软件工程概论
Software Engineering

刘伟

liuwei@xidian.edu.cn

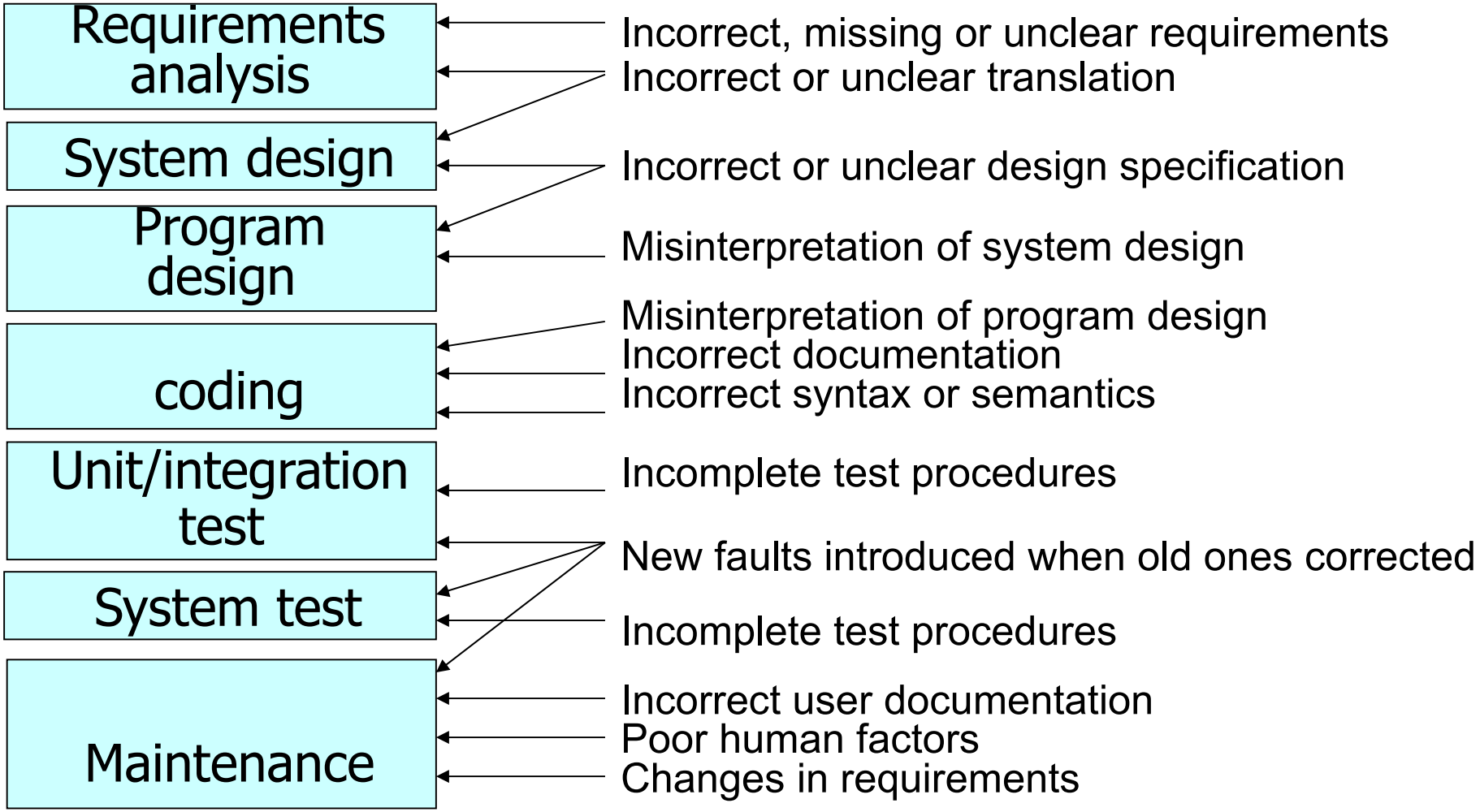
88204608

CH9. Testing the Systems

Content

- 系统测试的原则
- 功能测试
- 性能测试
- 可靠性、可用性和可维护性
- 验收测试
- 安装测试
- 自动化系统测试
- 测试文档

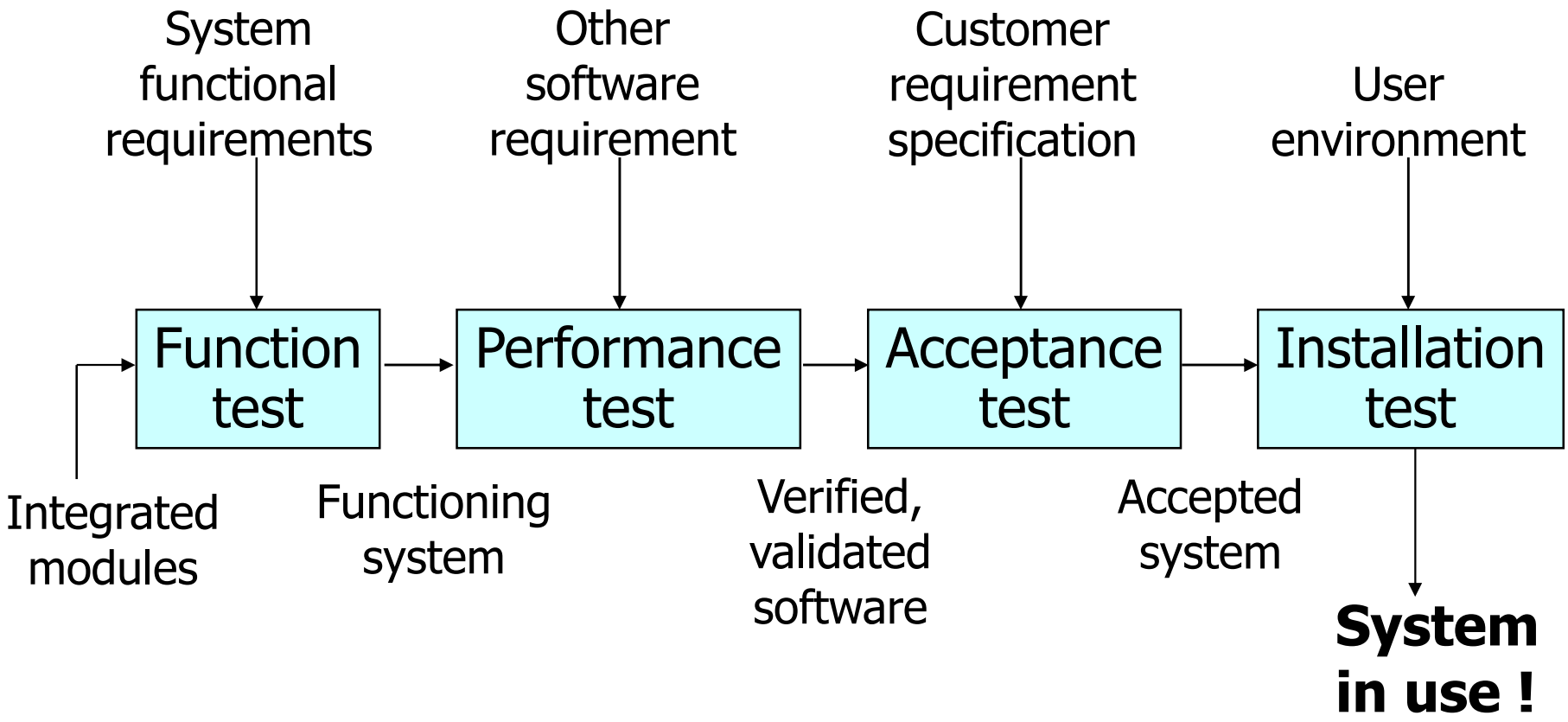
Sources of Software Faults软件错误之源



System testing process系统测试过程

- Function testing功能测试: does the integrated system perform as promised by the requirements specification系统能按需求规格说明的要求运行吗?
- Performance testing性能测试: are the non-functional requirements met是需求说明中的非功能需求?
- Acceptance testing验收测试: is the system what the customer expects是客户所希望的系统?
- Installation testing安装测试: does the system run at the customer site(s)系统能运行在客户的环境吗?

