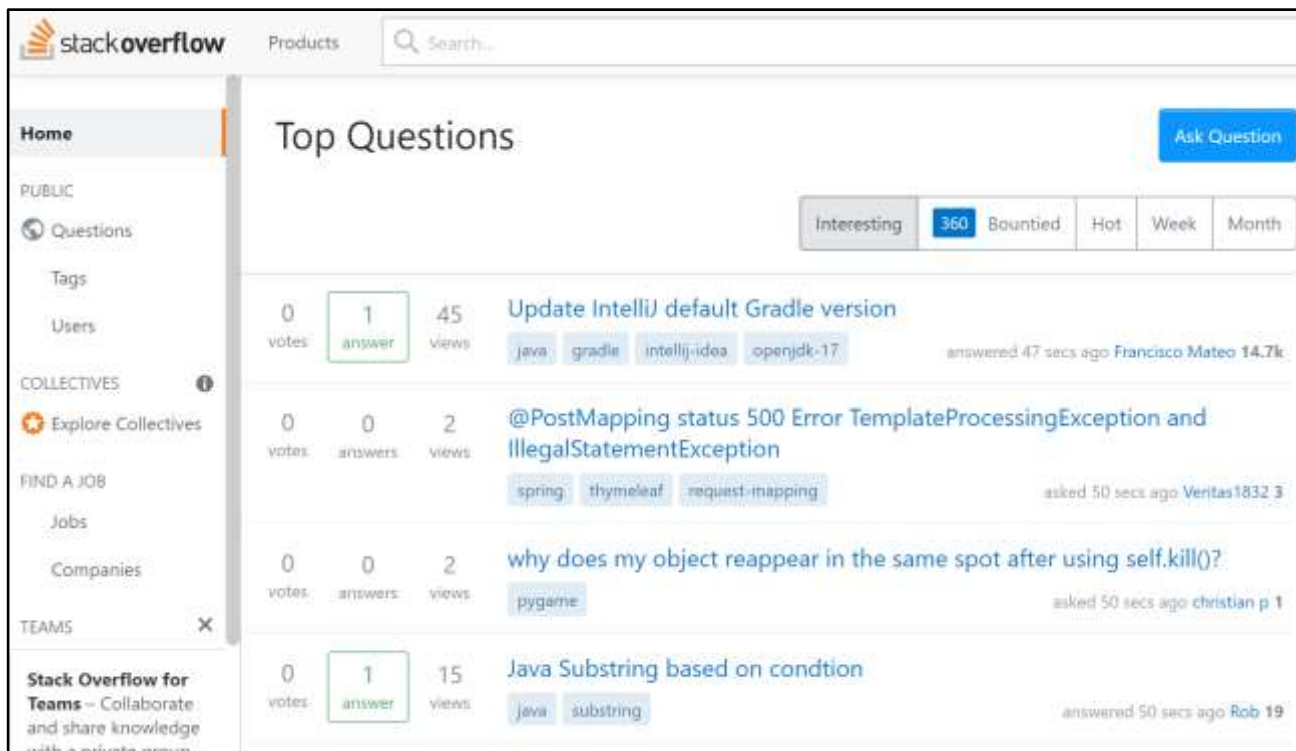
□接受 (**Accept**)

- ✓如果某个回答有效地解决了问题，那么提出者可以接受该回答

□搜索 (**Retrieve**)

- ✓检索社区是否有相同或类似的问题，并找到相应的问题解答

□ Stack overflow和CSDN



stackoverflow.com



ask.csdn.net

示例：Stack Overflow发布的问题信息

问题标题

问题信息

React too many re-renders because of data loading?

Asked today Active today Viewed 46 times

I have this chunk of code:

1

```
export default function CloseInquiryComponent(props) {  
  
  const auth = getAuth();  
  const db = getFirestore();  
  
  const { id } = useParams();  
  
  const [someVar, setSomeVar] = useState("");  
  
  // get user  
  const [user, authLoading, authErr] = useAuthState(auth);  
  // get user data of a collection
```

I get an error "too many re-renders".

each re-render calls the function and sets the variable, which causes more re-renders.

How can I set the data once it's been loaded, while also keeping the react rules in mind?

问题描述及代码

示例: Stack Overflow的问题回答

□问题回答

1 Answer

Active Oldest Votes

▲ Use `useEffect` hook.

