

The background features a light gray grid. Overlaid on this are several large, stylized, semi-transparent swirls in purple, green, and blue. Scattered throughout are numerous small, yellow, four-pointed starburst shapes.

1. 软件测试概述



本章目标

- 了解软件测试的发展
- 了解软件测试中的核心概念和原则
- 能够在软件开发周期中开展软件测试相关活动



第1章 软件测试概述

1.1 软件测试的发展

1.2 软件测试基础

1.3 软件测试与软件开发过程

1.4 软件测试过程

1.1 软件测试的发展

- 早期:

测试等同于“调试”;
由开发人员完成;
投入少,介入晚

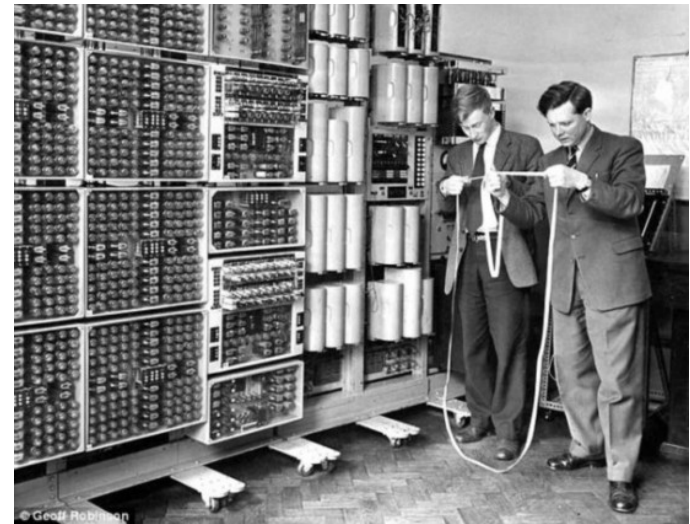
- 1950S:

测试被视为一种发现软件缺陷的活动;

开始与调试区别开;

测试始终后于开发;

缺乏有效的测试方法



测试: 发现软件失效

调试: 定位软件错误并将其修复





1.1 软件测试的发展

• 1970S:

软件工程思想开始深入人心;

-1972, **Bill Hetzel** 组织了第一次软件测试会议

-1973, **Bill Hetzel** 给出软件测试第一个定义:

“测试就是建立一种信心，认为程序能够按预期设想运行”

核心思想：测试是试图验证软件是工作的

1.1 软件测试的发展

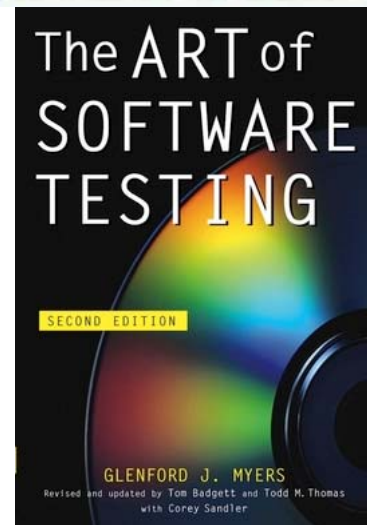
- **1970S:**

Glenford J. Myers:

测试是尽可能多地发现软件错误

测试是为发现错误而执行一个程序或系统的过程

测试不应该着眼于验证软件是工作的，相反应该首先认定软件是有错误的，然后用逆向思维去发现尽可能多的错误





1.1 软件测试的发展

• 1980S:

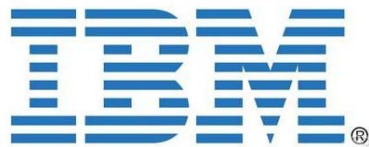
- 软件趋向大型化、复杂化，软件质量越来越重要
- 开发从混乱无序过渡到结构化开发
- 测试基础理论和实用技术开始形成
- 测试被作为软件质量保证的重要手段
- 1983年, **IEEE**给出软件测试的定义



1.1 软件测试的发展

- **1990S-至今:**

- 测试理论和技术进一步完善
- 测试工具发展迅速，大大提高测试的自动化程度
- 商业化软件测试工具和开源软件测试工具
- Web**测试，面向对象测试，云测试，





第1章 软件测试概述

1.1 软件测试的发展

1.2 软件测试基础

1.3 软件测试与软件开发过程

1.4 软件测试过程



1.2 软件测试基础

- 基本术语
- 缺陷分类
- 测试用例
- 狭义软件测试
- 测试的目的、阶段、原则
- 软件测试的周期性
- 软件测试技术概要
- 广义软件测试

1.2 软件测试基础—基本术语

- 故障(**fault**) / 缺陷(**defect**)

A fault is the cause of an error (IEC 65A 122).

- 过错缺陷 / 遗漏缺陷
- 驻留故障密度（每千行代码的故障数目）：
 - 关键财务或财产软件为：每千行代码 **1~10**个故障
 - 关键的生命软件为：每千行代码**0.01~1**个故障

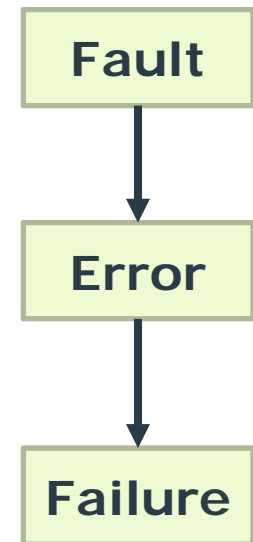
- 错误(**Error**)

an error is that part of the system state which is liable to lead to failure. (IEC65A 122) (IEC65A 94).

- 失效(**failure**)

the manifestation of an error in the system or software (IEC65A 122) (IEC65A 94)

- 缺陷 / 故障在运行期的不正常表现。执行缺陷才会导致失效，不执行并不会导致失效。
- 观察到的是失效，缺陷“看不到”，从失效定位缺陷不容易





1.2 软件测试基础—基本术语

- 测试用例(**test case**)

- 所谓测试用例是为特定的目的而设计的一组测试输入、执行条件和预期输出。

- 测试池(**test pool**)

- 供选择的测试用例的集合

- 测试集(**test suite**)

- 一轮测试中所选中的测试用例的集合

1.2 软件测试基础—基本术语

- 测试(test)

- 采用测试用例执行软件的活动。目标是发现失效，或者演示正确的执行。
- 测试是比较广义的概念，对于没有预期输出的情况，只能称测试，而不是测试用例

- 测试预言(test oracle)

- 一种独立于被测系统的程序或机制，用于确认对于给定的输入，系统是否有一个正确的输出。

六爻金钱卦工具





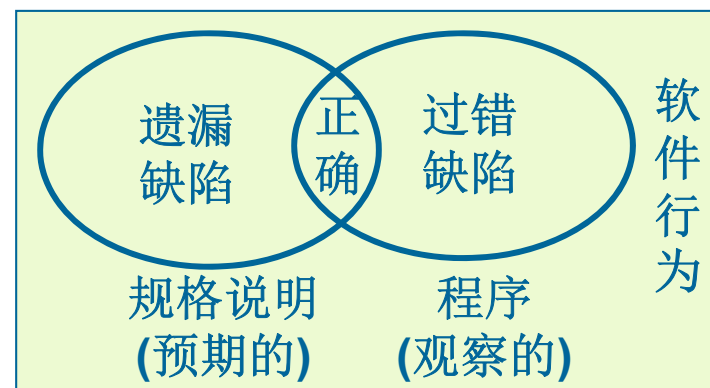
1.2 软件测试基础—缺陷分类

- 
- Q: 你觉得软件缺陷可以分为哪些类别?