System Testing Process



Techniques used in system testing

- Build or spin plan for gradual testing构建或螺旋化渐进测试计划
- Configuration management配置管理
  - versions and releases版本和发布
  - production system vs. development system产品系统与开发系统
  - deltas, separate files and conditional compilation差别文件、单独文件和条件编译
  - change control变动控制
- Regression testing回归测试

Test team测试小组

- Professional testers专业测试人员：organize and run the tests组织并执行测试
- Analysts分析人员：who created requirements创建需求定义和规格说明的人
- System designers系统设计人员：understand the proposed solution了解所提议的解决方案
- Configuration management specialists配置管理专家：to help control fixes帮助控制修改
- Users用户：to evaluate issues that arise评估出现的问题

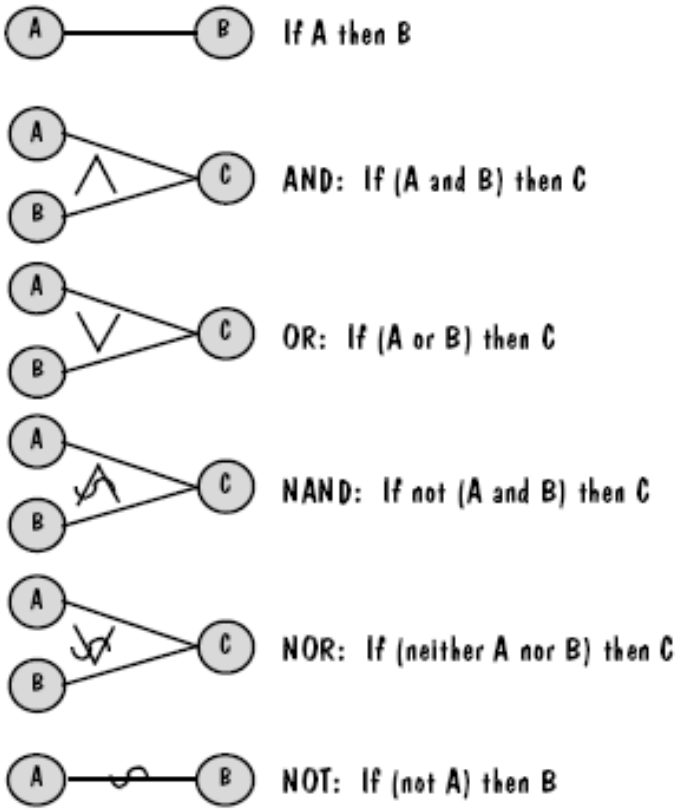
Function Testing功能测试

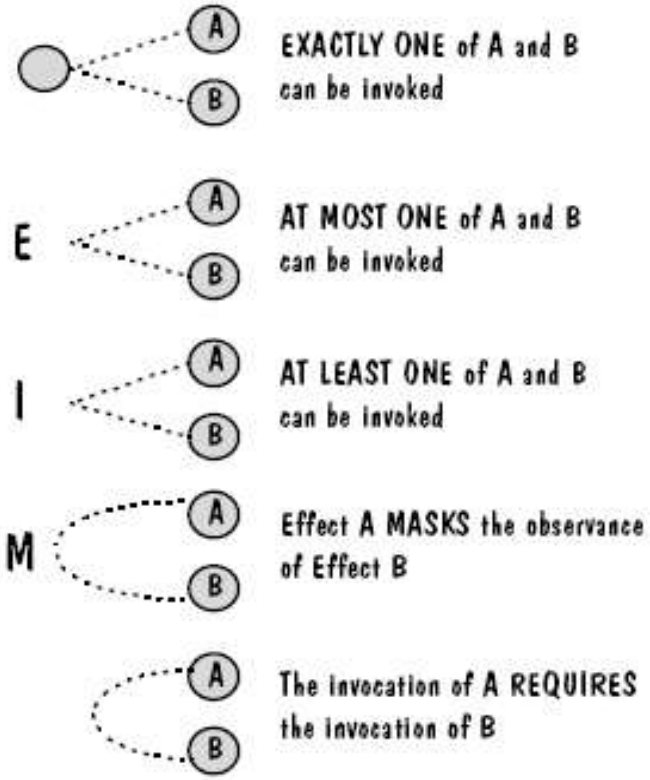
- Thread testing线程测试？？
- Guidelines for function testing功能测试指导
  - Have a high probability of detecting a fault以高概率检测出错误
  - Use a test team independent of designers and programmers使用与设计人员和编程人员无关的测试小组
  - Know the expected actions and output知道期望的活动与输出
  - Never modify the system just to make testing easier决不为了测试容易而去修改系统
  - Have stopping criteria具有停止测试标准

