

软件质量与评测技术

Software Quality & Evaluation Technology

计算机学院 单纯
sherryshan@bit.edu.cn
2025年11月

软件测试实践

Software Testing Practice

计算机学院 单纯

sherryshan@bit.edu.cn

2025年12月3日

7.7 软件测试的误区（1）

■ 误区一

- 如果发布出去的软件有质量问题，都是软件测试人员的错

■ 误区二

- 软件测试技术要求不高，至少比编程容易多了

■ 误区三

- 有时间就多测试一些，来不及就少测试一些

7.7 软件测试的误区（2）

■ 误区四

- 软件测试是测试人员的事，与开发人员无关

■ 误区五

- 根据软件开发瀑布模型，软件测试是开发后期的一个阶段

7.8 软件测试的原则（1）

- 所有测试的标准都是建立在用户需求之上
- 软件测试必须基于“质量第一”的思想去开展各项工作，当时间和质量冲突时，时间要服从质量
- 事先定义好产品的质量标准，只有有了质量标准，才能根据测试的结果，对产品的质量进行分析和评估

7.8 软件测试的原则（2）

- 软件项目一启动，软件测试也即开始，而不是等程序写完，才开始进行测试
- 穷举测试是不可能的。甚至一个大小适度的程序，其路径排列的数量也非常大，因此，在测试中不可能运行路径的每一种组合
- 第三方进行测试会更客观，更有效
- 软件测试计划是做好软件测试工作的前提

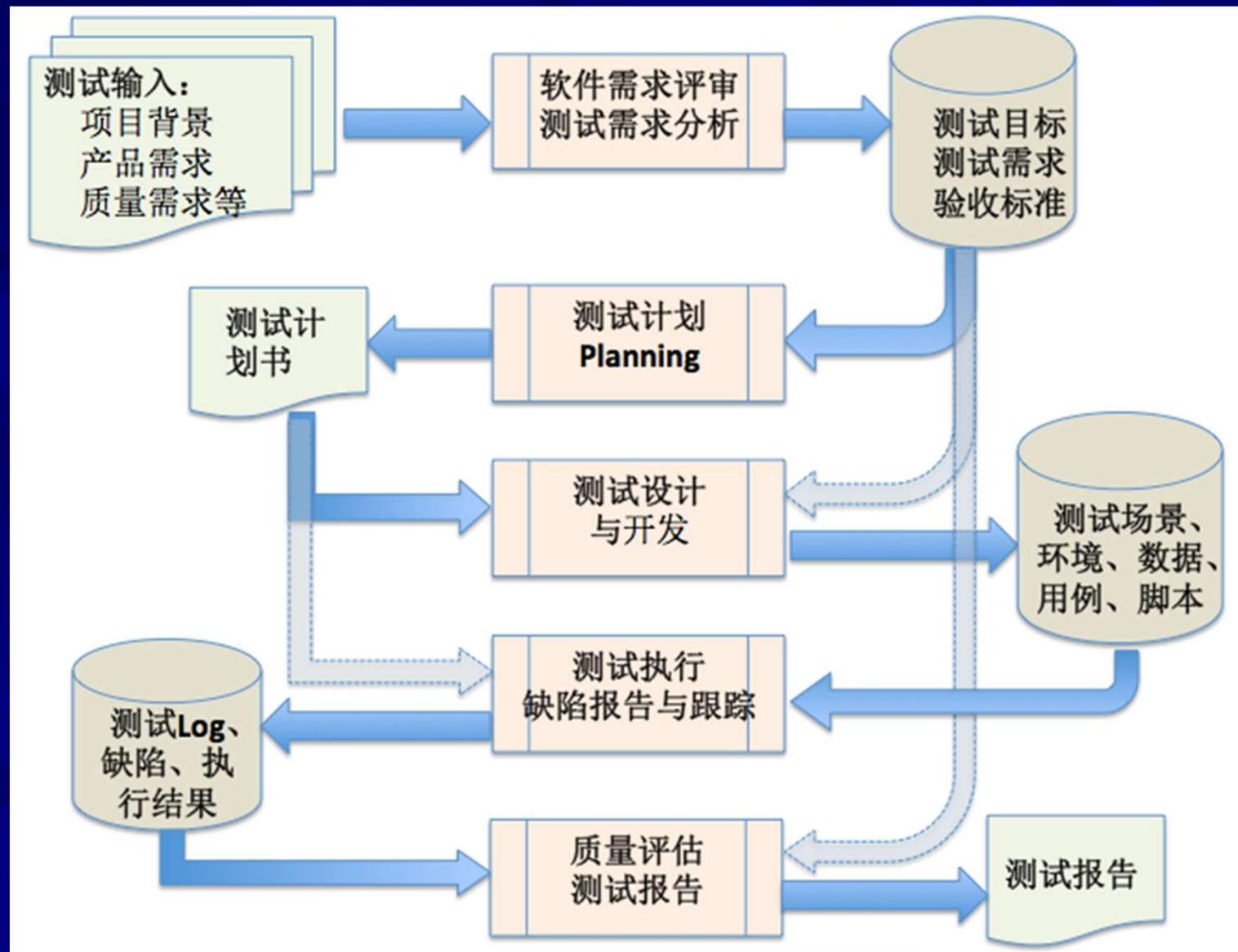
7.8 软件测试的原则（3）

- 测试用例是设计出来的，不是写出来的，所以要根据测试的目的，采用相应的方法去设计测试用例，从而提高测试的效率，更多地发现错误，提高程序的可靠性
- 对发现错误较多的程序段，应进行更深入的测试。一般来说，一段程序中已发现的错误数越多，其中存在错误的概率也就越大
- 重视文档，妥善保存一切测试过程文档（测试计划、测试用例、测试报告等）

课堂练习

■ 课堂练习
— 课堂练习2

软件测试工作和测试件



参考资料 (1)

- 第10~13章，朱少民 主编，软件测试方法和技术（第4版），北京：清华大学出版社，2022
- Adbook项目开发文档，单纯等，Adbook 北京项目组，2008~2009
- 张旸旸 于秀明主编，软件评测师教程（第2版），北京：清华大学出版社，2021

参考资料 (2)

- [美] Rex Black著, 天宏工作室译, 测试流程管理, 北京: 北京大学出版社, 2001
- William E. Lewis, Software Testing and Continuous Quality Improvement, Gunasekaran Veerapillai, Technical Contributor, Second Edition, Auerbach Publications, A CRC Press Company, 2005

参考资料 (3)

- 第2章，宫云战 主编，软件测试教程（第3版），北京：机械工业出版社，2021

软件测试标准

Software Testing Standards

计算机学院 单纯

sherryshan@bit.edu.cn

2025年12月3日

内容概览

- 1. 测试过程模型
- 2. 测试过程改进模型
- 3. 软件测试标准和规范
- 4. 软件测试管理和评判体系

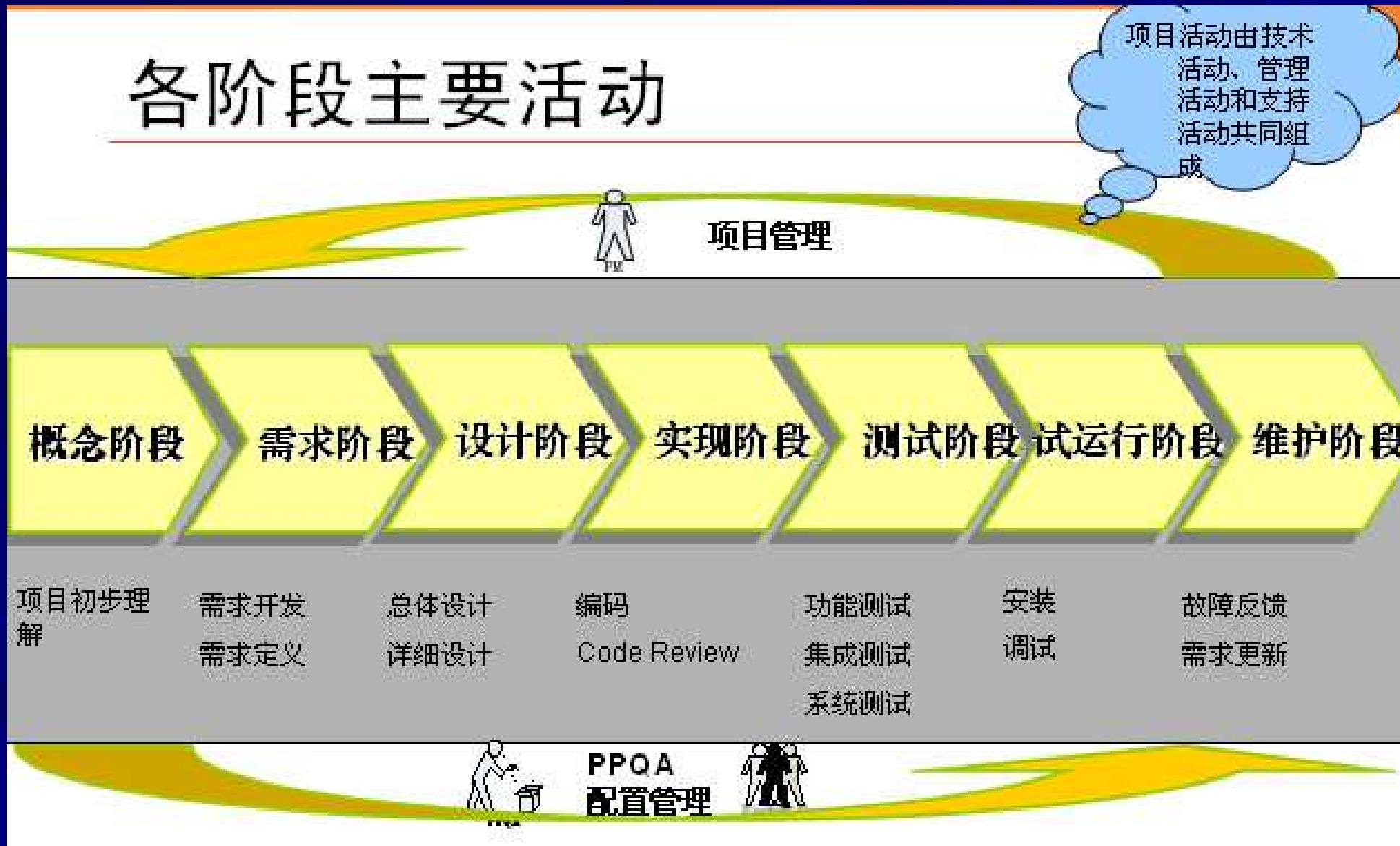
1. 测试过程模型

- 1.1 软件生命周期
- 1.2 软件开发模型
- 1.3 软件测试模型

1.1 软件生命周期（1）

- 软件生命周期是指从规划—软件产品开始到不再使用为止的一段时间间隔
- 软件生命周期一般包括
 - 概念阶段
 - 需求阶段
 - 设计阶段
 - 实现阶段
 - 测试阶段
 - 试运行阶段
 - 运行和维护阶段