1.2 软件测试基础—缺陷分类

从外部需求的满足情况来看

- (1) 软件未达到产品说明书中已经标明的功能；
- (2) 软件出现了产品说明书中指明不会出现的错误；
- (3) 软件未达到产品说明书中虽未指出但应当达到的目标
- (4) 软件功能超出了产品说明书中指明的范围；
- (5) 软件测试人员认为软件难以理解、不易使用，或者最终用户认为该软件使用效果不良。



1.2 软件测试基础—缺陷分类

从软件内部设计来看，主要有

- (1) 输入/输出缺陷;
- (2) 逻辑缺陷;
- (3) 计算缺陷;
- (4) 接口缺陷;
- (5) 数据缺陷。

《IEEE Standard Classification for Software Anomalies (IEEE 1044-2009)》



1.2 软件测试基础—缺陷分类

- 输入/输出缺陷

- 输入

- 不接受正确的输入
 - 接受不正确的输入
 - 描述有错或遗漏
 - 参数有错或遗漏

- 输出

- 格式有错
 - 结果有错
 - 在错误的时间产生正确的结果
 - 不一致或遗漏结果
 - 不合逻辑的结果
 - 拼写/语法错误
 - 修饰词错误

- 逻辑缺陷

- 遗漏情况
 - 重复情况
 - 极端条件出错
 - 解释有错
 - 遗漏条件
 - 外部条件有错
 - 错误变量的测试
 - 不正确的循环迭代
 - 错误的逻辑运算符

1.2 软件测试基础—缺陷分类

• 计算缺陷

不正确的算法
遗漏计算
不正确的操作数
不正确的操作
括号错误
精度不够（四舍五入，截断）
错误的内置函数

• 接口缺陷

不正确的中断处理
I/O时序有错
调用了错误的过程
调用了不存在的过程
参数不匹配（类型，个数）
不兼容的类型
过量的包含

• 数据缺陷

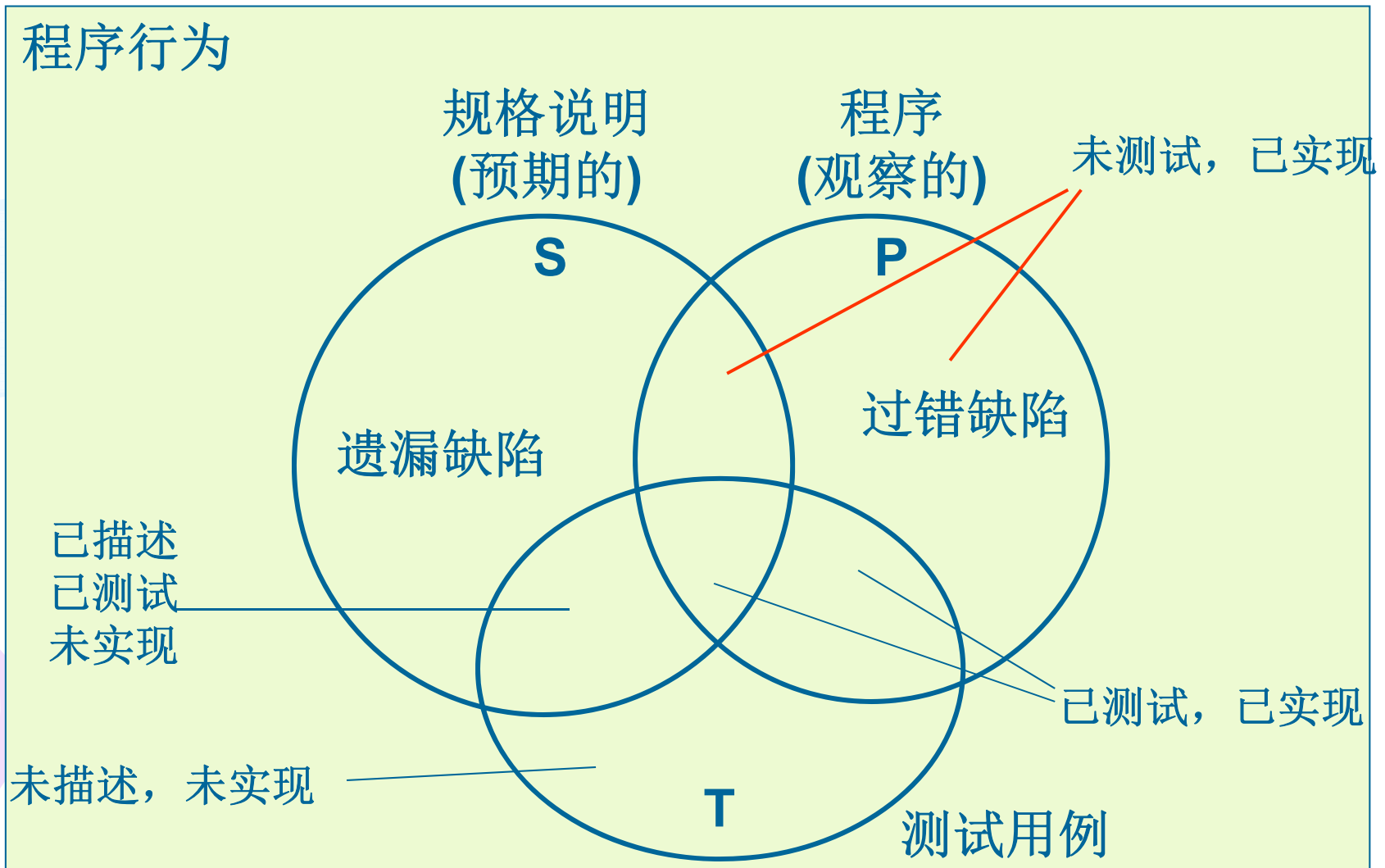
不正确的初始化
不正确的存储/访问
错误的标志/索引值
不正确的打包/拆包
使用了错误的变量
错误的数据引用
缩放数据范围或单位错误
不正确的数据维数
不正确的下标
不正确的类型
不正确的数据范围
传感器数据超出限制
出现1次断开
不一致的数据

1.2 软件测试基础—缺陷分类

从严重程度来看

级别	常见现象
轻微	词语拼写错误
中等	误导或重复信息
使人不悦	被截断的名称，如 0.00 元账单
影响使用	有些事务没有处理
严重	丢失事务
非常严重	不正确的事务处理
极为严重	经常出现“非常严重”的错误
无法忍受	数据库破坏
灾难性	系统停机
容易传染	扩展到其它系统的系统停机

缺陷与测试



1.2 软件测试基础—测试用例

- 成分

- 输入

- 前提（测试用例执行前已经存在的环境）
 - 实际的软件输入

- 输出

- 后果
 - 实际输出

- 输入和预期输出的确定是影响测试自动化的关键

测试用例ID	001
模块	SMS
标题	短信接收
设计人	joe
设计日期	2010-09-15
版本	0.01
级别	1
目的	验证正常接收短信
前提	待测手机1, 参照手机2, SIM卡两张
输入	参照手机给待测手机发一条短信
预期输出	手机能够正常接收短信
后果	无
执行历史	...

1.2 软件测试基础—测试用例

- 应妥善管理测试用例信息，以便复用
 - 一块硬盘**300**块，本科毕业生日均收入大于**300**
- 测试用例的评价
 - 检测软件缺陷的有效性
 - 测试用例的可重用性
 - 测试用例的经济性
 - 测试用例的可维护性





1.2 软件测试基础—狭义软件测试

通常对软件测试的定义有两种描述：

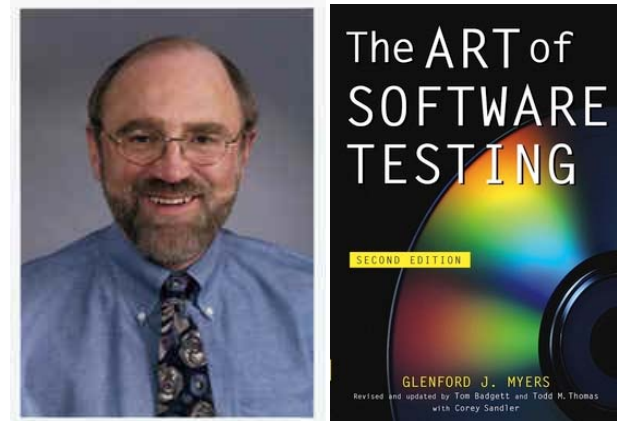
- **定义1：**软件测试是为了发现错误而执行程序的过程[G. J. Myers]。