Cause-and-Effect Graphs因果图

- Describe logical relationships between inputs and outputs or between inputs and transformations.描述输入与输出之间或输入与转化之间的逻辑关系
- Cause — input — left node
- effect — output — right node

Notation（因果图符号）





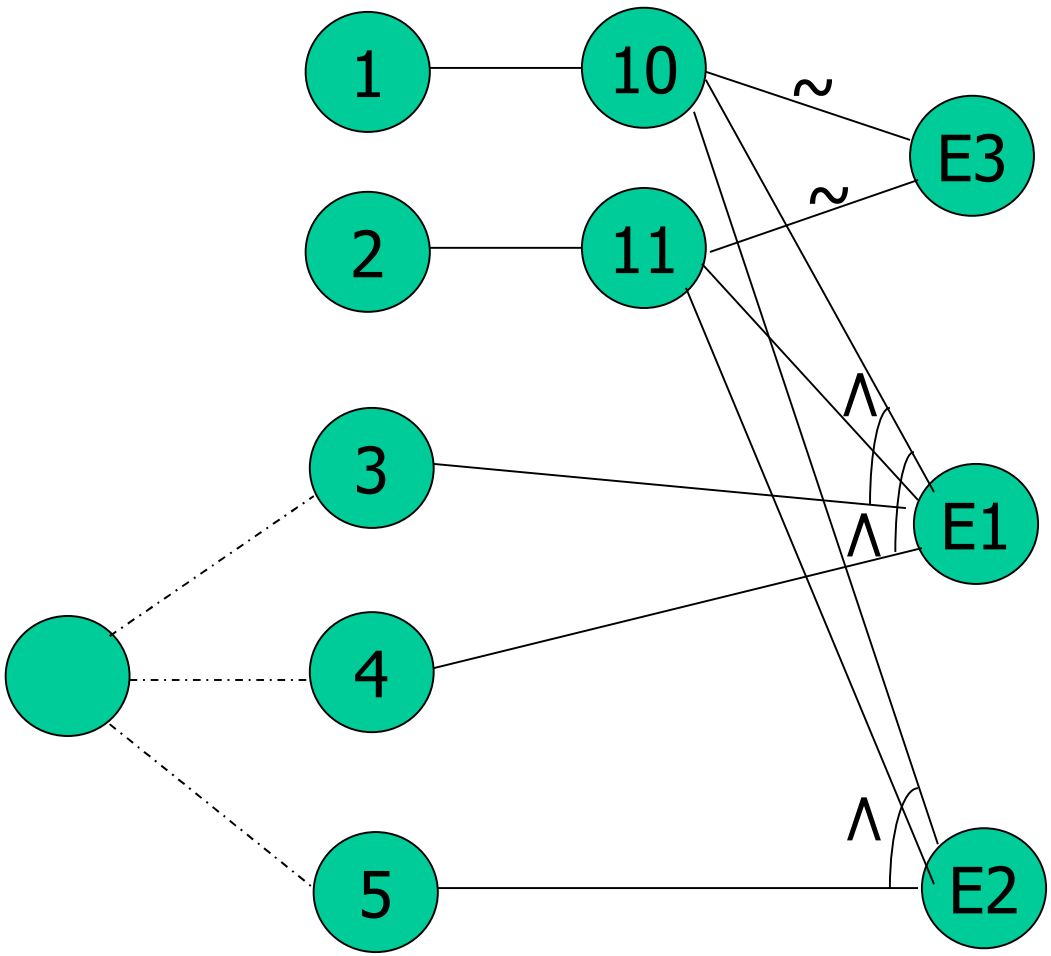
因果图示例1

- 水文监控系统中某一功能的需求：
  - The system sends a message to the dam operator about the safety of the lake level.系统向大坝操作人员发送有关湖泊水位安全性的消息
- Input: 函数LEVEL(A, B)
  - A为坝上游水位；B为24小时内的降雨量
- Processing: 计算函数LEVEL(A, B)的值
- Output :显示下列信息之一
  - “LEVEL = SAFE”
  - “LEVEL = HIGH”
  - “INVALID SYNTAX”