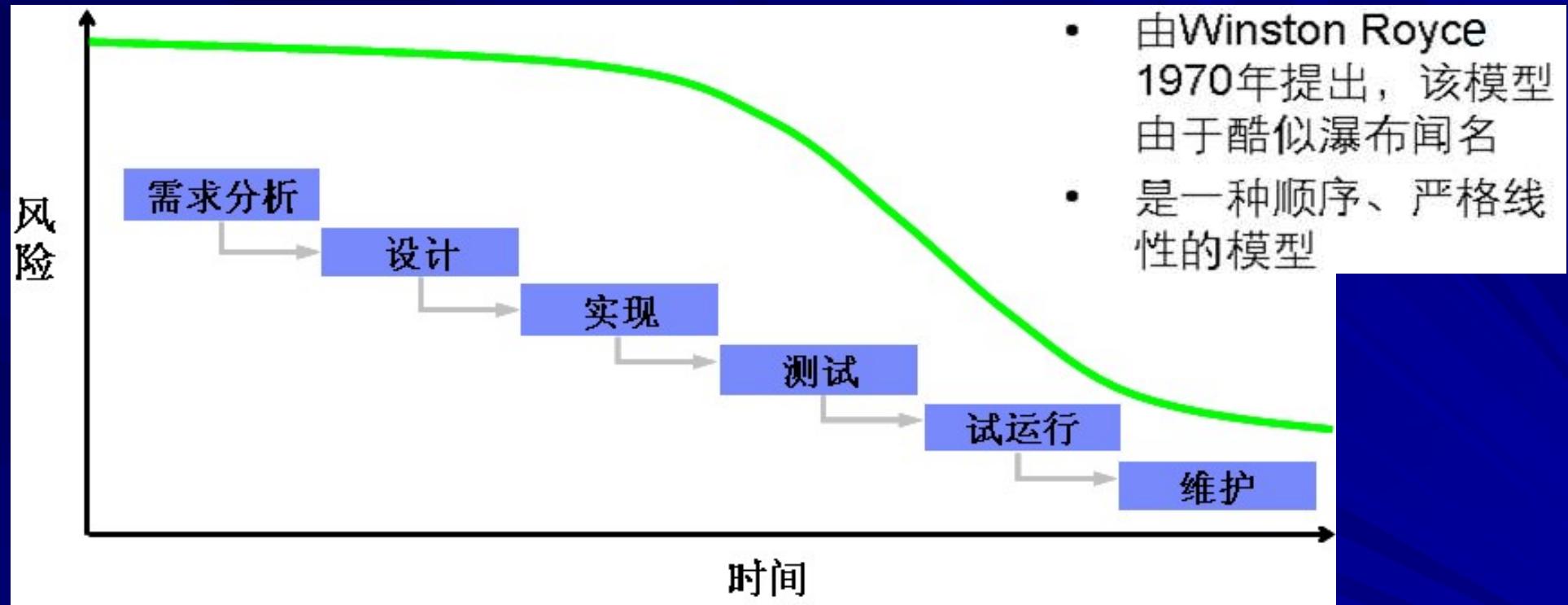
1.1 软件生命周期（2）



1.2 软件开发模型（1）

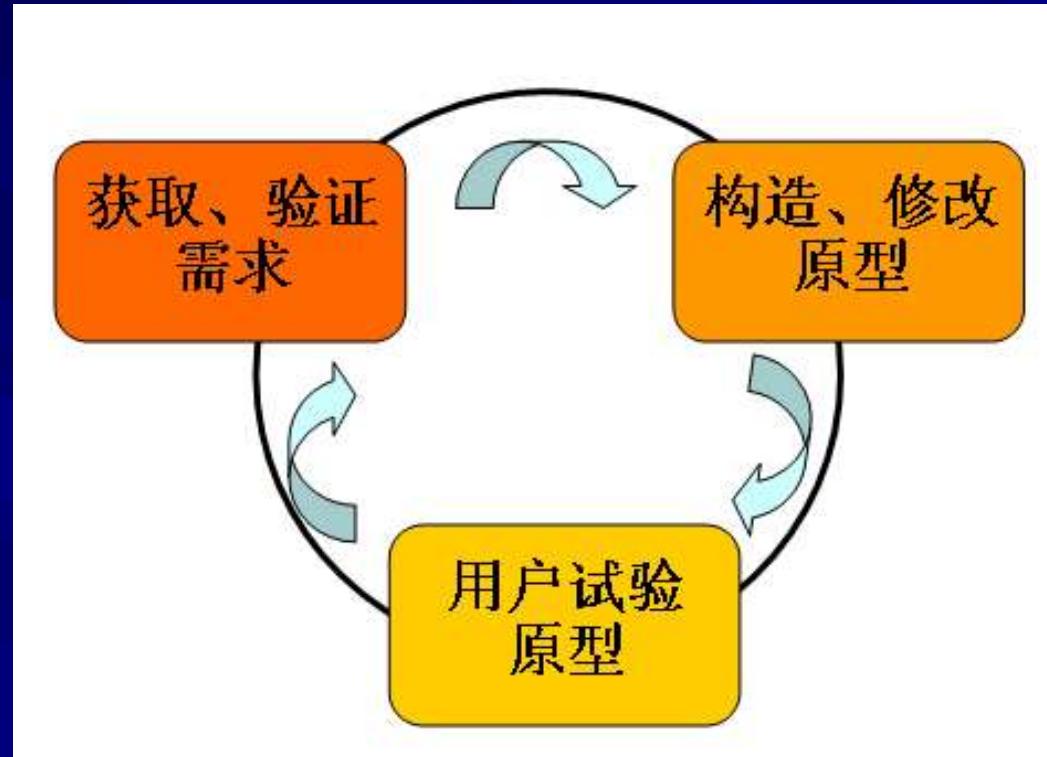
- 1.2.1 瀑布模型
- 1.2.2 原型模型
- 1.2.3 螺旋模型
- 1.2.4 增量模型
- 1.2.5 Rational统一过程（RUP）
- 1.2.6 敏捷开发

1.2.1 瀑布模型



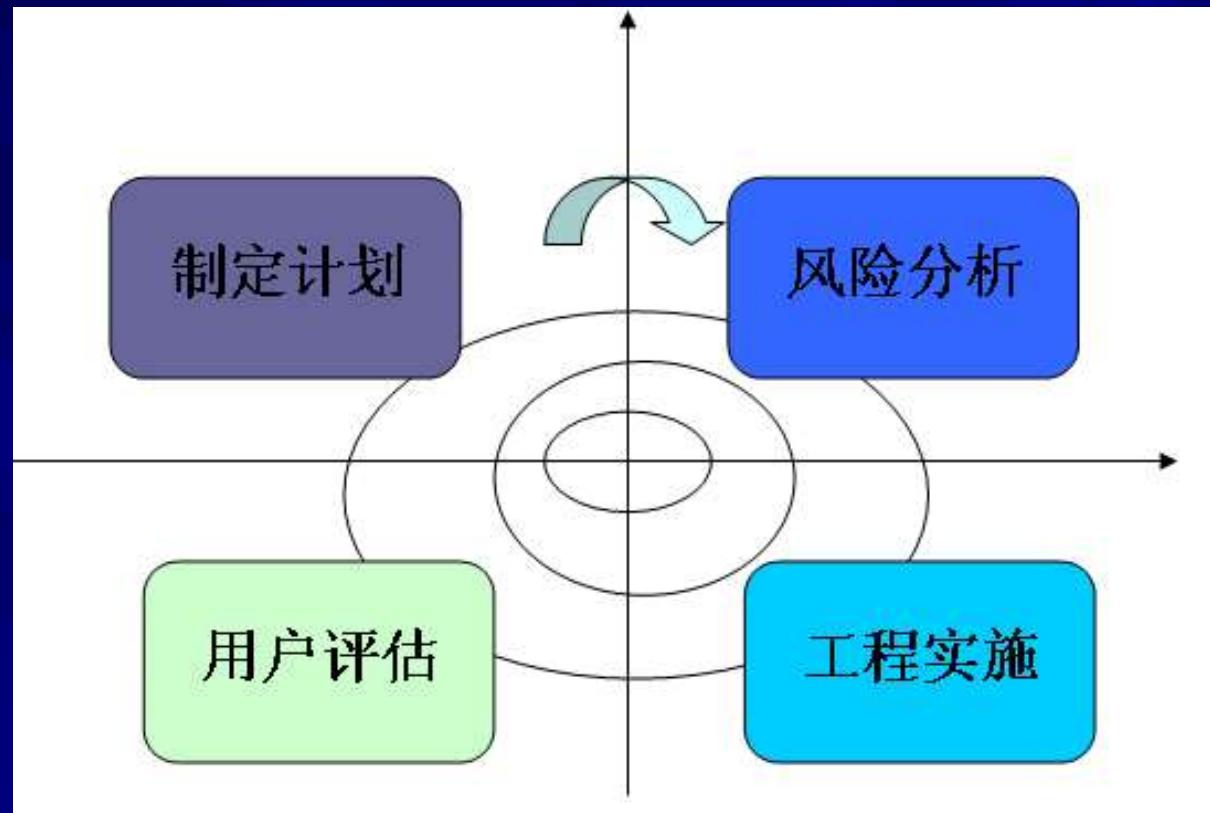
- 优点
 - 简单、线性
- 缺点
 - 理想化的模型，无法克服“变化”引发的问题；如果上一步做错，下一步就会将错就错；
 - 开发人员常常陷入“阻塞状态”，等待上一环节完成

1.2.2 原型模型



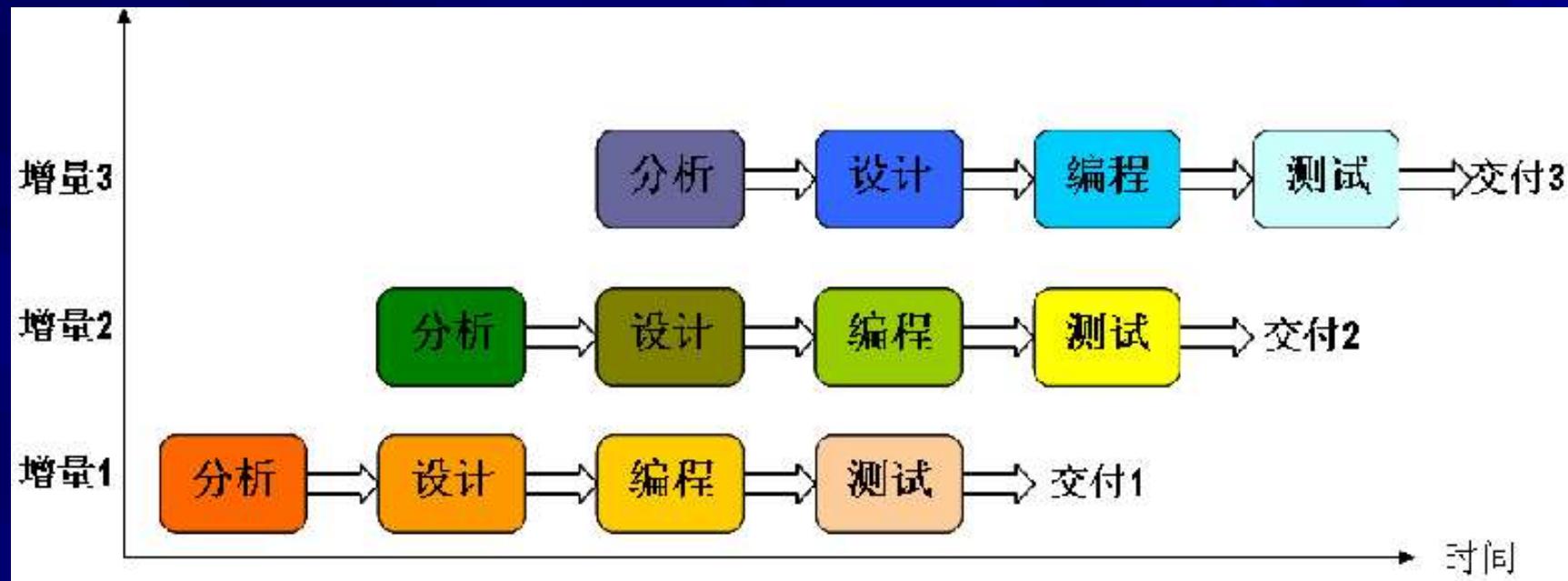
- 主要目的获取、验证需求
- 主要特色：快速
- 原型可以保留，也可以抛弃

1.2.3 螺旋模型



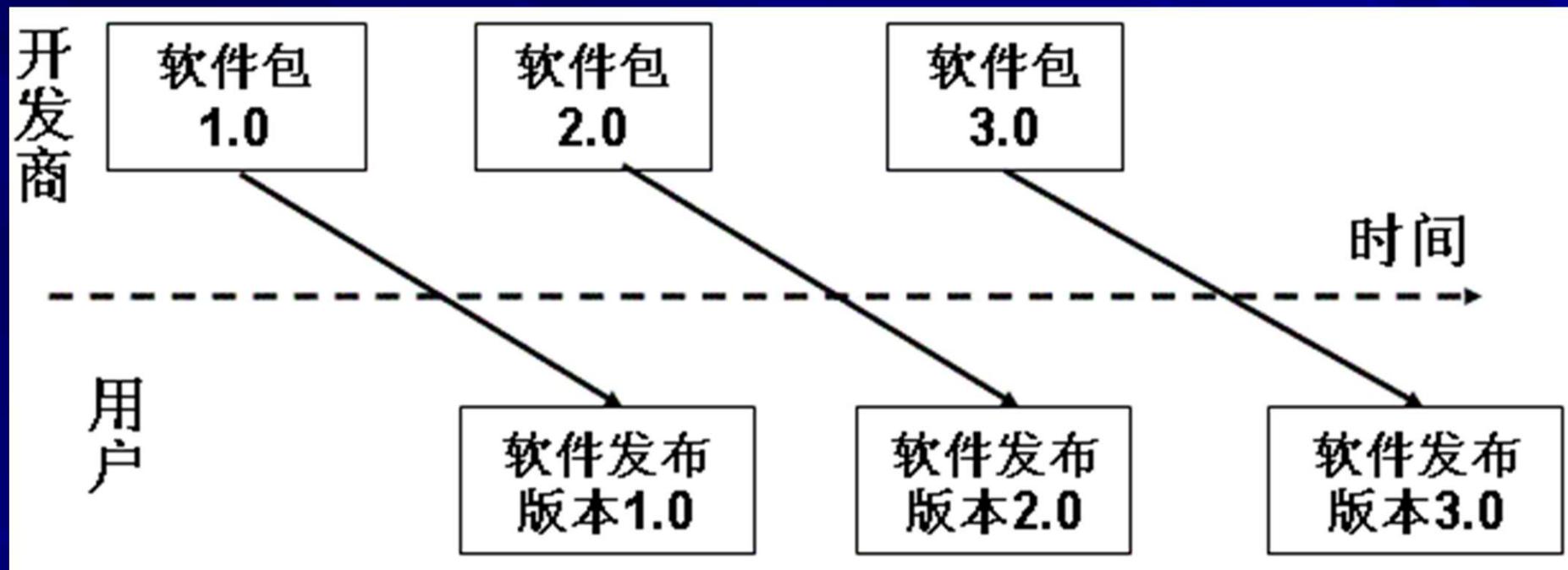
- 每旋转一圈，开发出一个完善的版本

1.2.4 增量模型



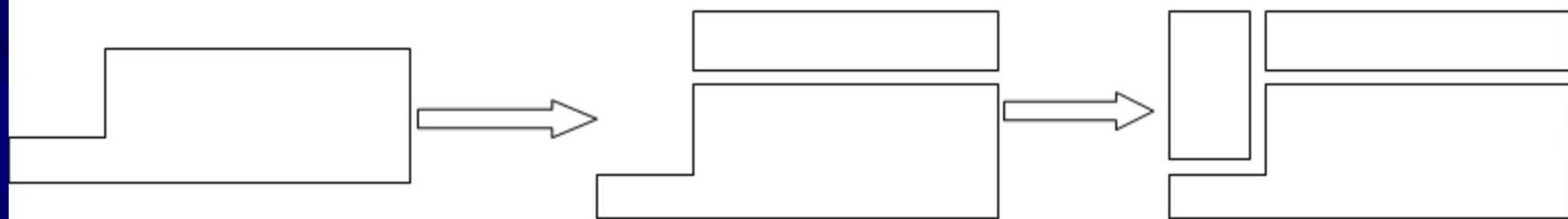
- 由若干开发序列组成，每个序列瀑布模式；分段的线性模型
- 下一增量基于上一个增量
- 每个增量有一个版本