1.2 软件测试基础—测试的目的

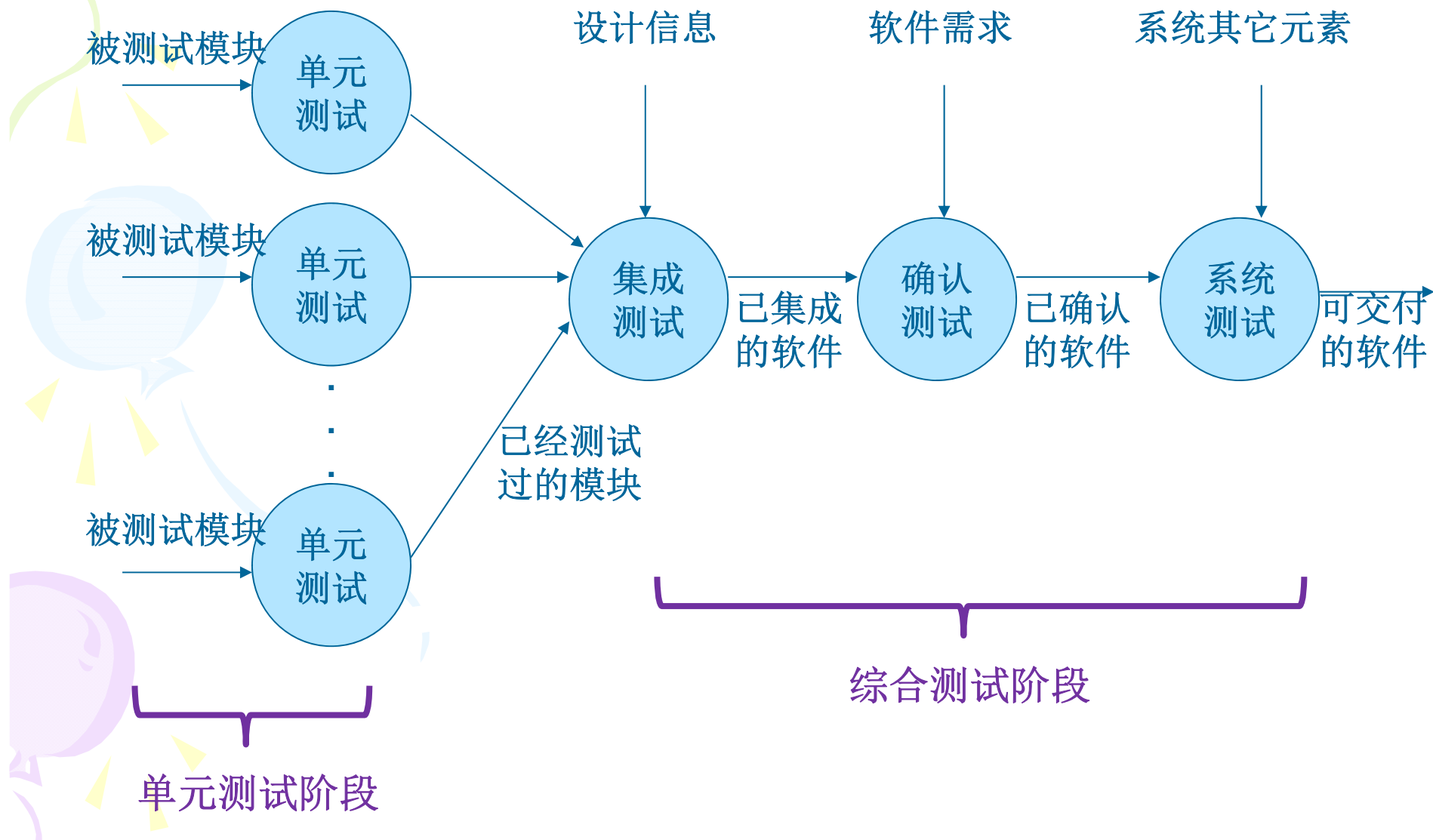
- (1) 测试是程序的执行过程，目的在于发现错误；不能证明程序的正确性，除非仅处理有限种情况。
- (2) 检查系统是否满足需求也是测试的期望目标。
- (3) 一个好的测试用例在于发现了还未曾发现的错误；一次成功的测试则是发现了错误的测试。

注意：测试无法说明错误不存在，只能说明软件错误已出现。

Myers. 《The Art of Software Testing》




1.2 软件测试基础—测试的阶段



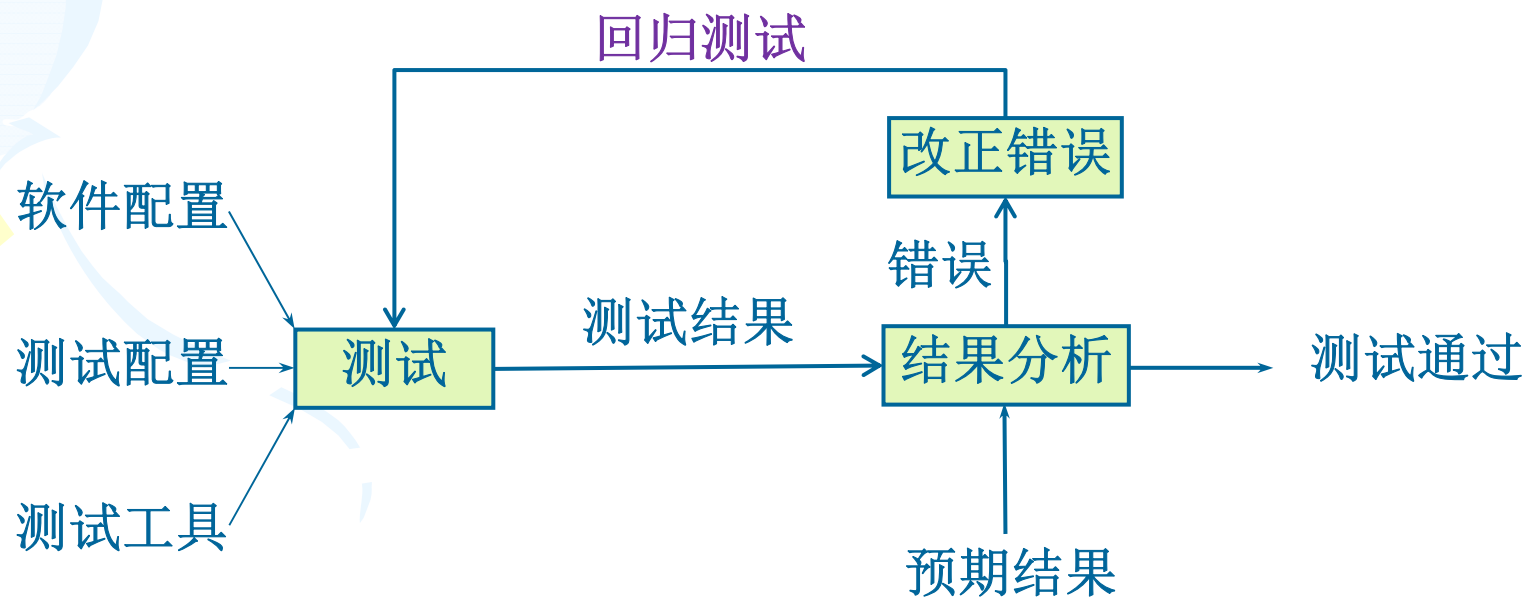


1.2 软件测试基础—测试的阶段

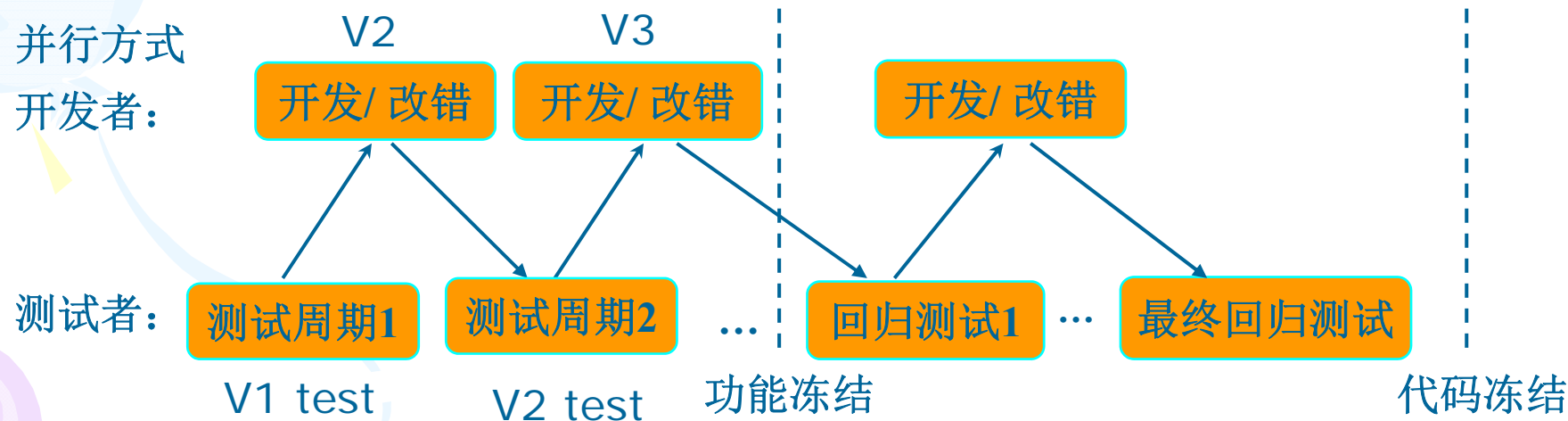
- **单元测试：**针对软件设计的最小单元—程序模块，进行正确性检验的测试工作。以发现各模块内部可能存在的各种差错。
 - **集成测试：**在单元测试的基础上，将所有模块按照设计要求组装成为系统进行测试。
 - **确认测试（有效性测试）：**验证软件的功能、性能和其它特性是否与用户的要求一致
 - **系统测试：**将测试的软件作为整个计算机系统的一个元素与计算机硬件、外设、某些支持软件、数据和人员等其他系统元素结合在一起，在实际运行环境下，对计算机系统进行组装测试和确认测试
- 