因果图示例2

- Separate the requirement into five “cause”这些需求分成5种“原因”
  - 函数名为 “LEVEL”
  - 参数体 “(A, B)”
  - 函数值= LOW
  - 函数值= SAFE
  - 函数值= HIGH
- Three “effects”三种“结果”
  - 显示 “LEVEL = SAFE”
  - 显示 “LEVEL = HIGH”
  - 显示 “INVALID SYNTAX”

因果图示例



① 函数名是否为“LEVEL”
② 参数体“(A, B)”
③ 函数值= LOW
④ 函数值= SAFE
⑤ 函数值= HIGH

10. 命令是合法的
11. 操作数是合法的

E1:显示“LEVEL = SAFE”
E2:显示“LEVEL = HIGH”
E3:显示“INVALID SYNTAX”

因果图的判定表

Table 9.2. Decision table for cause-and-effect graph.

	Test 1	*Test 2*	*Test 3*	*Test 4*	*Test 5*
Cause 1	I	I	I	S	I
Cause 2	I	I	I	X	S
Cause 3	I	S	S	X	X
Cause 4	S	I	S	X	X
Cause 5	S	S	I	X	X
Effect 1	P	P	A	A	A
Effect 2	A	A	P	A	A
Effect 3	A	A	A	P	P

I -- cause is invoked or true原因被调用或为真

S – cause is suppressed or false原因被禁止或为假

A: Absent（结果）未出现;

P : Present （结果）出现

2^5= 32, however, with the help of cause-and-effect graph, we reduces the number.

Performance tests

- Stress tests强度测试
- Volume tests容量测试
- Configuration tests配置测试
- Compatibility tests兼容性测试
- Regression tests回归测试
- Security tests安全性测试
- Timing tests计时测试
- Environmental tests
- Quality tests
- Recovery tests
- Maintenance tests
- Documentation tests
- Human factors (usability) tests

Reliability, Availability, and Maintainability Definitions

- Software Reliability (R)软件可靠性
  - Probability that the system will operate without failure for a given period of time系统在给定的时间间隔内无失效运行的概率
- Software Availability (A)软件可用性
  - Probability that a system is operating successfully at a given point in time系统在一个给定时刻完整地工作的可能性
- Software Maintainability (M)软件可维护性
  - Probability that a maintenance activity can be carried out within a stated time interval系统在指定的时间间隔内完成维护活动的可能性

几个公式

- MTTF (Mean Time to Failure)平均失效等待时间
- MTTR (Mean Time to Repair)平均修复时间
- MTBF (Mean Time between Failures)平均失效间隔时间
- MTBF=MTTF+MTTR
- R=MTTF/(1+MTTF) (可靠性)
- A=MTBF/(1+MTBF) (可用性)
- M=1/(1+MTTR) (可维护性)

Acceptance tests验收测试

- Benchmark test基准测试：由实际的用户或测试系统功能的某个专门小组进行
- Pilot test试验性测试：install on experimental basis在试验的基础上安装系统
- Alpha test：in-house test内部测试
- Beta test：customer pilot客户测试
- Parallel testing并行测试：new system operates in parallel with old system新老系统并行运作