阶段开发

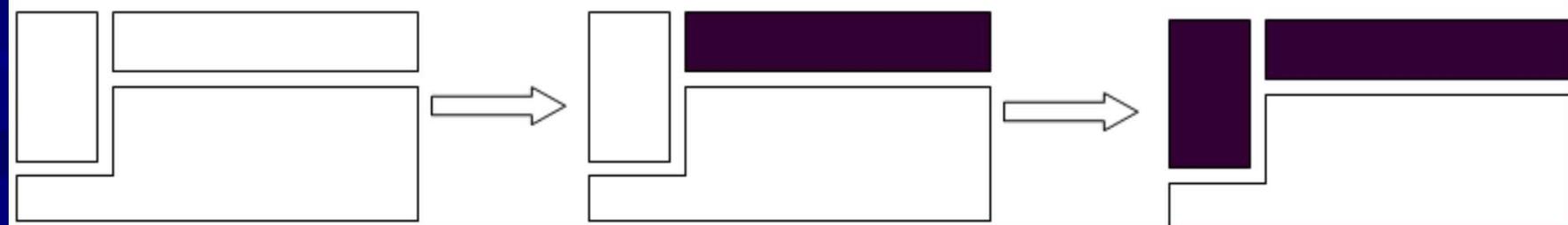


增量和迭代

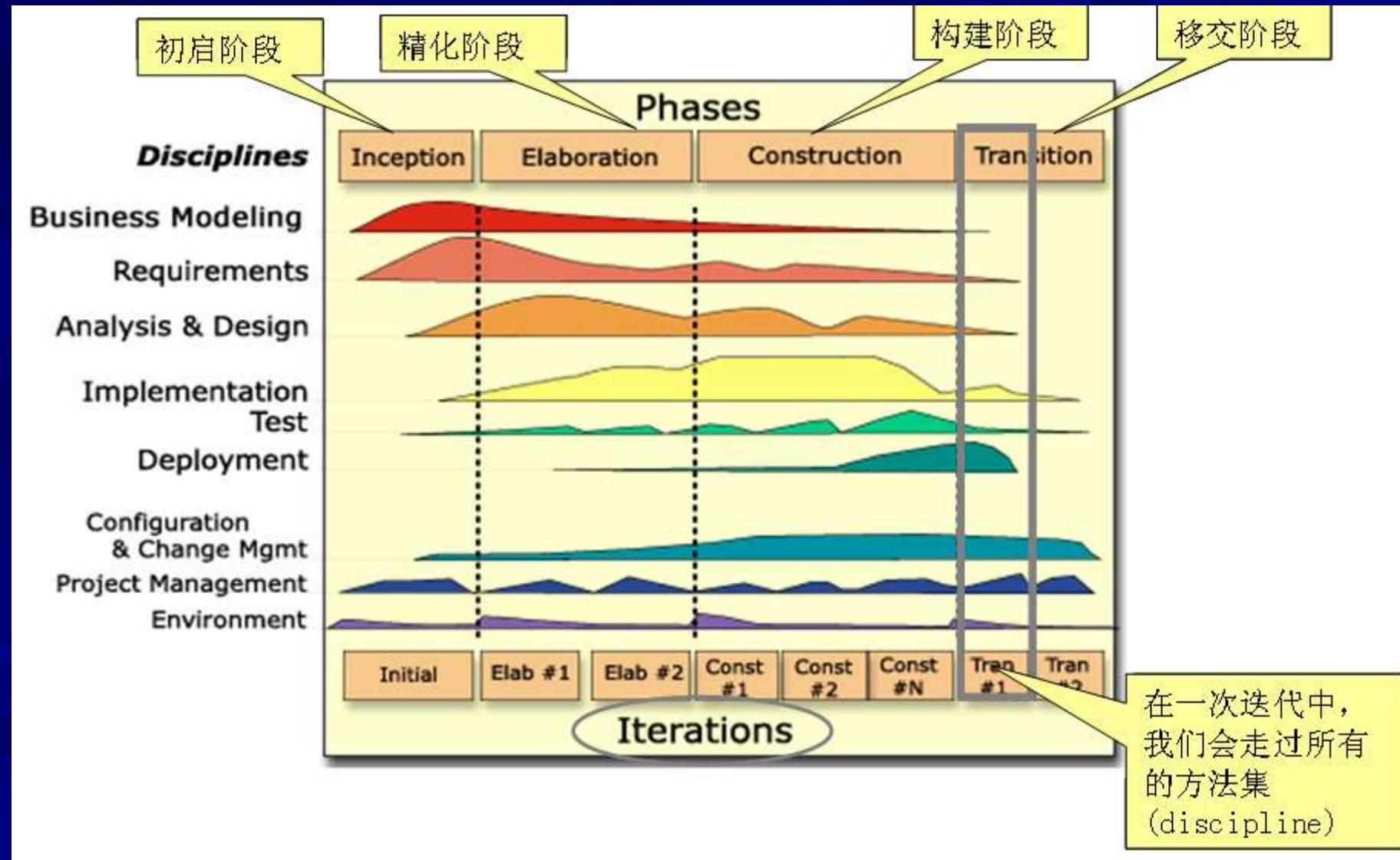
增量开发



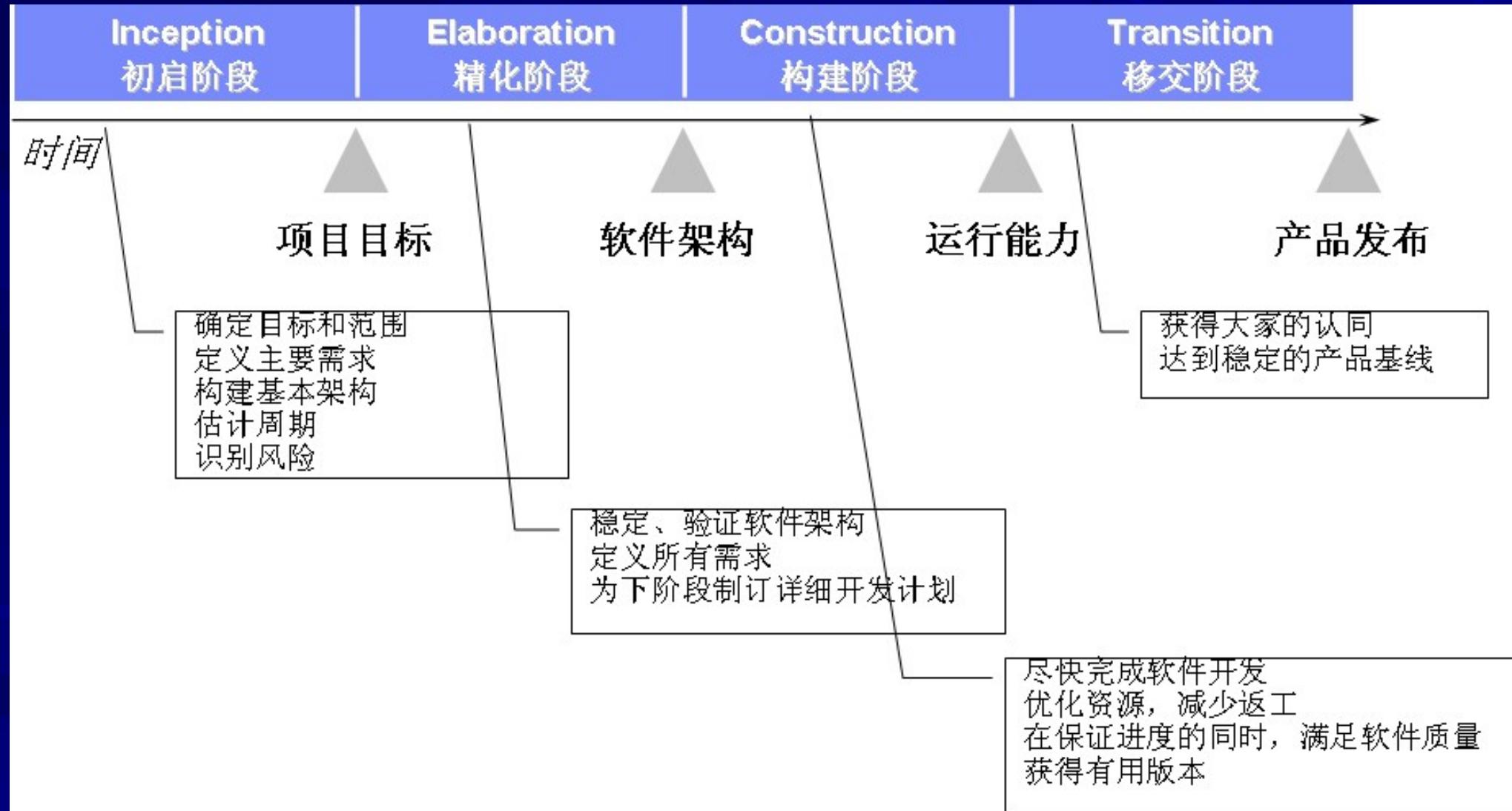
迭代开发



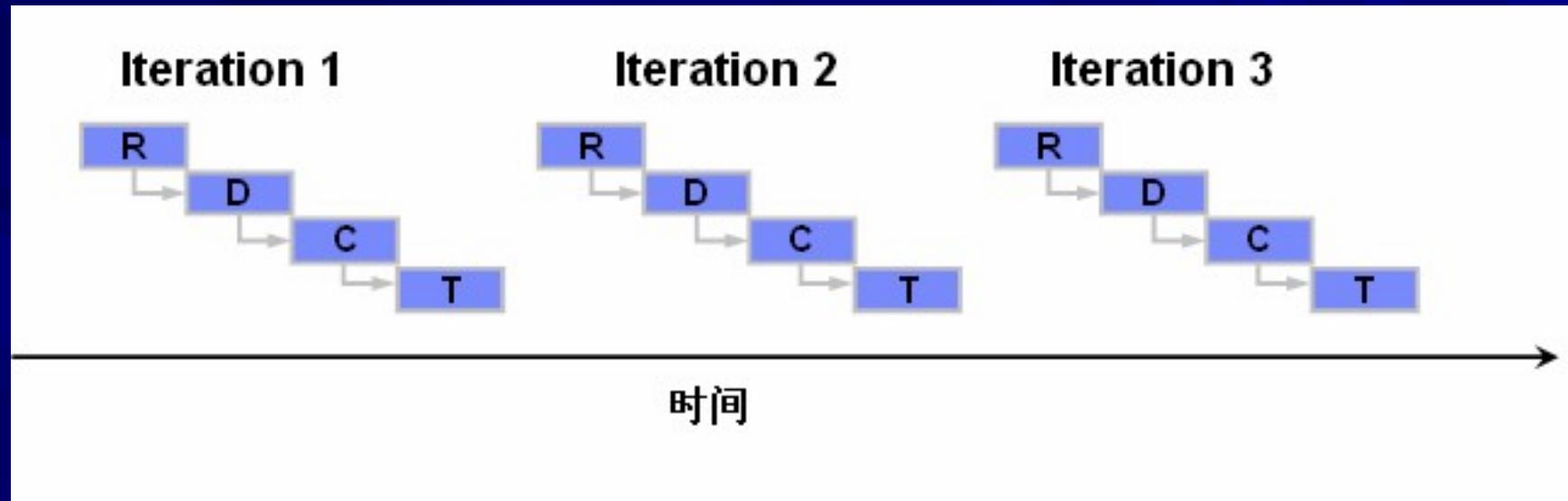
1.2.5 Rational统一过程 (1)