1.2 软件测试基础—测试的周期性

- 软件测试是“测试→改错→再测试→再改错”的循环过程

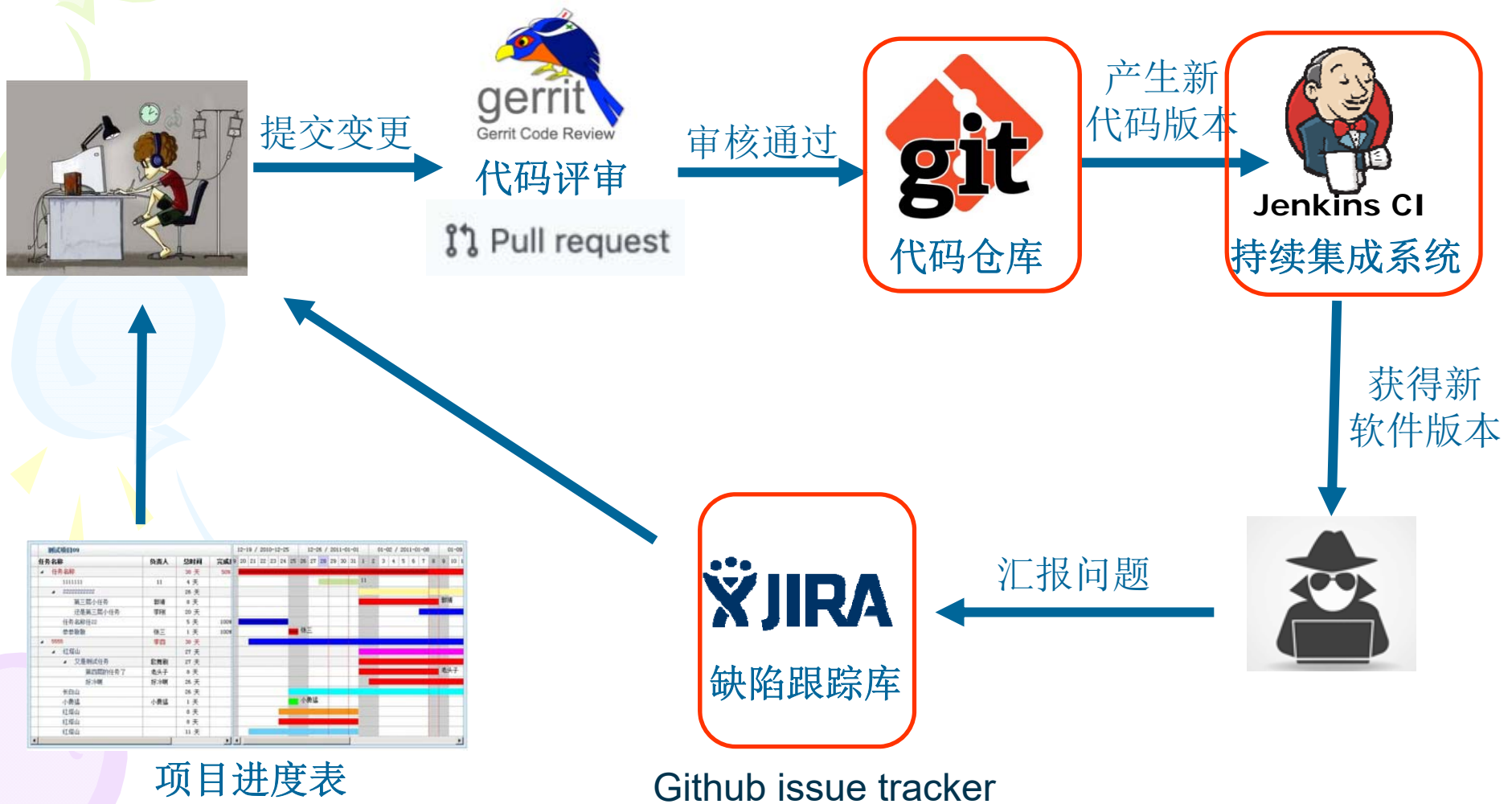


串行与并行开发测试模式



可以针对某一版本开展测试，而不是当前最新代码

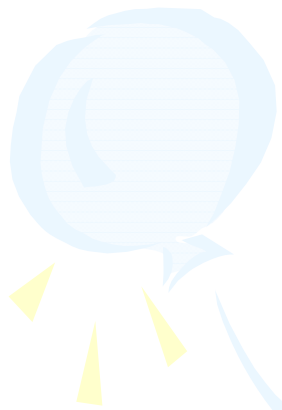
软件开发与测试：项目管理



<https://tower.im>

演示

作业一





1.2 软件测试基础—测试的原则

一些原则：

- (1) 尽早地和及时地测试；
- (2) 测试用例应当由测试数据和预期结果这两部分组成；
- (3) 测试用例应包括合理的输入条件和不合理的输入条件；
- (4) 严格执行测试计划，排除测试的随意性；
- (5) 应对每一个测试结果做全面的检查；
- (6) 保存测试计划、测试用例、出错统计和最终分析报告，为维护工作提供充分的资料。

1.2 软件测试基础—正确认识软件测试

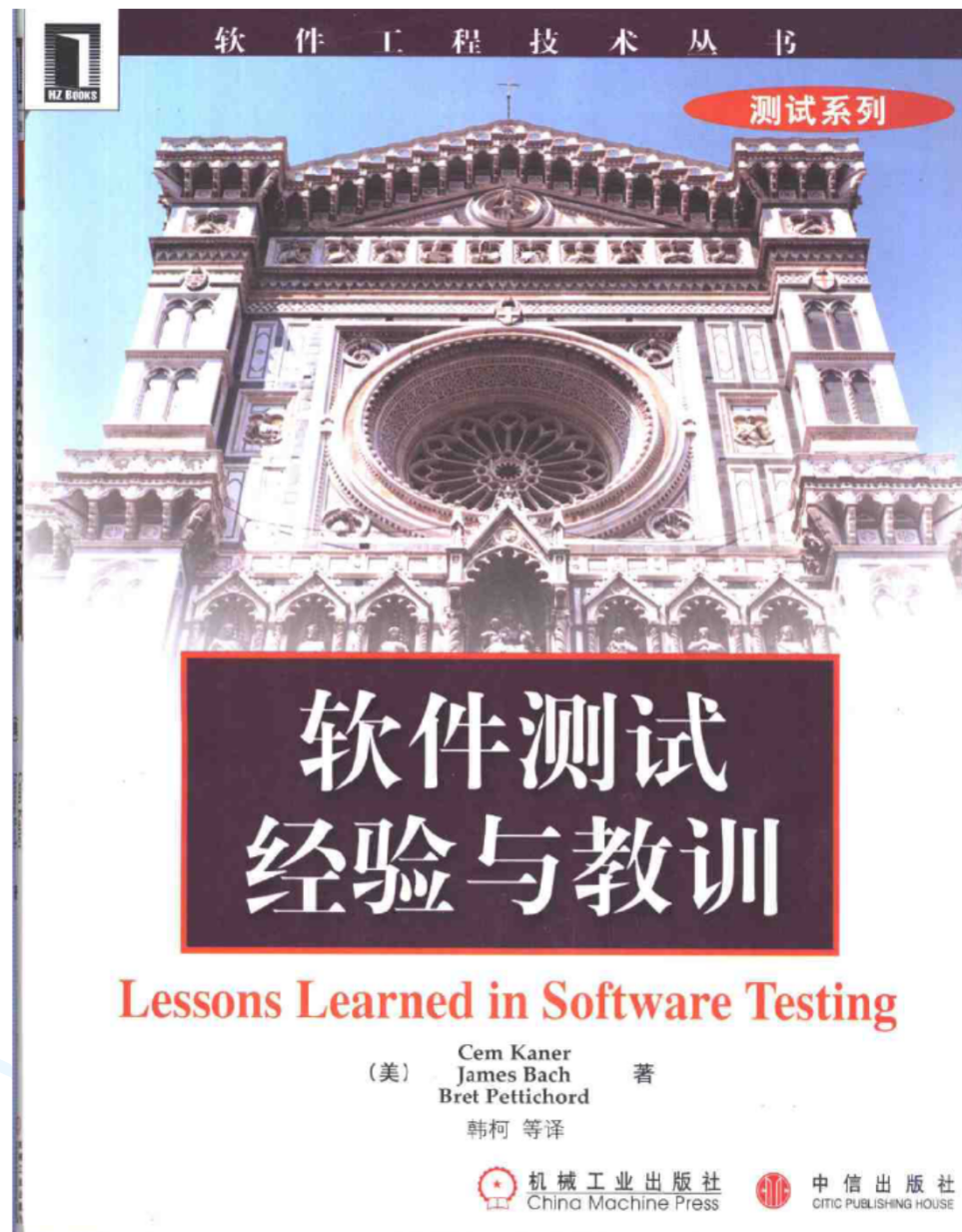
- 完全测试程序是不可能的，许多缺陷很难发现
 - 程序输入 x 、 y ,输出 z ,在32位机上运行,如 x 和 y 只取整数,则完全测试时测试数据有： $2^{32} * 2^{32} = 2^{64}$
- 软件测试是有风险的行为
 - 用户可能要求赔偿软件缺陷造成的损失
- 找到的软件缺陷越多，未来引发更多缺陷的风险越大
- 并非所有软件缺陷都能修复、应该修复
 - 难以说清的软件缺陷
 - 没有足够的时间
 - 修复的风险太大（也许会引入更大的错误）
 - 不值得修复（用户可以预防）