3


```
useEffect(() => {  
  if (inquiryLoading && inquiry) setSomeVar(inquiry.SomeData);  
}, [inquiryLoading, inquiry])
```



Share Edit Follow



edited 34 mins ago

 Sinan Yaman
4,723 ● 2 ● 11 ● 31

answered 36 mins ago

 Ali Yaghoby
319 ● 2 ● 11

`inquiry` should also be in the dependency array. – Sinan Yaman 36 mins ago

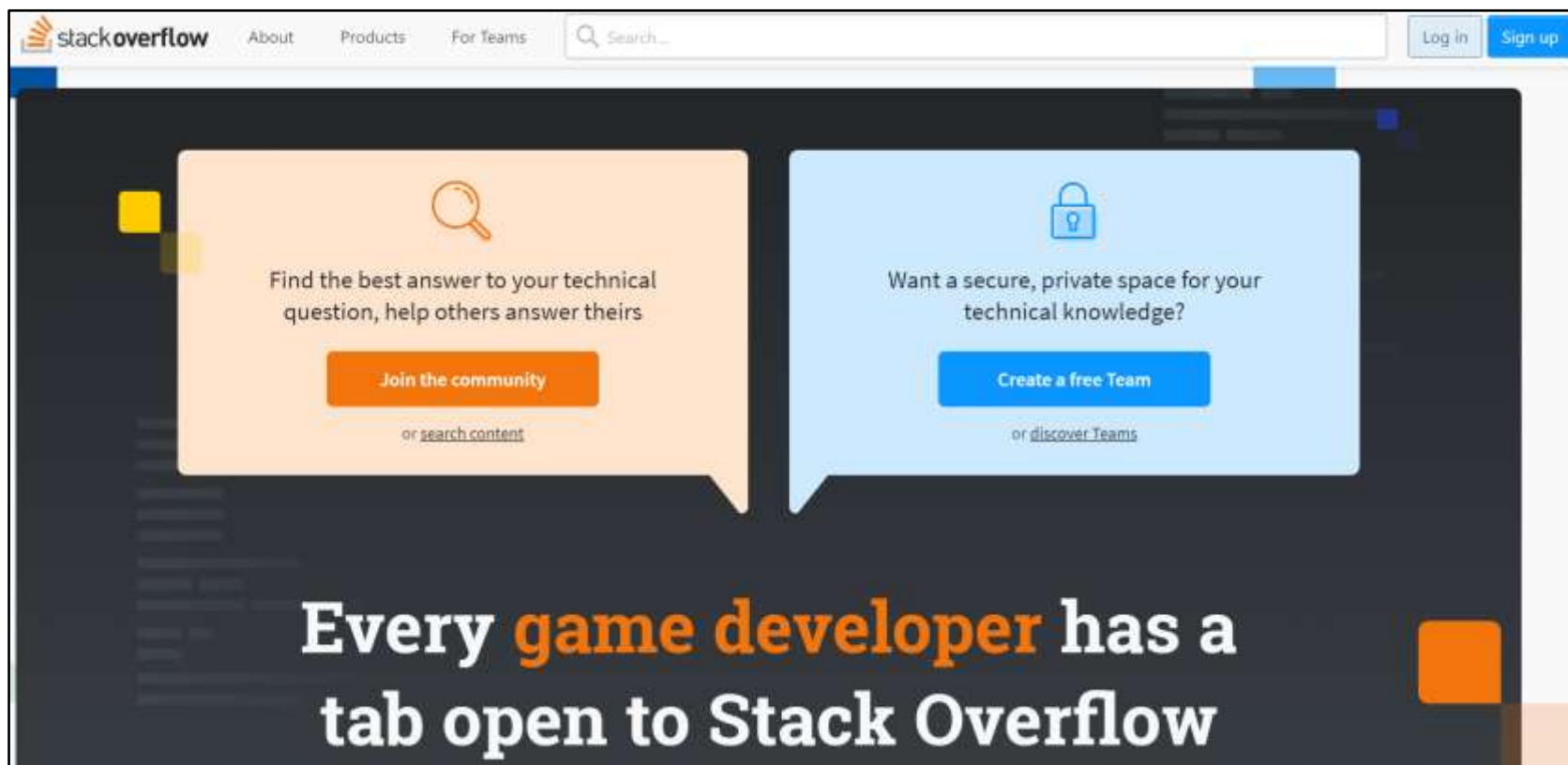
1 Thanks for your comment – Ali Yaghoby 33 mins ago

Add a comment

问题回答描述
回答相关信息

海量的软件开发知识

- 用户**1500**万以上，月访问用户量**1亿**，用户获得帮助**450亿**
- **2000**多万个问题，**3200**万回答，**8200**万评论



充分利用知识
分享社区中的
知识来解决课
程学习和实践
中遇到的问题

stackoverflow.com

总结：群体化软件开发方法

□开放和共享

- ✓吸引互联网大众自主参与，汇聚大众群体的贡献和智慧
- ✓自由分享程序代码、知识问答等内容

□组织和管理

- ✓采用基于社区的组织模式，借助去中心化的分布式协同开发模式
- ✓借助互联网平台的支持，如Github、Stack Overflow等

□多样化目的

- ✓既支持软件开发，也支持软件开发知识的分享
- ✓将软件创作和软件生产融合在一起来支持软件开发