1.2.5 Rational统一过程 (2)

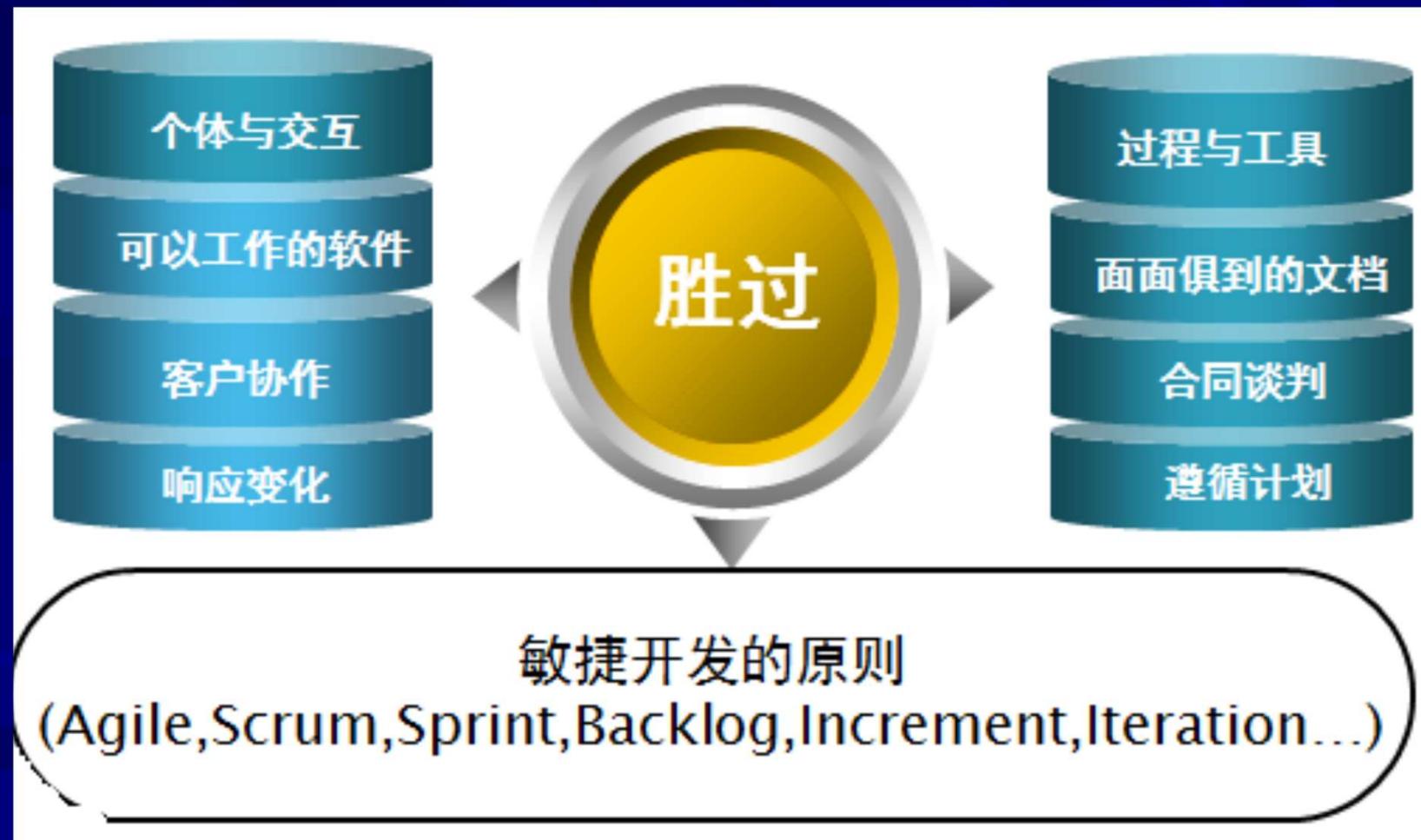


1.2.5 Rational统一过程 (3)

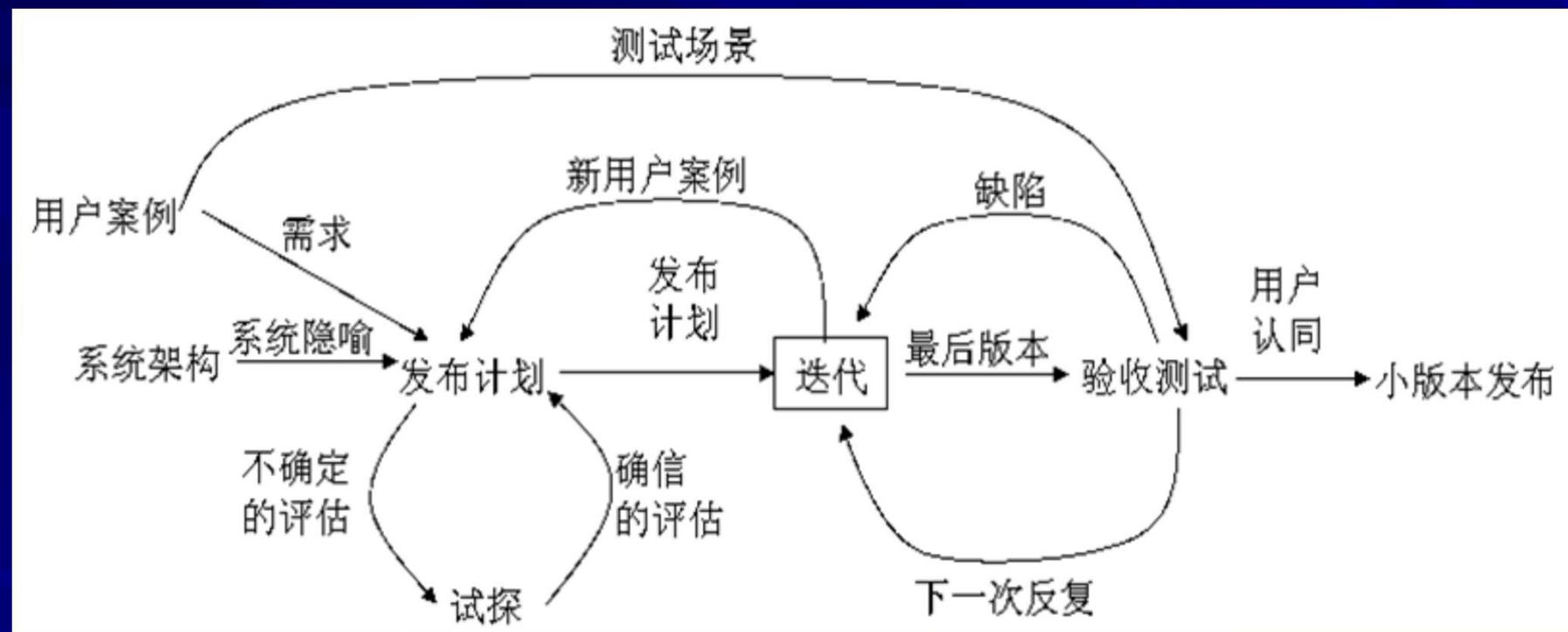


- 把复杂的问题分解成一系列相对简单的问题
- 早期的迭代解决风险最高的问题
- 每次迭代都增加系统的功能并产生一个可运行的结果
- 每次迭代都包括有测试工作

1.2.6 敏捷开发 (1)



1.2.6 敏捷开发 (2)



1.2.6 敏捷开发（3）

敏捷测试特点

- ✓ SM、开发人员等也都会参与测试。
- ✓ 提高软件质量是敏捷团队的工作，在开发过程中每个成员都有责任提交高质量的工作产品（需求、代码、设计文档...）。
- ✓ 增量测试和回归测试相结合

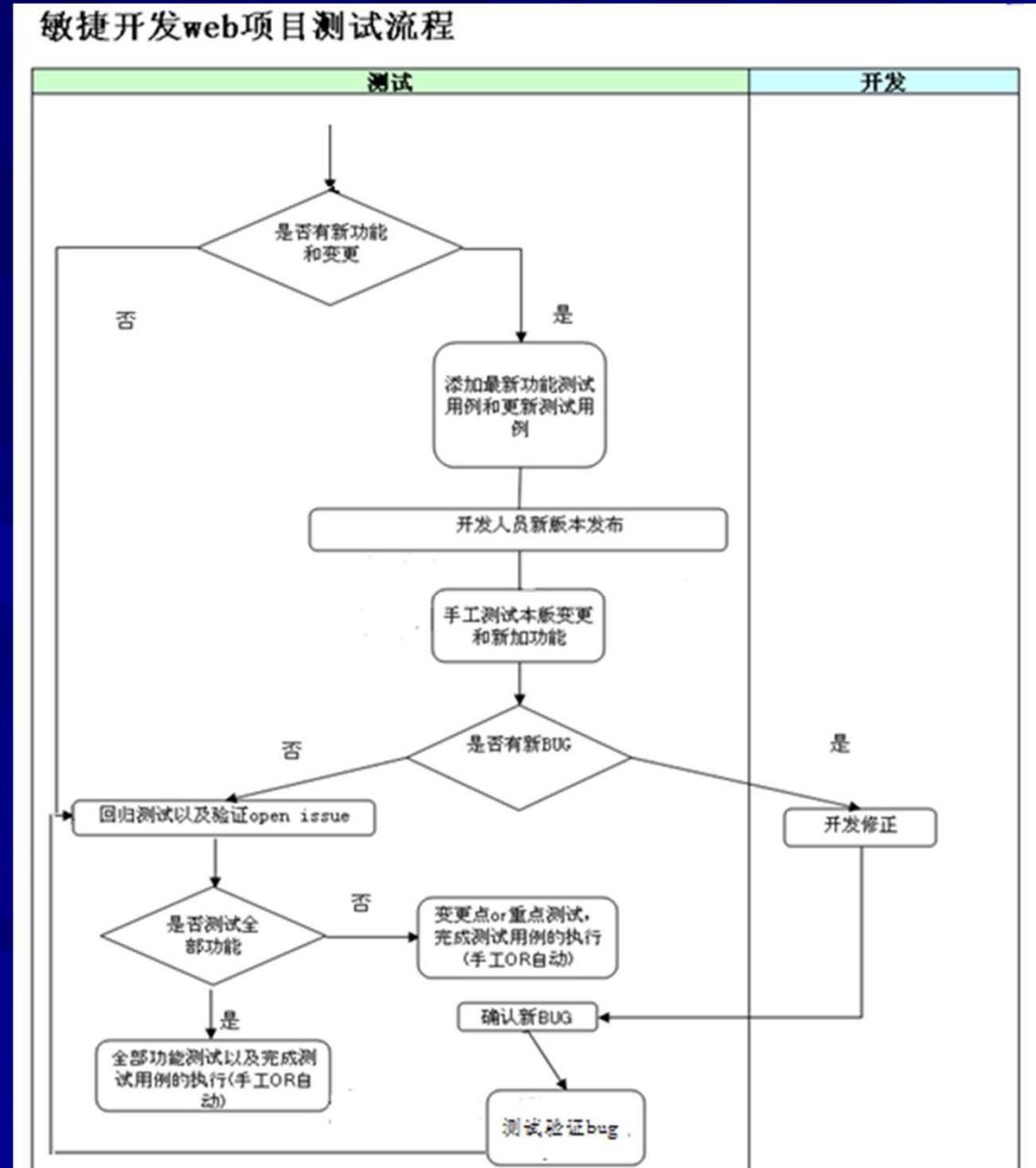
✓ 只有需求的雏形，没有详细的需求文档、设计文档。
↓

✓ 没有办法根据需求来设计测试用例
↓

✓ 解决方法：先设计测试框架，通过迭代逐步补充、完善测试用例；动态调整测试用例的粒度；使用不同的测试方法。

1.2.6 敏捷开发 (4)

■ 实际案例



1.2 软件开发模型（2）

■ 遗憾

– 在这些过程方法中，并没有充分强调测试的价值，也没有给测试足够的重视，利用这些模型无法更好地指导测试实践

■ 软件测试是与软件开发紧密相关的一系列有计划的系统性活动，软件测试也需要测试模型去指导实践

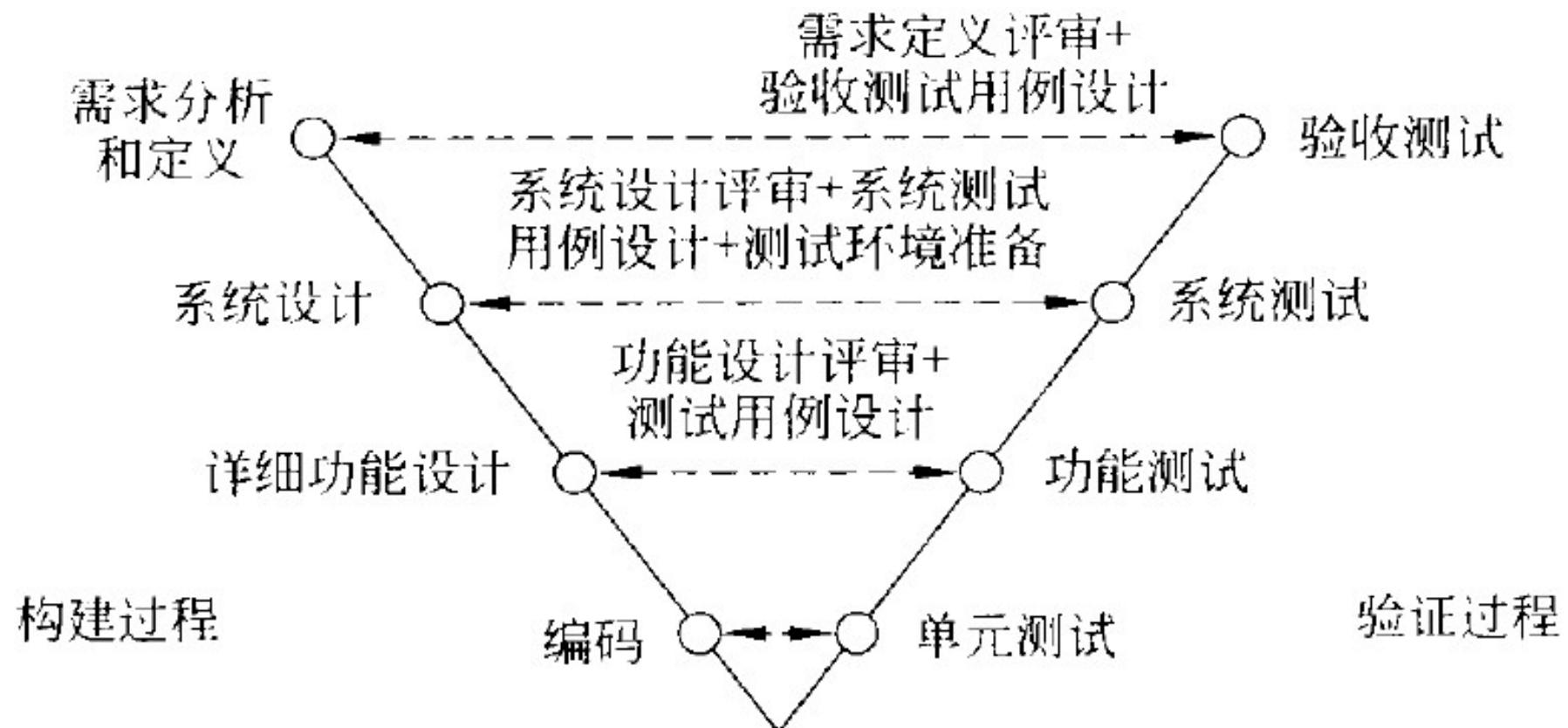
1.3 软件测试模型

- 1.3.1 V模型
- 1.3.2 W模型
- 1.3.3 H模型
- 1.3.4 X模型
- 1.3.5 测试模型的比较
- 1.3.6 测试模型的使用

1.3.1 V模型（1）

- 最具有代表意义的测试模型。最早由 Paul Rook 在 20 世纪 80 年代后期提出，V 模型在英国国家计算中心文献中发布，旨在改进软件开发的效率和效果
- V 模型是软件开发瀑布模型的变种，它反映了测试活动与分析设计的关系