1.2 软件测试基础—正确认识软件测试

- 不正确的理解

- 如果发布的软件有质量问题，那是软件测试人员的错。
- 软件测试是测试人员的事，与开发人员无关。
- 设计—实现—测试，软件测试是开发后期的一个阶段。



近300条经验教训

1.2 软件测试基础—软件测试技术概要

- 软件测试的策略

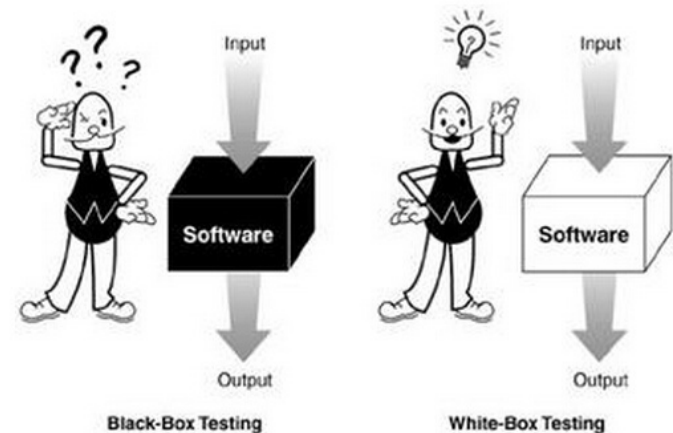
- 就是测试将按照什么样的思路 and 方式进行。如采用什么技术、什么步骤等

- 软件测试技术：

- (1) 黑盒（功能性）测试和白盒（结构性）测试

- (2) 静态测试和动态测试

- (3)





软件测试技术的发展趋势

- 与形式验证相结合的测试技术
- 静态测试分析技术

代码审查、走查，代码度量、规则检查等

- 测试自动化与集成化
- 测试与开发过程的一体化

测试驱动的开发、模型驱动测试等

- 智能化测试

1.2 软件测试基础—广义软件测试

- 所有发布的软件产品都会因为缺陷而导致用户的困扰和开发者时间和金钱上的额外开支。
- 这些导致成本风险的软件问题可以通过在软件生命周期的每一个阶段中充分规划和执行验证(**verification**)和确认(**validation**)而大大降低。
- 由此，广义的软件测试实际是由确认、验证、测试三个方面组成。

1.2 软件测试基础—广义软件测试

- 确认(Validation)

- 评估将要开发的软件产品是否是正确无误、可行和有价值的。包含了对用户需求满足程度的评价,意味着确保一个待开发软件是正确无误的,满足用户对产品的构想。

- 验证(Verification)

- 检测软件开发的每个阶段、每个步骤的结果是否正确无误,是否与软件开发各阶段的要求或期望的结果相一致。主要采用数学方法,是对软件功能属性的严格检查。

- 测试(Testing)

- 即狭义的测试

1.2 软件测试基础—广义软件测试

- 软件生命期中，确认、验证、测试各有侧重阶段
 - 确认：主要体现在计划阶段、需求分析阶段、也会出现在测试阶段
 - 验证：主要体现在设计阶段和编码阶段
 - 测试：主要体现在编码阶段。
- 确认、验证、测试是相辅相成的。确认无疑会产生验证和测试的标准，而验证和测试通常又会帮助完成一些确认，特别是在系统测试阶段。

1.2 软件测试基础—广义软件测试

- 广义地看

- 软件测试不等于程序测试。
- 软件测试贯串于软件定义和开发的整个过程。
- 软件开发过程中所产生的需求规格说明、概要设计规格说明、详细设计规格说明以及源程序等都是软件测试的对象。

- 软件产品组成部分

- (1) 程序代码 (2) 帮助文件 (3) 用户手册
- (4) 样本和示例 (5) 标签 (6) 产品支持信息
- (7) 图表和标志 (8) 错误信息 (9) 广告与宣传材料
- (10) 软件的安装 (11) 软件说明文件
- (12) 测试错误提示信息