安装测试

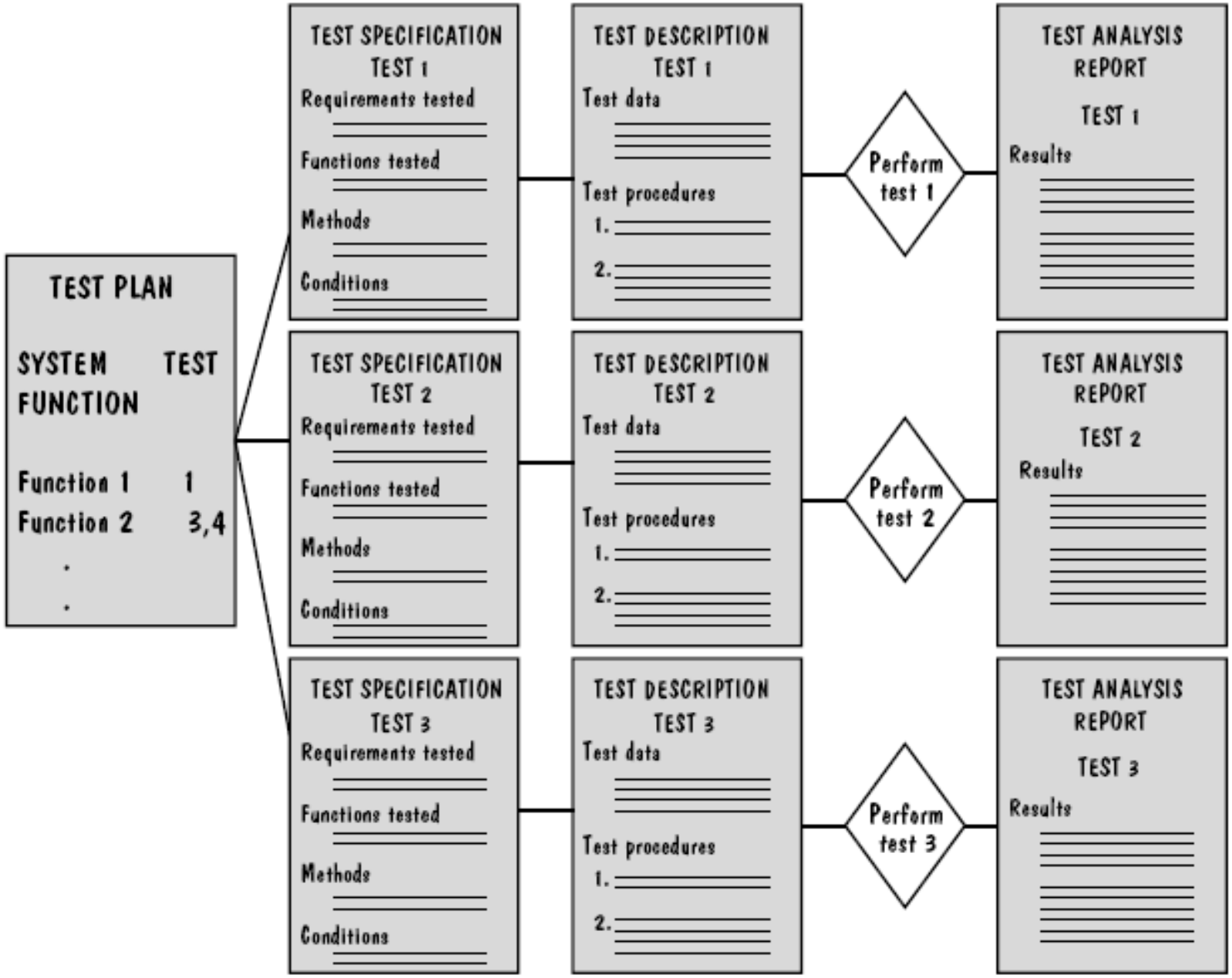
- 一般情况下，如果进行了验收测试则不需要进行安装测试；
- 但如果在验收测试时，其环境与真实的用户环境存在差距，则安装测试是必要的。
- 还有一种可能，就是软件安装的环境可能存在较大差异，软件厂商提供了详细的安装文档，因此需要对安装文档的正确性，以及系统的安装进行详细的测试。
  - 例如oracle数据库同时提供了对多个操作系统的支持，因此需要对其在多个操作系统之上的安装进行详细的测试。