1.3.1 V模型 (2)



1.3.1 V模型（3）

- 需求、功能、设计和编码的开发活动随时
间而进行，而相应的测试活动（即针对需
求、功能、设计和编码的测试）开展的次
序正好相反
- 成功应用V模型的关键因素是设计测试案例
的时机

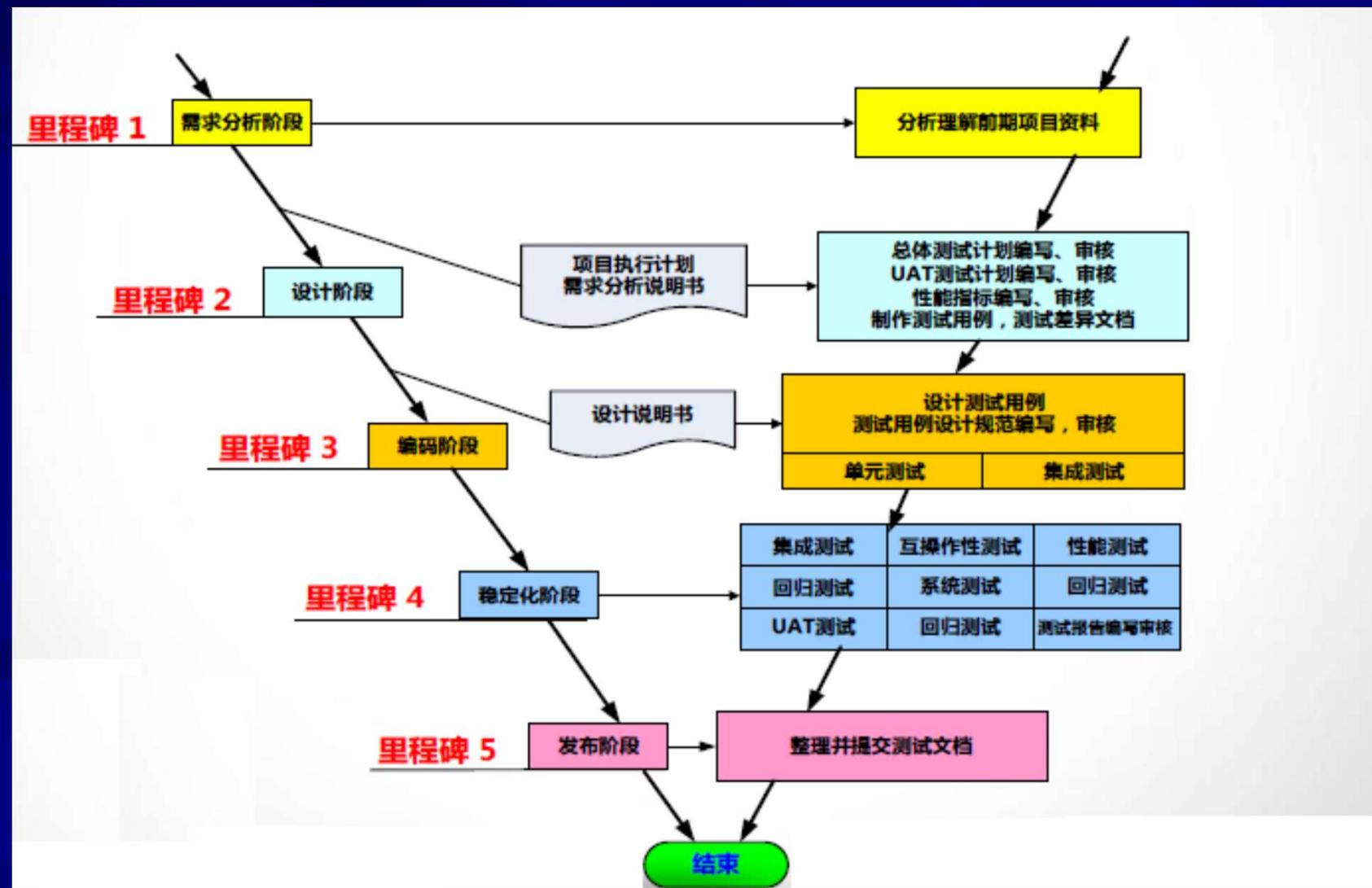
1.3.1 V模型 (4)

■ V模型的局限

- 仅仅把测试过程作为在需求分析、概要设计、详细设计及编码之后的一个阶段。容易使人理解为测试是软件开发的最后的一个阶段，主要是针对程序进行测试寻找错误，而需求分析阶段隐藏的问题一直到后期的验收测试才被发现

实际案例

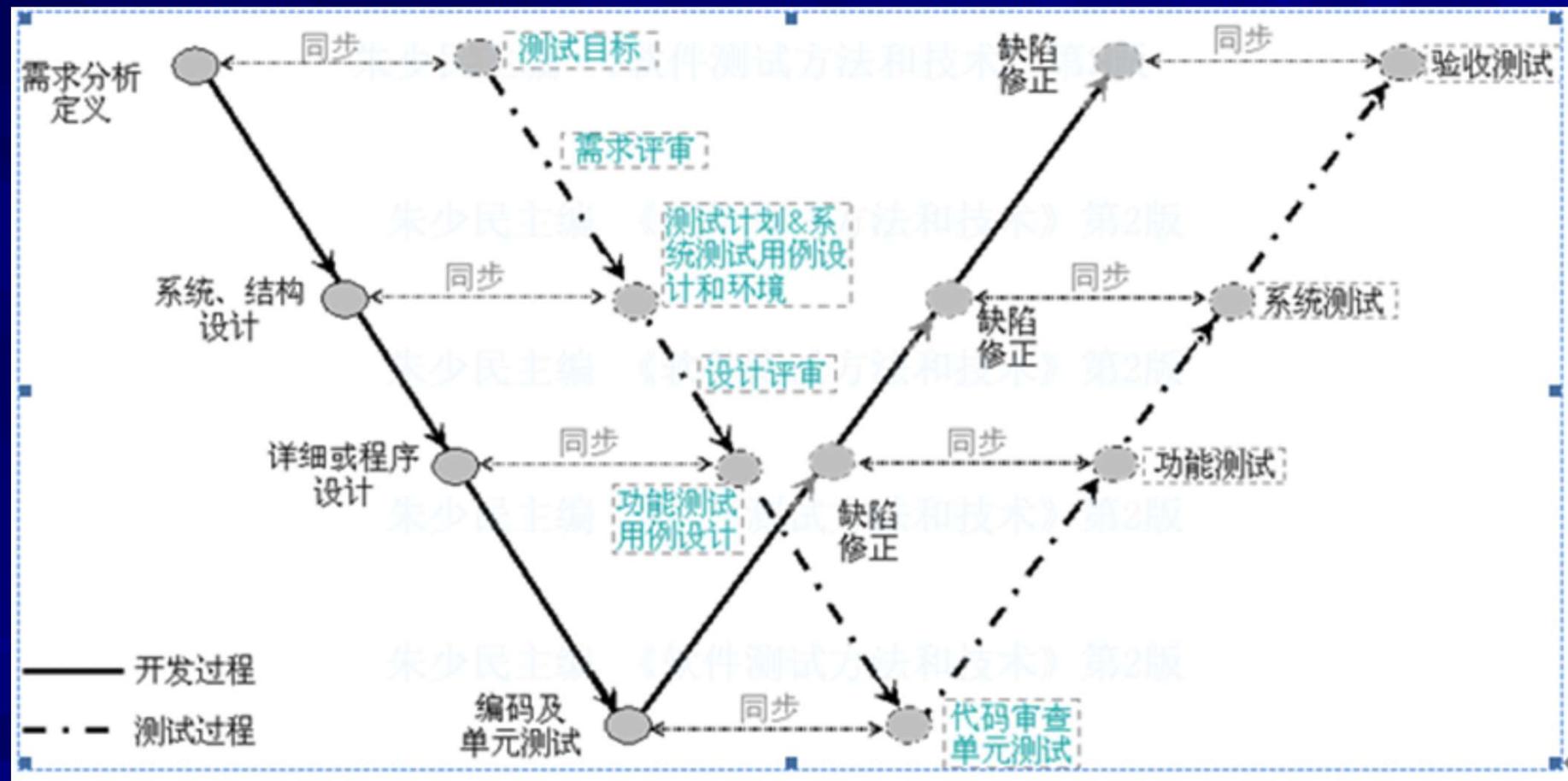
■ 某文档格式转换测试项目



1.3.2 W模型（1）

- W模型由Evolutif公司提出，相对于V模型，W模型更科学
- 在V模型中增加软件各开发阶段应同步进行的测试，被演化为一种W模型。开发是一个“V”，测试是与此并行的“V”。
- W模型基于IEEE std 1012-1998《软件验证和确认（V&V）》原理

1.3.2 W模型 (2)



1.3.2 W模型（3）

■ W模型强调：测试伴随着整个软件开发周期，而且测试的对象不仅仅是程序，需求、功能和设计同样要测试。只要相应的开发活动完成，就可以开始执行测试（例如，需求分析一完成，我们就可以对需求进行测试，而不是到最后才进行针对需求的验收测试）。测试与开发是同步进行的

1.3.2 W模型（4）

■ 局限性

— W模型和V模型都把软件的开发视为需求、设计、编码等一系列串行的活动。同样的，软件开发和测试保持一种线性的前后关系，需要有严格的指令表示上一阶段完全结束，才可正式开始下一阶段。这样就无法支持迭代、自发性以及变更调整。对于当前很多文档需要事后补充，或者根本没有文档的做法下（这已成为一种开发的文化），开发人员和测试人员都面临同样的困惑

1.3.3 H模型（1）

- 虽然软件开发期望有清晰的需求、设计和编码阶段，但实践告诉我们，严格的阶段划分只是一种理想状况。相应的测试之间也不存在严格的次序关系，各层次之间的测试也存在反复触发、迭代和增量关系

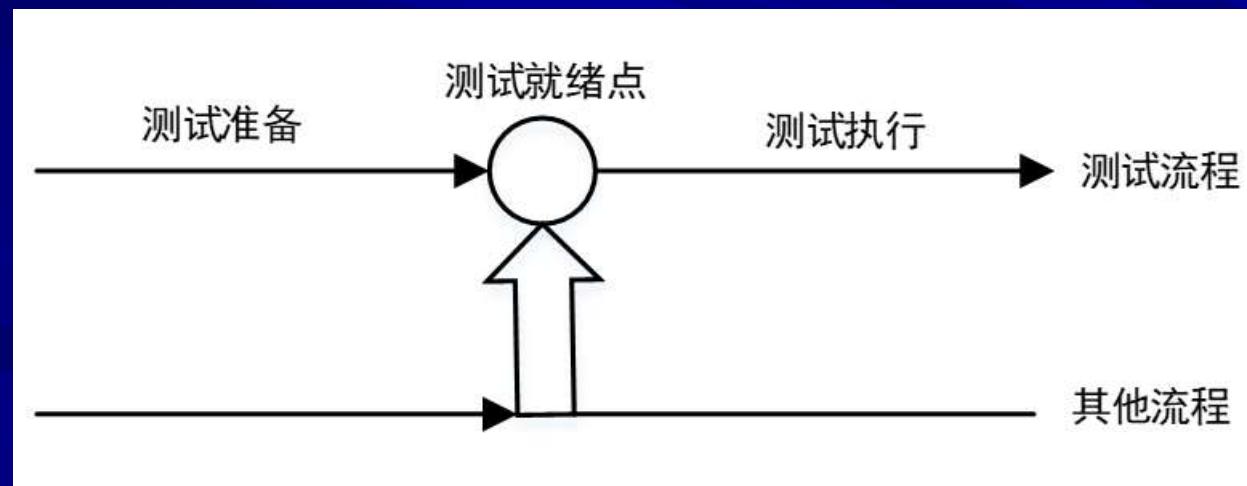
1.3.3 H模型（2）

- V模型和W模型都没有很好地体现测试流程的完整性
- H模型将测试活动完全独立出来，形成一个独立的流程，将测试准备活动和测试执行活动清晰地体现出来

1.3.3 H模型（3）

■ 测试流程

- 测试准备活动：需求分析、测试计划、测试分析、测试编码、测试验证
- 测试执行活动：测试运行、测试报告、测试分析



1.3.3 H模型 (4)

■ H模型揭示了

- 软件测试不仅仅指测试的执行，还包括很多其他的活动
- 软件测试是一个独立的流程，贯穿产品整个生命周期，与其他流程并发地进行
- 软件测试要尽早准备，尽早执行
- 软件测试是根据被测物的不同而分层次进行的。
。不同层次的测试活动可以是按照某个次序先后进行的，但也可能是反复的