1.2 软件测试基础—软件测试的局限性

• 软件测试的局限性

— 软件质量缺陷的复杂性

- 软件是不可见的复杂的逻辑体
- 软件开发的环节较多
- 测试投入有限，不能做穷举测试

— 技术上的局限性

- 测试工具本身也需要测试
- 测试工具对软件的理解程度达不到人的理解程度
- 软件测试方法与技术需要不断地扩展
- 软件质量没有统一指标来评价



第1章 软件测试概述

1.1 软件测试的发展

1.2 软件测试基础

1.3 软件测试与软件开发过程

1.4 软件测试过程



软件测试与软件开发过程

- 一般性的问题研究和解决过程



- 分析和定义问题

- 探索解决方案

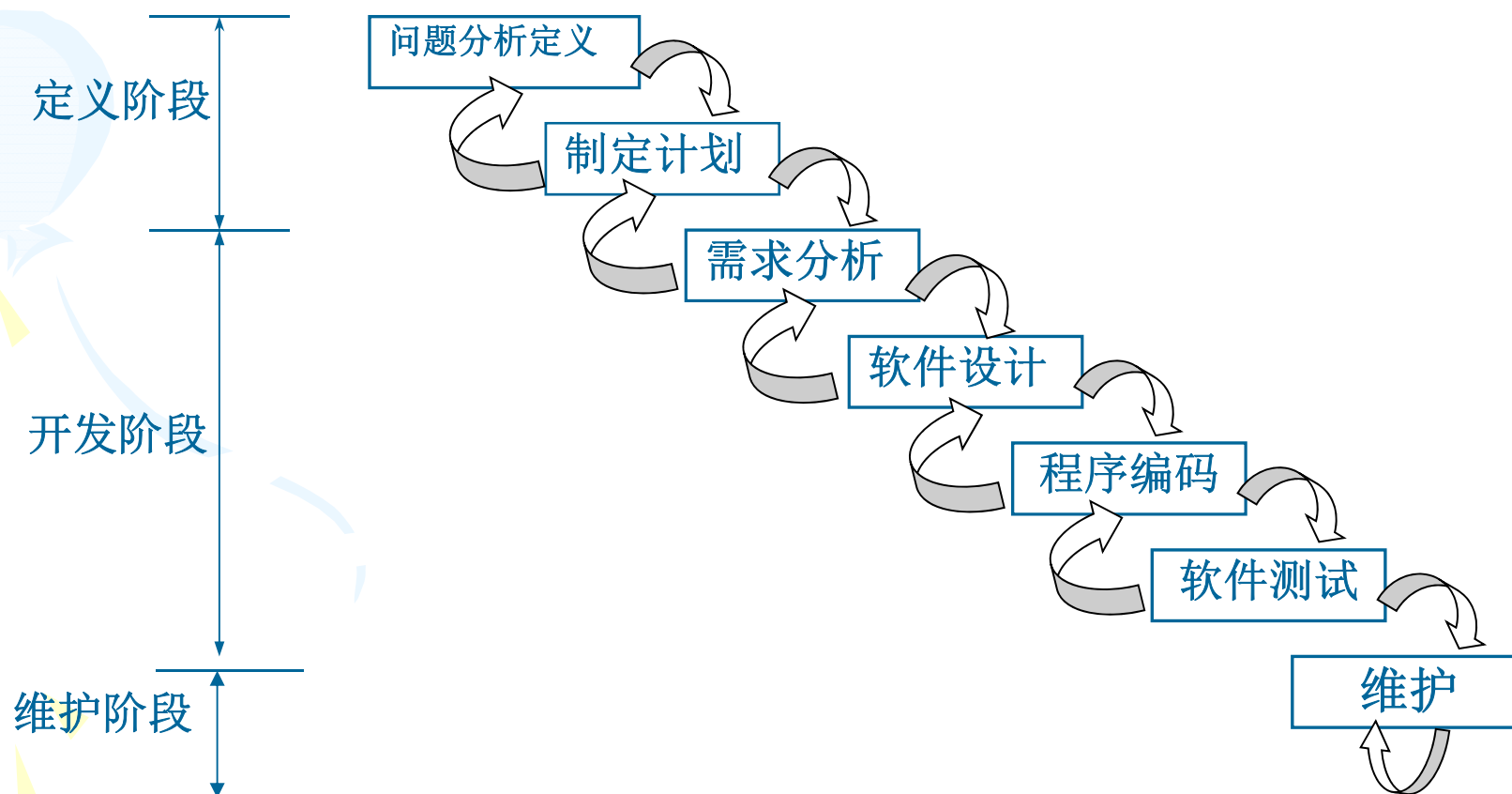
- 验证解决方案



1.3 软件测试与软件开发过程

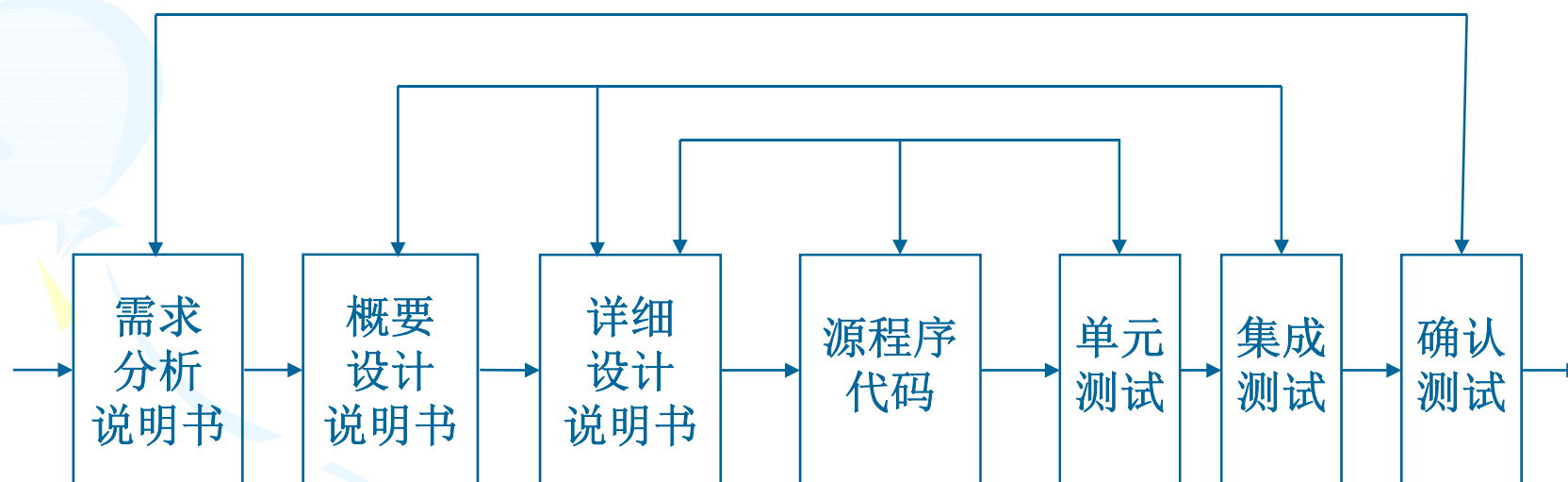
- 一般的软件开发过程

一个软件生命周期包括制定计划、需求分析定义、软件设计、程序编码、软件测试、软件运行、软件维护、软件停用等8个阶段。



测试与开发各阶段的关系

- 测试与开发各阶段产出之间的关系



软件测试包含计划与设计测试，以及执行测试多个阶段
相关测试基础获得后，可以开始计划与设计测试

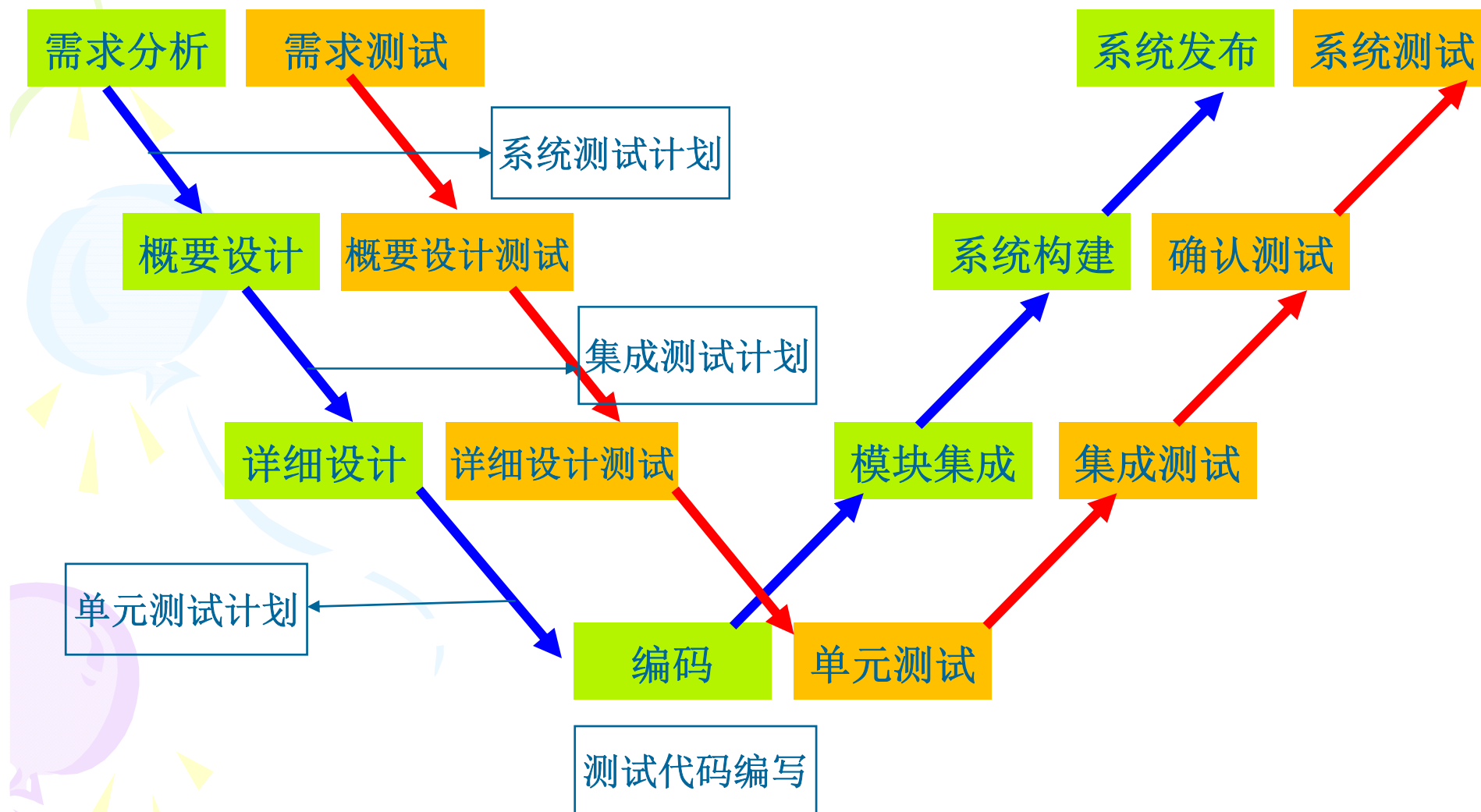


测试与开发各阶段的关系

测试在各开发阶段的任务：

- 项目规划阶段
 - 完成从单元测试到系统测试的整个测试阶段的规划。
- 需求分析阶段
 - 进行测试需求分析、制定系统测试计划；需求本身的测试。
- 详细设计和概要设计阶段
 - 制定集成测试计划和单元测试计划；详细、概要设计本身的测试。
- 编码阶段
 - 由开发人员编写自己负责部分的单元测试代码。在项目较大时，由专人进行编码阶段的测试任务。
- 测试阶段（单元、集成、系统测试）
 - 依据测试代码进行测试，并提交相应的测试状态报告和测试结束报告。