自动化系统测试

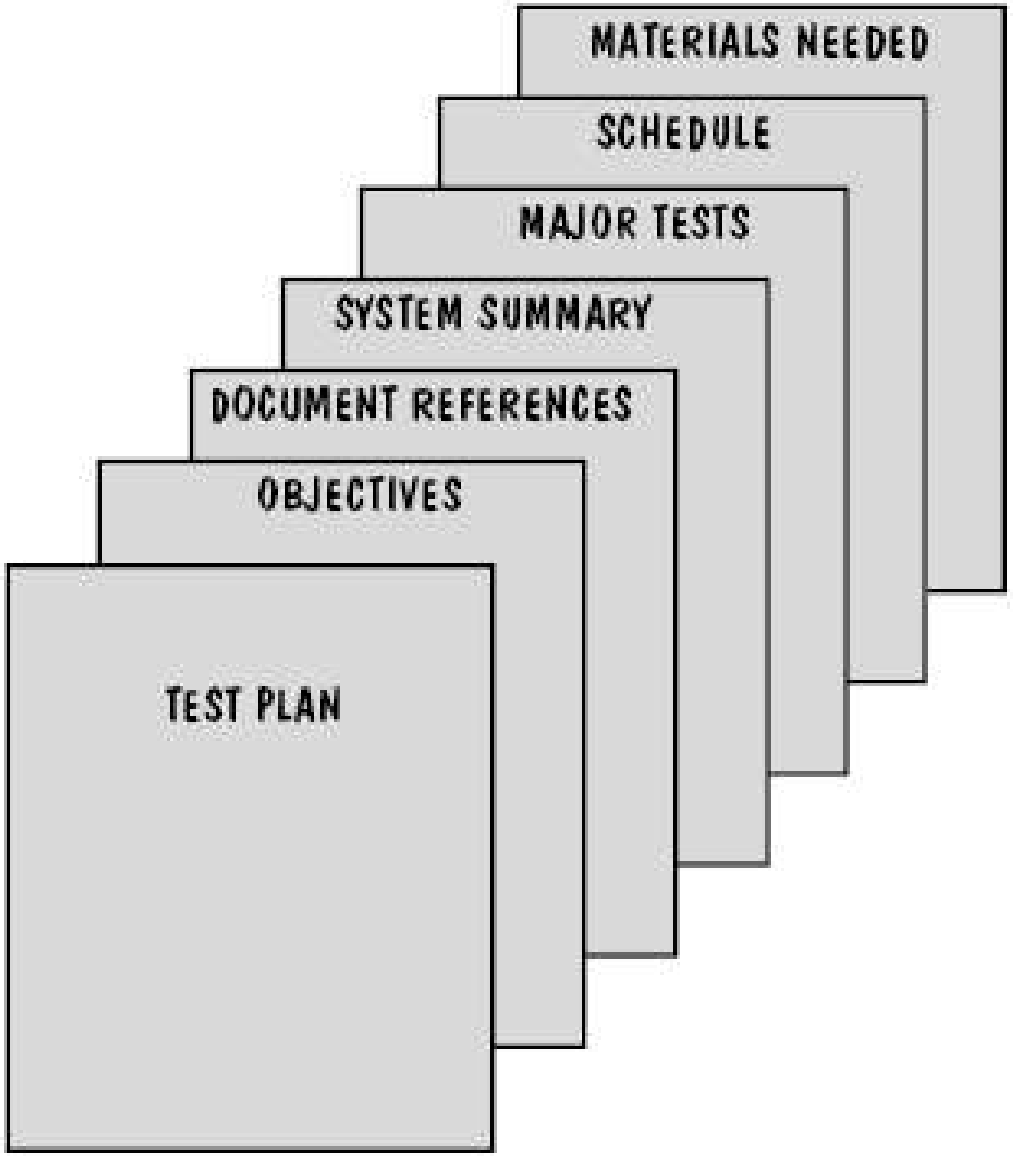
- 自动化系统测试主要是将系统运行起来之后，采用各种专用测试软件对系统的外部特征进行测试，测试的内容包括：
  - 性能测试，例如对系统的吞吐量、并发能力、交易响应时间等进行测试；
  - 安全测试，例如可以采用各种攻击手段，对应用系统进行攻击，以检测应用系统的安全性；
  - 可恢复性测试，认为的创造系统的失败，进行应用程序的恢复性测试