1.3.3 H模型（5）

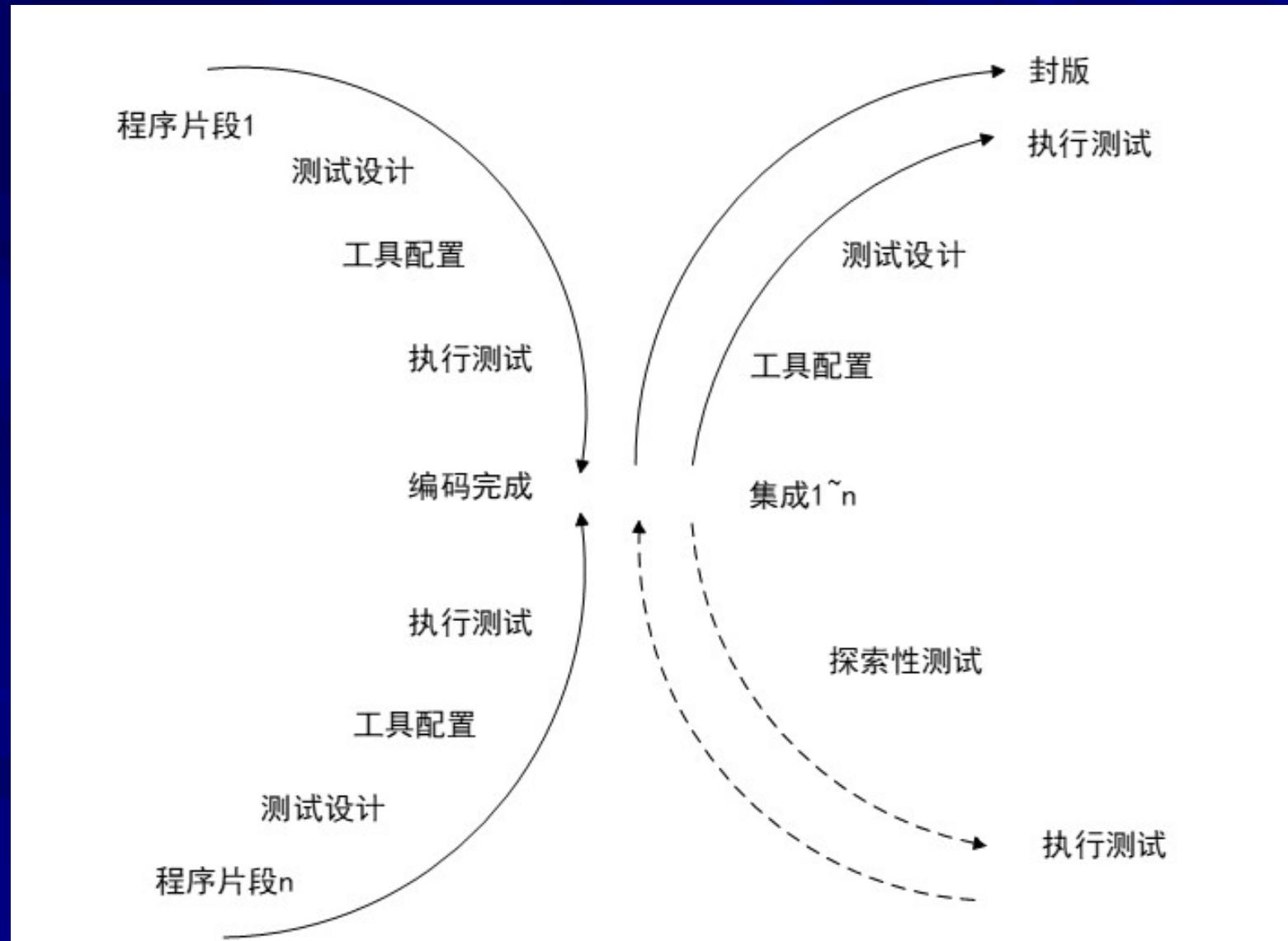
■ 应用H模型的意义

- 测试准备和测试执行分离，有利于资源调配，降低成本，提高效率
- 充分体现测试过程（不是技术）的复杂性
- 有组织、独立化的测试流程，有助于跟踪测试投入的流向

1.3.4 X模型（1）

- X模型左边是针对单独程序片段进行的相互分离的编码和测试，此后将进行频繁的交接，通过集成，最终成为可执行的程序，然后再对这些可执行程序进行测试。已通过集成测试的成品可以进行封装并提交给用户，也可以作为更大规模和范围内集成的一部分。多根并行的曲线表示变更可以在各个部分发生

1.3.4 X模型 (2)



1.3.4 X模型（3）

■ 优点

- 强调单元测试及集成测试的重要性
- 引入探索性测试使测试模型与现实更接近
- 缺陷修复时不受项目组内部人员限制

■ 缺点

- 只强调测试过程中的部分内容
- 没有对需求测试，验收测试等内容进行说明

1.3.5 测试模型的比较（1）

- V模型强调了在整个软件项目开发中需要经历若干个测试级别，而且每一个级别都与一个开发级别相对应，但它忽略了测试的对象不应该仅仅包括程序，或者说它没有明确地指出应该对软件的需求、设计进行测试，而这一点在W模型中得到了补充

1.3.5 测试模型的比较（2）

- W模型强调了测试计划等工作的先行和对系统需求和系统设计的测试，但W模型和V模型一样也没有专门针对软件测试的流程予以说明。事实上，随着软件质量要求越来越为大家所重视，软件测试也逐步发展成为一个独立于软件开发部的组织，就一个软件测试的细节而言，它都有一个独立的操作流程
- H模型表现为测试是独立的。只要测试前提具备了，就可以开始测试了

1.3.6 测试模型的使用

- 在实际的工作中，我们要灵活地运用各种模型的优点，在W模型的框架下，运用H模型的思想进行独立地测试，并同时将测试和开发紧密结合，寻找恰当的就绪点开始测试并反复迭代测试，最终保证按期完成预定目标

软件测试贯穿全生命周期

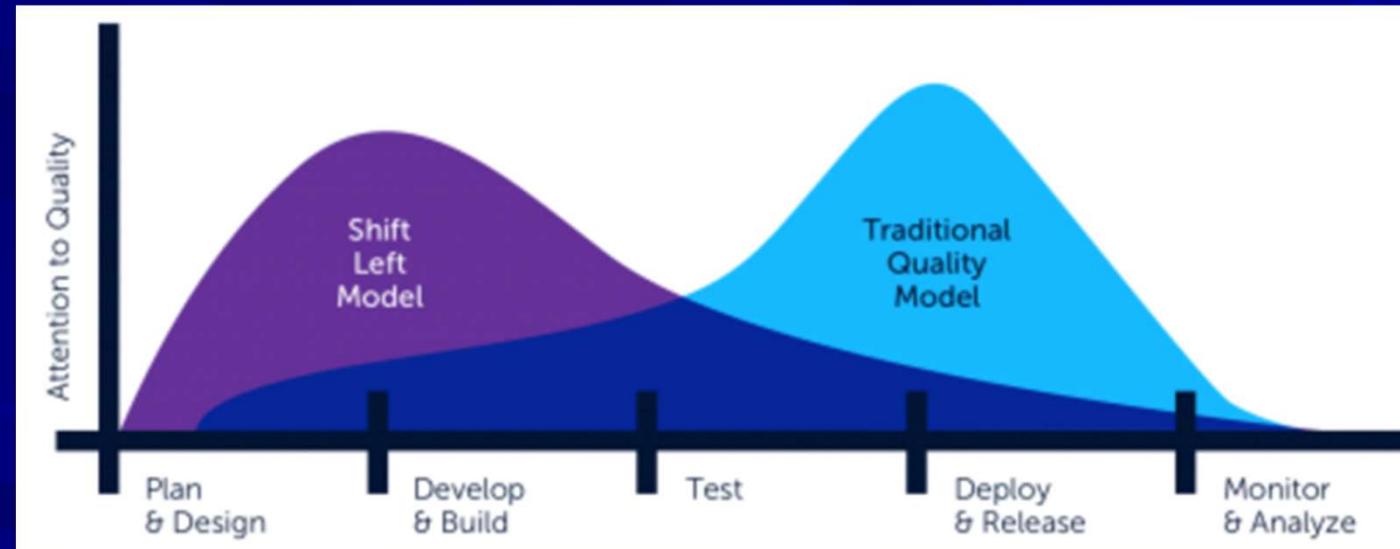
- 测试左移：不仅让开发人员做更多的测试，而且需要做需求评审、设计评审，以及验收测试驱动开发（ATDD）
- 测试右移：是在线测试（**Test in Production, TiP**），包括在线性能监控与分析、A/B测试和日志分析等，可以和现在流行的DevOp联系起来