软件测试过程W模型



测试阶段的代价

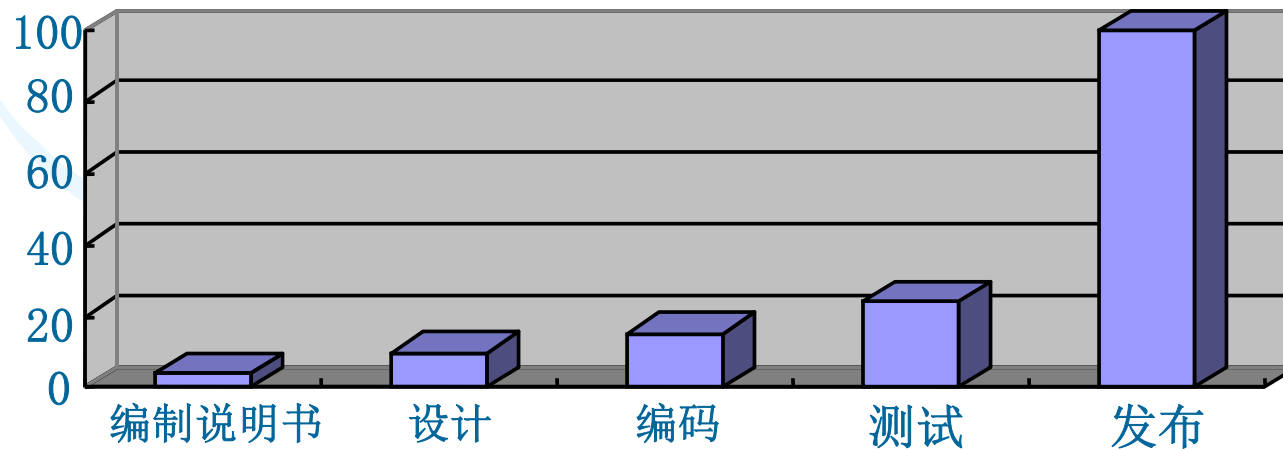
- 软件开发最大的成本是检测、修正软件错误的成本
 - 软件开发中，测试工作量一般 30%~40%，甚至 $\geq 50\%$
 - 人命关天的软件(如飞机控制，核反应堆等)测试所花费的时间往往是其它软件工程活动时间之和的三到五倍

软件工程各阶段工作量

阶段	需求分析 (Requirement Analysis)	设计 (Design)	编码 (Coding)	测试 (Testing)	运行和维护 (Run and Maintenance)
工作量	20%	15%	20%	45%	

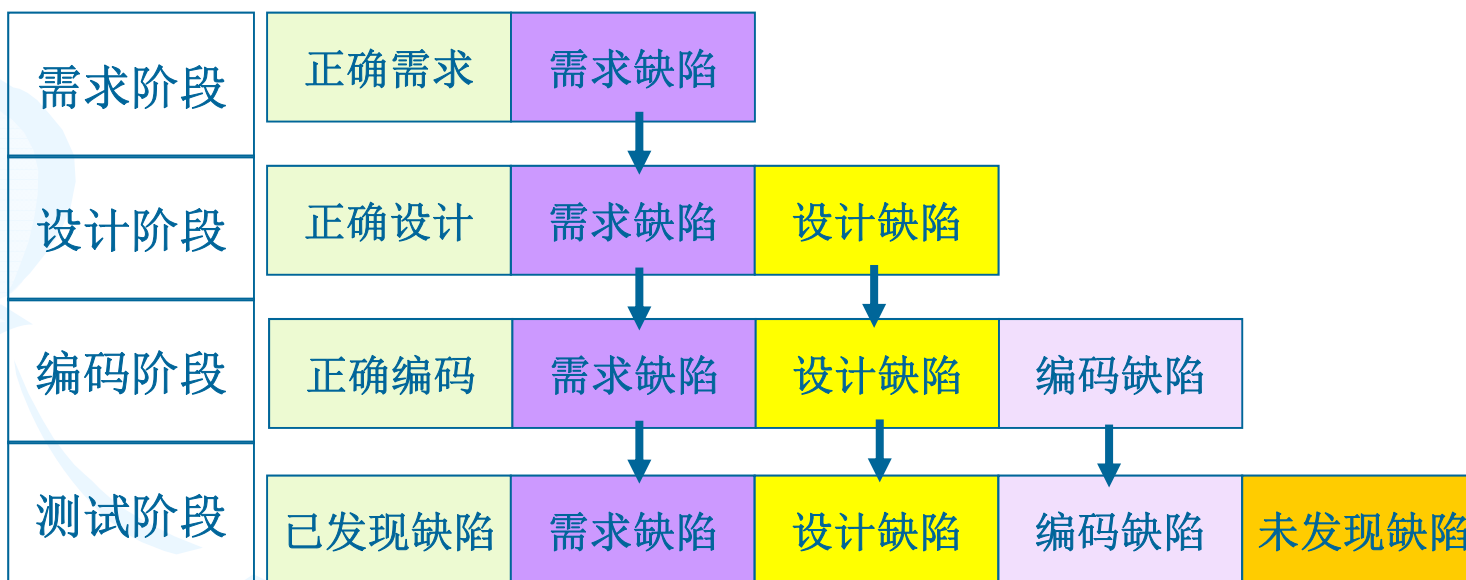
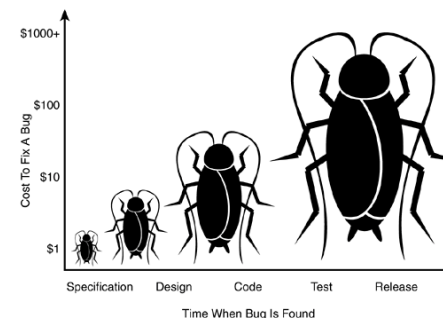
缺陷修复的代价

- 软件在从需求、设计、编码、测试一直到交付用户公开使用后的过程中，都有可能产生和发现缺陷。
- 随着整个开发过程的时间推移，更正缺陷或修复问题的费用呈几何级数增长。



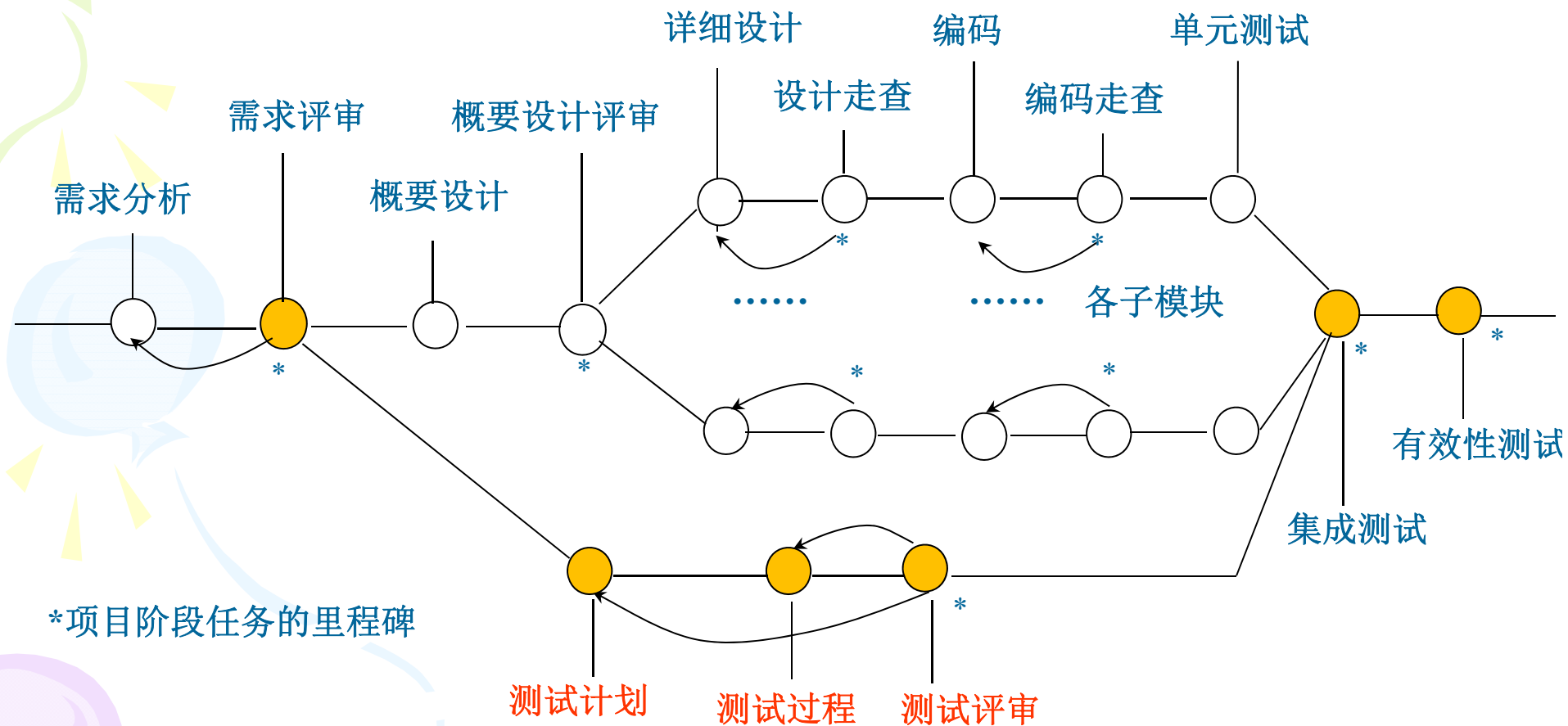
软件缺陷在不同阶段发现时修复的费用示意图

缺陷的不断积累与放大效应



缺陷如何是软件开发成本增加

测试与开发的并行性



软件测试不仅仅时运行测试，还包括各项前期准备



第1章 软件测试概述

1.1 软件测试的发展

1.2 软件测试基础

1.3 软件测试与软件开发过程

1.4 软件测试过程



1.4 软件测试过程

- 制定测试计划
- 测试执行过程



1.4 软件测试过程—制定测试计划

- 本阶段的主要工作内容
 - 对需求规格说明书的仔细研究
 - 将要测试的产品分解成可独立测试的单元
 - 为每个测试单元确定采用的测试技术
 - 为测试的下一个阶段及其活动制定计划
- 制定计划包括：
 - (1) 概要测试计划
 - (2) 详细测试计划



1.4 软件测试过程—制定测试计划

- 测试大纲（用例）

- 测试大纲是软件测试的依据，包括测试项目、测试步骤、测试完成的标准。

- 测试大纲的本质

- 从测试的角度对被测对象的功能和各种特性的细化和展开。

- 测试大纲的好处

- 保证测试功能不被遗漏，也不被重复测试
 - 合理安排测试人员
 - 使得软件测试不依赖于个人

测试大纲示例

Microsoft Excel - Purple Dinosaur Test Tracking.xls					
File Edit View Insert Format Tools Data Window Help					
	A	B	C	D	E
1	Purple Dinosaur Test Tracking				
2	Test Suite /Cases	Test Pass 10/15/1997	Test Pass 11/30/1997	Test Pass 1/5/1998	Bug ID List
3	Basic Hardware Functionality				
4	Left Arm Motion	Pass	Pass	Pass	
5	Right Arm Motion	Pass	Pass	Pass	
6	Head Motion	Fail	Pass	Pass	12
7	Touch Sensors	Pass	Pass	Pass	
8	Peek-a-Boo Sensor	Pass	Pass	Pass	
9	PC Radio Transmission	Fail	Fail	Pass	19, 22
10	PC Radio Reception	Pass	Pass	Pass	
11	TV Radio Transmission	Pass	Pass	Pass	
12	TV Radio Reception	Pass	Pass	Pass	
13	Summary	FAIL	FAIL	PASS	
14					
15	Basic Software Functionality				
16	Songs	Pass	Pass	Pass	
17	Games	Fail	Pass	Pass	13
18	Peek-a-Boo	Pass	Pass	Pass	
19	Cleanup Song	Pass	Pass	Pass	
20	Timeout Sleep	Pass	Pass	Pass	
21	Commanded Sleep	Pass	Pass	Pass	
22	VCR Broadcast Mode	Pass	Fail	Fail	14, 29
23	PC Single Unit Mode	Pass	Pass	Pass	
24	Summary	FAIL	FAIL	FAIL	
25					
Test Case Summary					



1.4 软件测试过程—测试执行过程

- 三个阶段


- (1) 初测期

- 测试主要功能和关键的执行路径，排除主要障碍。

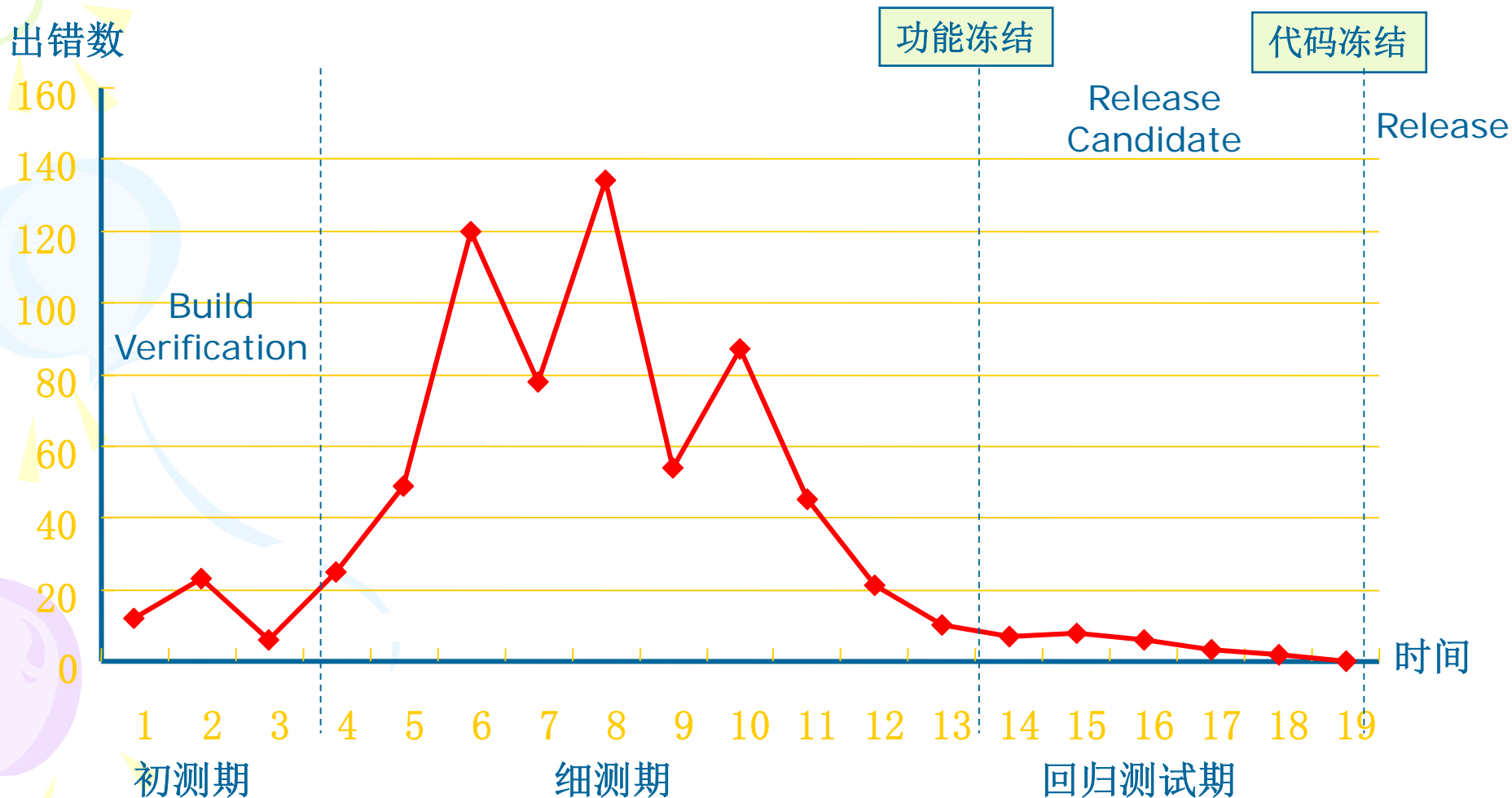
- (2) 细测期

- 依据测试计划和测试大纲、测试用例，逐一测试大大小小的功能、方方面面的特性、性能、用户界面、兼容性、可用性等等；预期可发现大量不同性质、不同严重程度的错误和问题。

- (3) 回归测试期

- 系统已达到稳定，在一轮测试中发现的错误已十分有限；复查已知错误的纠正情况，确认未引发任何新的错误时，终结回归测试。
- 

1.4 软件测试过程—测试执行过程



1.4 软件测试过程—测试执行过程

测试过程中的两个重要里程碑

- 功能冻结(**Function/Feature Freeze**)
 - 经过测试, 符合设计要求, 确认系统功能和其他特性均不再做任何改变。
- 代码冻结(**Code Freeze**)
 - 理论上, 在无错误时冻结程序代码, 但实际上, 代码冻结只标志系统的当前版本的质量已达到预期的要求, 冻结程序的源代码, 不再对其做任何修改。这个里程碑是设置在软件通过最终回归测试之后。

功能冻结和代码冻结 — 界定出了回归测试期的起止界限



1.4 软件测试过程—测试执行过程

- 测试停止的依据(充分性标准)
 - 测试超过了预定时间
 - 执行了所有的测试用例，但并没有发现故障
 - 使用特定的测试用例设计方案作为判断测试停止的基础。
 - 正面指出停止测试的具体要求，即停止测试的标准可定义为查出某一预订数目的故障。
 - 根据单位时间内查出故障的数量决定是否停止测试。



1.4 软件测试过程—测试执行过程

- 测试报告

- 记录问题发生的环境

- 如：各种资源的配置情况

- 记录问题的再现步骤

- 记录问题性质的说明

- 记录问题的处理进程

- 问题处理进程从一定角度上反映测试的进程和被测软件的质量状况以及改善过程。