测试左移和测试右移



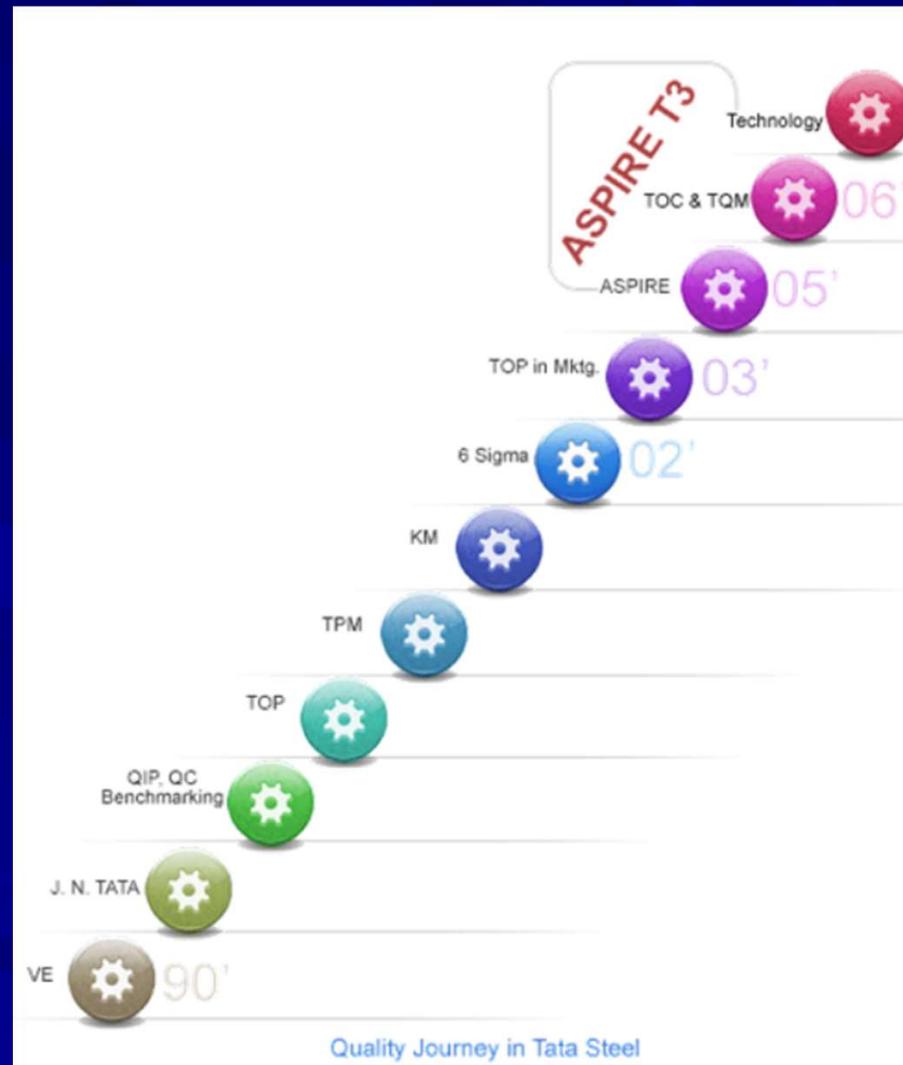
测试左移

- 需求评审、设计评审等
- 测试计划、测试设计可以在更早时候开始
- TDD、ATDD



2. 测试过程改进模型

- 2.1 TMM
- 2.2 TPI
- 2.3 CTP
- 2.4 STEP



2.1 TMM (1)

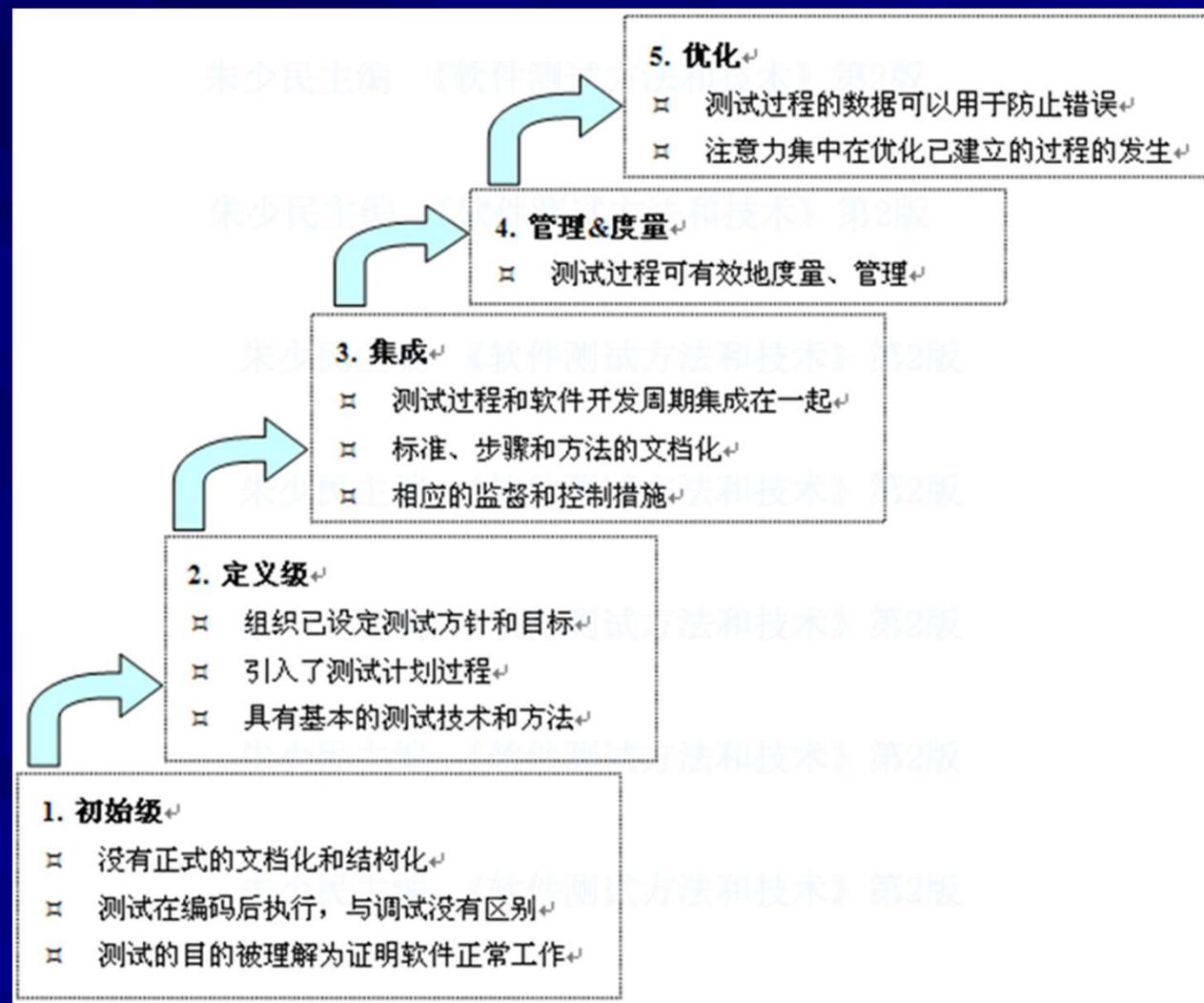
- 过程能力描述了遵循一个软件测试过程可能达到的预期结果的范围。TMM的建立，得益于以下3点
 - 充分吸收CMM的精华
 - 基于历史演化的测试过程
 - 业界的最佳实践

2.1 TMM (2)

- 5个级别的系列测试能力成熟度的定义，每个级别的组成包括到期目标、到期子目标活动、任务和职责等
- 一套评价模型，包括一个成熟度问卷、评估程序 和团队选拔培训指南

2.1 TMM (3)

■ TMM的5个级别简要描述



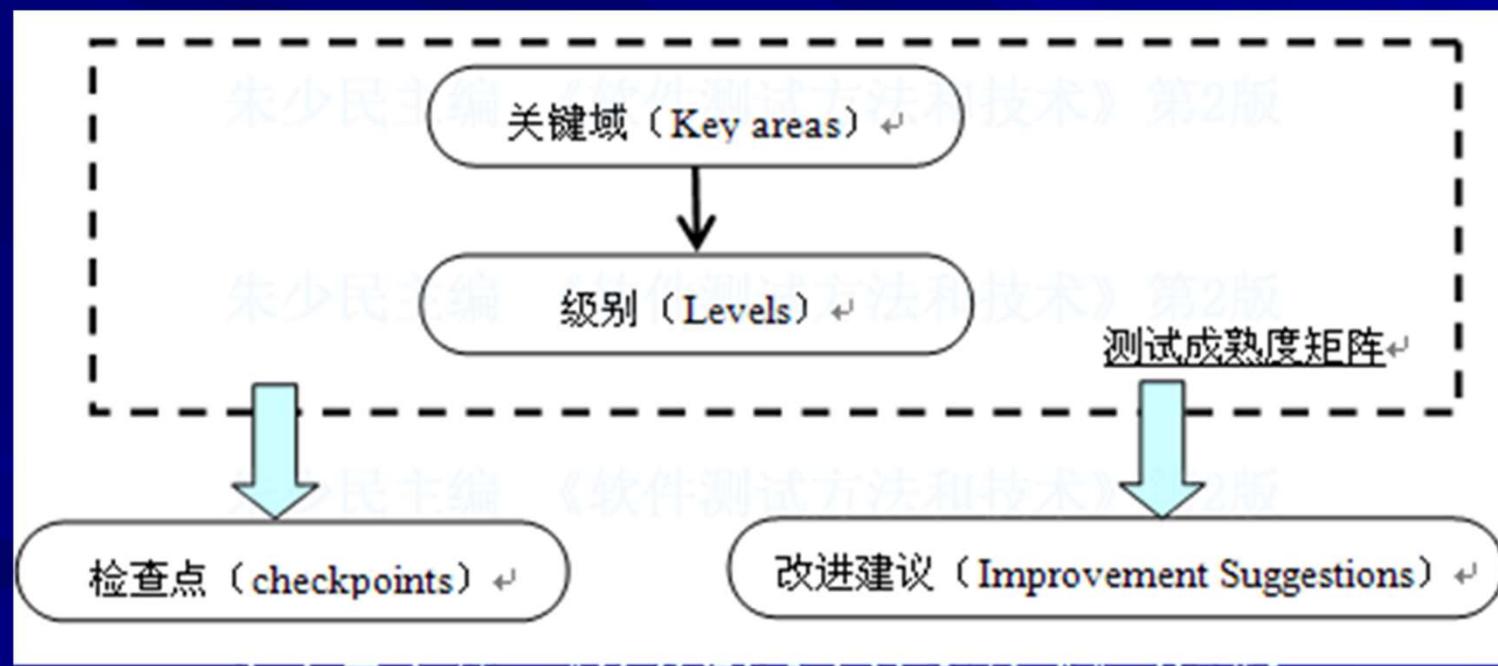
2.1 TMM (4)

■ TMM的4个级别内容

2 ^o	Phase Definition (阶段定义级)。 测试目标是验证软件符合需求，会采用基本的测试技术和方法。 ^o	测试被看做是有计划的活动； ^o 测试同调试分开； ^o 但编码完成后才进行测试工作。 ^o	启动测试计划过程； ^o 为基本的测试技术和方法制度化。 ^o
3 ^o	Integration (集成级)。 ^o 测试不再是编码后的一个阶段，而是把测试贯穿在整个软件生命周期中。测试是建立在满足用户或客户的需求上。 ^o	具有独立的测试部门； ^o 根据用户需求设计测试用例； ^o 有测试工具辅助进行测试工作； ^o 没有建立起有效的评审制度； ^o 没有建立起质量控制和质量度量标准。 ^o	建立软件测试组织； ^o 制订技术培训计划； ^o 测试在整个生命周期内进行； ^o 控制和监视测试过程。 ^o
4 ^o	Management and Measurement (管理和度量级)。 ^o 测试是一个度量和质量控制过程。 在软件生命周期中评审作为测试和软件质量控制的一部分。 ^o	进行可靠性、可用性和可维护性等方面的测试； ^o 采用数据库来管理测试用例； ^o 具有缺陷管理系统并划分缺陷的级别； ^o 还没有建立起缺陷预防机制，且缺乏自动地对测试中产生的数据进行收集和分析的手段。 ^o	实施软件生命周期中各阶段评审。 ^o 建立测试数据库并记录、收集有关测试数据； ^o 建立组织范围内的评审程序； ^o 建立测试过程的度量方法和程序； ^o 软件质量评价。 ^o
5 ^o	Optimization (优化级)。 ^o 具有缺陷预防和质量控制的能力； ^o 已经建立起测试规范和流程，并不断地进行测试过程改进。 ^o	运用缺陷预防和质量控制措施； ^o 选择和评估测试工具存在一个既定的流程； ^o 测试自动化程度高； ^o 自动收集缺陷信息； ^o 有常规的缺陷分析机制。 ^o	应用过程数据预防缺陷。统计质量控制。 ^o 建立软件产品的质量目标。 ^o 持续改进测试过程； ^o 优化测试过程。 ^o

2.2 TPI (1)

- TPI是基于连续性表示法的测试过程改进的参考模型，是在软件控制、测试知识以及过往经验的基础上开发出来的



2.2 TPI (2)

- 2.2.1 TPI 20个关键域
- 2.2.2 TPI 级别
- 2.2.3 TPI 检查点和建议
- 2.2.4 TPI 成熟度矩阵

2.2.1 TPI 20个关键域

- 测试策略
- 生命周期模型
- 介入时间
- 估计和计划
- 测试规格技术
- 静态测试技术
- 度量
- 测试自动化
- 测试环境
- 办公环境
- 承诺与动力
- 测试功能与培训
- 方法的范围
- 沟通
- 报告
- 缺陷管理
- 测试件管理
- 测试过程管理
- 评估
- 底层测试

2.2.2 TPI 级别

- 为了了解过程在每个关键域所处的状态，即对关键域的评估结果，通过级别来体现。模型提供了4个级别，由A到D，A是最低级。根据测试过程的可视性改善、测试效率的提高、或成本的降低以及质量的提高，级别会有所上升

2.2.3 TPI 检查点和建议

- 为了能客观地决定各个关键域的级别，TPI 模型提供了一种度量工具——检查点。每个级别都有若干个检查点，测试过程只有在满足了这些检查点的要求之后，才意味着它达到了特定的级别
- 检查点帮助我们发现测试过程中的问题，而建议会帮助我们解决问题，最终改进测试过程。建议不仅包含对如何达到下个级别的指导，而且还包括一些具体的操作技巧、注意事项等

2.2.4 TPI成熟度矩阵

关键域	总体等级	可控的					有效的					不断优化的			
		0	1	2	3	4	5	6	7	8	9	10	11	12	13
测试策略		A						B				C		D	
生命周期模型		A				B									
介入时间			A				B				C		D		
估计和计划				A							B				
测试规格技术		A		B											
静态测试技术					A		B								
度量						A			B			C		D	
测试自动化				A				B			C				
测试环境					A			B						C	
办公环境				A											
承诺与动力		A				B					C				
测试功能与培训			A				B			C					
方法的范围				A						B				C	
沟通			A		B						C				
报告		A		B		C					D				
缺陷管理		A			B		C				D				

2.3 CTP (1)

- 关键测试过程（Critical Test Process, CTP）评估模型主要是一个内容参考模型，一个上下文相关的方法，并能对模型进行裁剪
- 使用CTP的过程改进，始于对现有测试过程的评估，通过评估以识别过程的强弱，并结合组织的需要提供改进的意见

2.3 CTP (2)

- 计划（Plan）、准备（Prepare）、执行（Perform）和完善（Perfect）；计划和完善主要是管理工作，准备和执行是实践工作

CTP 12个关键过程（1）

- 测试
- 建立上下文关系和测试环境
- 质量风险评估
- 测试估算
- 测试计划
- 测试团队开发

CTP 12个关键过程 (2)

- 测试（管理）系统开发
- 测试发布管理
- 测试执行
- 缺陷报告
- 测试结果报告
- 变更管理

2.4 STEP (1)

- STEP (Systematic Test and Evaluation Process, 系统化测试和评估过程) 是一个内容参考模型，认定测试是一个生命周期活动，在明确需求后开始直到系统退役
- STEP与CTP比较类似，而不像TMMI和TPI，不要求改进需要遵循特定的顺序
- 某些情况下，STEP评估模型可以与TPI成熟度模型结合起来使用

2.4 STEP (2)

- 基于需求的测试策略
- 在生命周期初始开始进行测试
- 测试用作需求和使用模型
- 由测试件设计导出软件设计（测试驱动开发）
- 及早发现缺陷或完全的缺陷预防
- 对缺陷进行系统分析
- 测试人员和开发人员一起工作

3. 软件测试标准和规范

- 3.1 概述
- 3.2 ISO/GB软件质量体系标准
- 3.3 软件测试规范



3.1 概述

- 3.1.1 标准的定义
- 3.1.2 标准的层次
- 3.1.3 标准的作用

3.1.1 标准的定义

■ 标准是由一个公认的机构制定和批准的文件。它对活动或活动的结果规定了规则、导则或特性值，供共同和反复使用，以实现在预定领域内最佳秩序的效益

3.1.2 标准的层次（1）

■ 国际标准

- ISO（国际标准化组织）
- IEC（国际电工委员会）
- ITU（国际电信联盟）

3.1.2 标准的层次（2）

■ 区域标准

- CEN（欧洲标准化委员会）
- CENELEC（欧洲电工标准化委员会）
- ETSI（欧洲电信标准学会）
- ARSO（非洲地区标准化组织）
- ASMO（阿拉伯标准化与计量组织）

3.1.2 标准的层次 (3)

■ 国家标准

- GB (中华人民共和国国家技术监督局)
- ANSI (美国国家标准协会)

■ 行业标准

- IEEE (美国电气和电子工程师学会)
- DOD-STD (美国国防部标准)
- MIL-STD (美国军用标准)
- SJ (中华人民共和国电子行业标准)
- GJB (中华人民共和国国家军用标准)

3.1.2 标准的层次 (4)

- 地方标准
 - DB
- 企业标准
 - 美国波音飞机
 - 德国西门子
- 项目规范
 - 计算机集成制造系统（CIMS）的软件工程规范

3.1.3 标准的作用

- 标准是保障产品质量，规范市场的标尺
- 标准是评测的重要依据
- 标准是用户、开发方以及评测机构的桥梁
- 标准有利于软件评测与国际接轨

3.2 ISO/GB软件质量体系标准

- 3.2.1 ISO软件质量标准结构
- 3.2.2 ISO 9000-3

3.2.1 ISO软件质量标准结构 (1)

- ISO9000系列标准的主体部分分为两组
 - “需方对供方要求质量保证”的标准ISO9001—9003
 - “供方建立质量保证体系”的标准ISO9004

3.2.1 ISO软件质量标准结构 (2)

- ISO9001：设计/开发、生产、安装和服务中质量保证模式
- ISO9002：生产和安装中的质量保证模式
- ISO9003：最终检验和测试中的质量保证模式
- ISO9004：质量管理和质量体系要素导则

3.2.2 ISO 9000-3 (1)

- ISO9000-3其实是ISO质量管理和质量保证标准在软件开发、供应和维护中的使用指南，并不作为质量体系注册/认证时的评估准则，主要考虑软件行业的特殊性制定。参照ISO9001《质量体系 设计、开发、生产、安装和服务的质量保证模式》，并引用ISO8402《质量管理和质量保证术语》，使得ISO9000系列标准应用范围得以拓展

3.2.2 ISO 9000-3 (2)

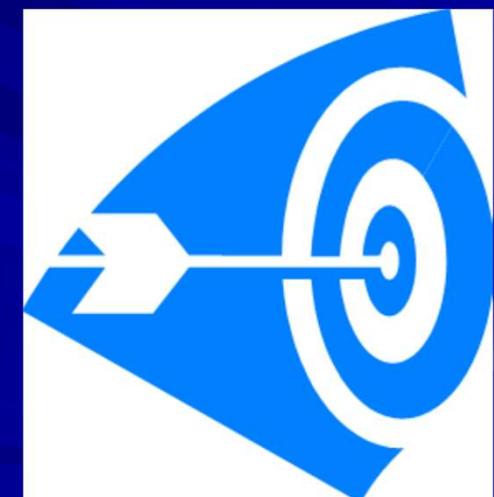
■ ISO 9000-3 体系结构

- 合同评审
- 需方需求规格说明
- 开发计划
- 质量计划
- 设计和实现
- 测试和确认
- 验收
- 复制、交付和安装
- 维护



3.3 软件测试规范

- 软件测试规范就是对软件测试的流程过程化并对每一个过程元素进行明确的界定，形成完整的规范体系



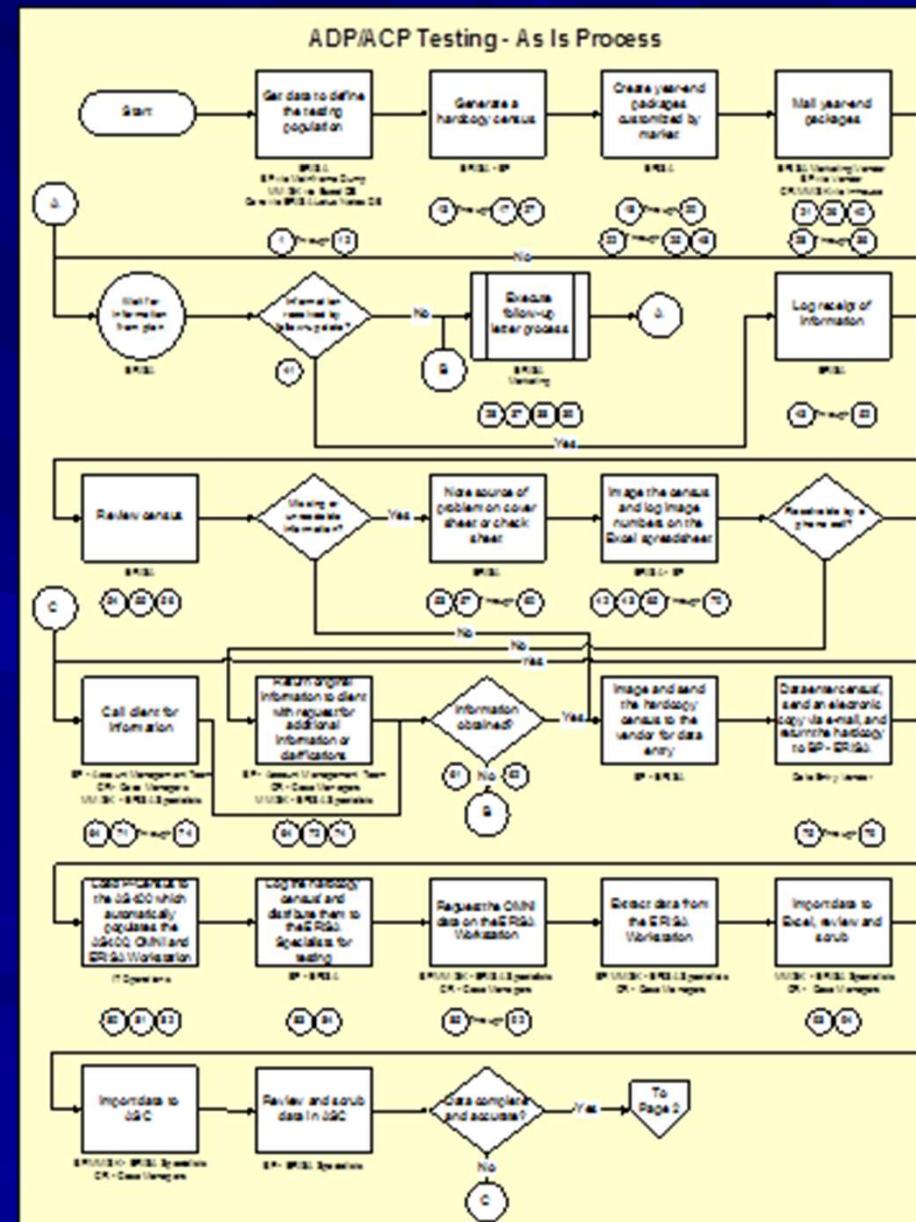
完整的软件测试规范是怎样的

- 规范本身的详细说明，比如规范目的、范围、文档结构、词汇表、参考信息、可追溯性、方针、过程/规范、指南、模板、检查表、培训、工具、参考资料等等



制定测试规范需要考虑的内容

- 角色的确定
 - 进入的准则
 - 输入项
 - 活动过程
 - 输出项
 - 验证与确认
 - 退出的准则
 - 度量



4. 软件测试管理和评判体系

- 4.1 测试管理与评判的必要性
- 4.2 软件测试的管理和评判体系发展现状
- 4.3 如何建立测试管理与评判体系



4.1 测试管理与评判的必要性

- 监视和测量软件产品
- 识别和控制不符合要求的产品
- 验证产品设计和开发
- 监视和测量软件过程



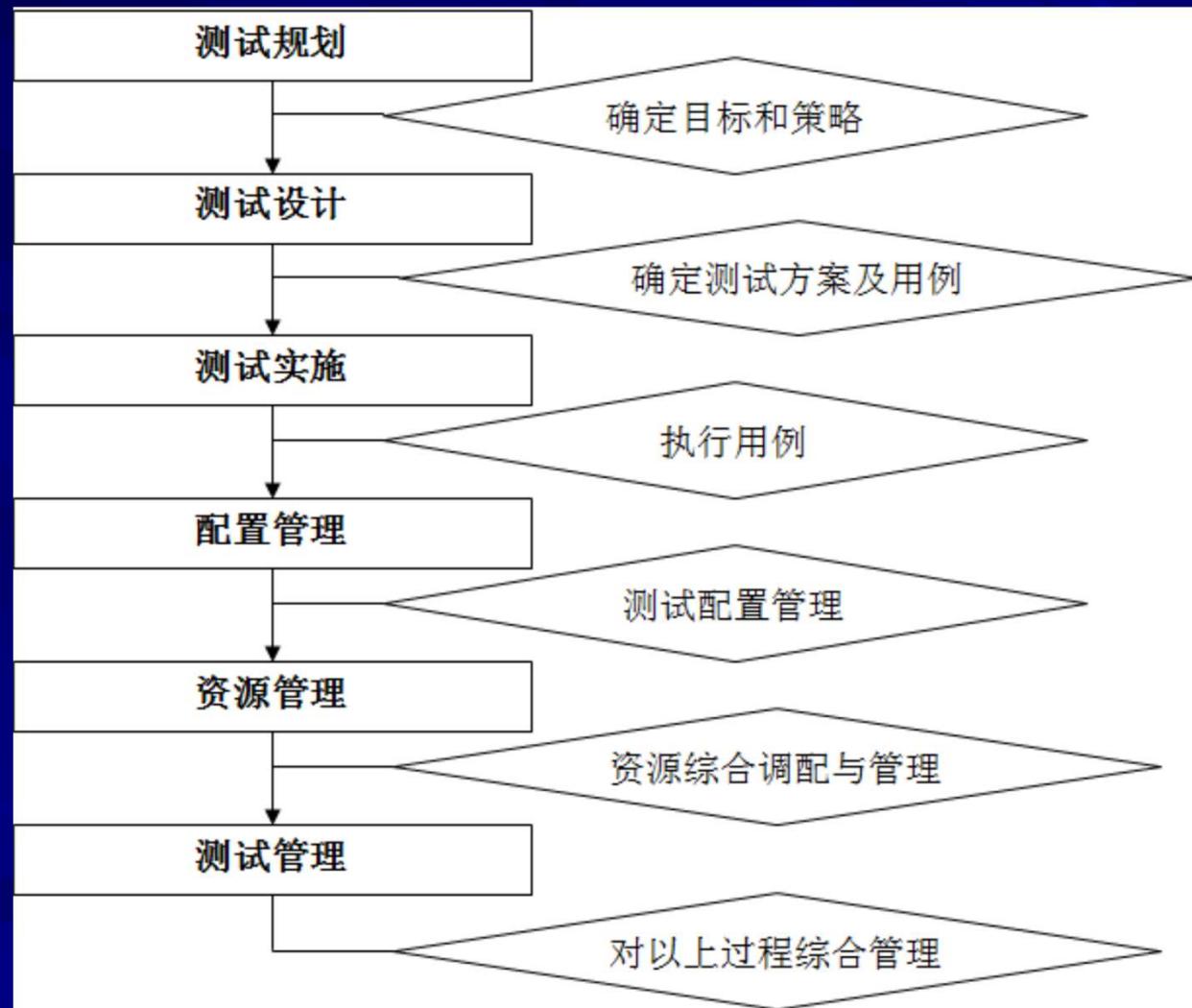
4.2 测试管理和评判体系发展现状（1）

- 美国质量保证研究所对软件测试的研究结果表明：越早发现软件中存在的问题，开发费用就越低；在编码后修改软件缺陷的成本是编码前的**10倍**，在产品交付后修改软件缺陷的成本是交付前的**10倍**；软件质量越高，软件发布后的维护费用越低。另外，根据对国际著名IT企业的统计，它们的软件测试费用占整个软件工程所有研发费用的**50%以上**

4.2 测试管理和评判体系发展现状（2）

■ 中国软件企业在软件测试方面与国际水准仍存在较大差距。首先，认识上重开发、轻测试，没有认识到软件项目的如期完成不仅取决于开发人员，更取决于测试人员；其次，管理上随意、简单，没有建立有效、规范的软件测试管理和评判体系；另外，缺少自动化工具的支持，大多数企业在软件测试时并没有建立软件测试管理与评判体系。

4.3 如何建立测试管理与评判体系



参考资料（1）

- 第2章，张旸旸 于秀明主编 软件评测师教程（第2版），北京：清华大学出版社，2021
- 第1.4、1.5节、4章，朱少民 主编，软件测试方法和技术（第4版），北京：清华大学出版社，2022

参考资料 (2)

- (以) Daniel Galin著, 王振宇 陈利 王志海等译, 软件质量保证, 北京: 机械工业出版社, 2004
- 杜庆峰编著, 高级软件测试技术, 北京: 清华大学出版社 , 2011
- 第1章, 宫云战 主编, 软件测试教程 (第3版), 北京: 机械工业出版社, 2021

测试设计技术

Test Design Techniques

计算机学院 单纯

sherryshan@bit.edu.cn

2025年12月3日

内容概览

- 1. 测试点的确定
- 2. 测试用例设计概述
- 3. 测试用例设计技术

1. 测试点的确定（1）

■ ISO质量体系

- 在概要设计或详细设计中应明确指出每个单元模块的测试要点、指标和方法

■ CMM质量体系

- 在系统的用例模型描述中应明确指出每个用例模型的优先级及用例工作流程，每一个用例模型为一个测试点，用例模型中每一个测试需求至少应有两个测试用例

1. 测试点的确定（2）

- 测试用例应由测试设计员或分析设计员来制定，而不是普通的测试员
- 测试点应由分析设计员确立，与测试人员无关
- 测试工作展开于项目立项后，而不是代码开发完成之后
- 测试对象不仅仅是源代码，还包括需求分析、需求规格说明书、概要设计、概要设计说明书、详细设计、详细设计说明书、使用手册等各阶段的文档

Thank You