Test documentation测试文档

- Test plan测试计划: describes system and plan for exercising all functions and characteristics描述了系统和测试所有功能与特征的计划
- Test specification and evaluation测试说明和评价: details each test and defines criteria for evaluating each feature详细描述了每个测试以及对（测试中针对的）每个特征定义评价的标准
- Test description测试描述: test data and procedures for each test对每个测试给出了测试数据和过程
- Test analysis report测试分析报告: results of each test描述了每个测试结果


TEST PLAN
SYSTEM FUNCTION
TEST
Function 1
1
Function 2
3,4
TEST SPECIFICATION
TEST 1
Requirements tested
Functions tested
Methods
Conditions
TEST DESCRIPTION
TEST 1
Test data
Test procedures
1.
2.
Perform test 1
TEST ANALYSIS REPORT
TEST 1
Results
TEST SPECIFICATION
TEST 2
Requirements tested
Functions tested
Methods
Conditions
TEST DESCRIPTION
TEST 2
Test data
Test procedures
1.
2.
Perform test 2
TEST ANALYSIS REPORT
TEST 2
Results
TEST SPECIFICATION
TEST 3
Requirements tested
Functions tested
Methods
Conditions
TEST DESCRIPTION
TEST 3
Test data
Test procedures
1.
2.
Perform test 3
TEST ANALYSIS REPORT
TEST 3
Results

测试中产生的文档



一个测试计划的各个部分

测试计划的参考模板

类别	递交时间	建议制定者	建议审批者
《单元测试计划》	在系统设计阶段就可以起草，最迟可在实现阶段之初递交。	开发小组的技术负责人	项目经理
《集成测试计划》	在系统设计阶段就可以起草，最迟可在实现阶段之初递交。	开发小组的技术负责人	项目经理
《系统测试计划》	在需求开发阶段就可以起草，最迟可在开发工作完成之际递交。	独立测试小组的负责人	项目经理
《验收测试计划》	在需求开发阶段就可以起草，最迟可在系统测试工作完成之际递交。	项目经理	客户代表 或上级领导

1. 测试范围与目的
2. 测试方法
3. 测试环境与测试辅助工具的描述
4. 测试启动准则与结束准则
5. 应递交的文档

文档名称	预计递交时间
《测试计划》	
《测试用例》	
《测试报告》	

6. 测试人员的组织
7. 任务表

测试任务	开始时间	结束时间	执行人员

8. 可能存在的问题与对策

- 附录：本计划的审批意见

测试用例

《测试用例》的类别	递交时间	建议设计者	建议审批者
单元测试阶段的《测试用例》	在系统设计阶段就可以起草，最迟可在实现阶段之初递交。	开发小组各成员	项目经理
集成测试阶段的《测试用例》	在系统设计阶段就可以起草，最迟可在实现阶段之初递交。	开发小组各成员	项目经理
系统测试阶段的《测试用例》	在需求分析计阶段就可以起草，最迟可在开发工作完成之际递交。	独立测试小组各成员	项目经理
验收测试阶段的《测试用例》	在需求分析计阶段就可以起草，最迟可在系统测试工作完成之际递交。	项目经理	客户代表或上级领导

测试报告的参考模板

1. 基本信息

测试计划的来源	提示：填写《测试计划书》名称，版本，时间
测试用例的来源	提示：填写《测试用例》名称，版本，时间
测试对象描述	
测试环境描述	
测试驱动程序描述	提示：可以把测试驱动程序当作附件
测试人员	
测试时间	

2. 实况记录

测试用例	测试情况	错误严重程度

3. 分析与建议

提示：对测试结果进行分析，提出建议。

4. 错误修改记录

错误名称	原因	修改人	修改时间	是否回归测试

- 附件

性能测试用例的参考模板

1. 被测试对象的介绍
2. 测试范围与目的
3. 测试环境与测试辅助工具的描述
4. 测试驱动程序的设计
5. 性能测试用例列表

性能 A 描述		
用例目的		
前提条件		
输入数据	期望的性能（平均值）	实际性能（平均值）
性能 B 描述		
……		

……

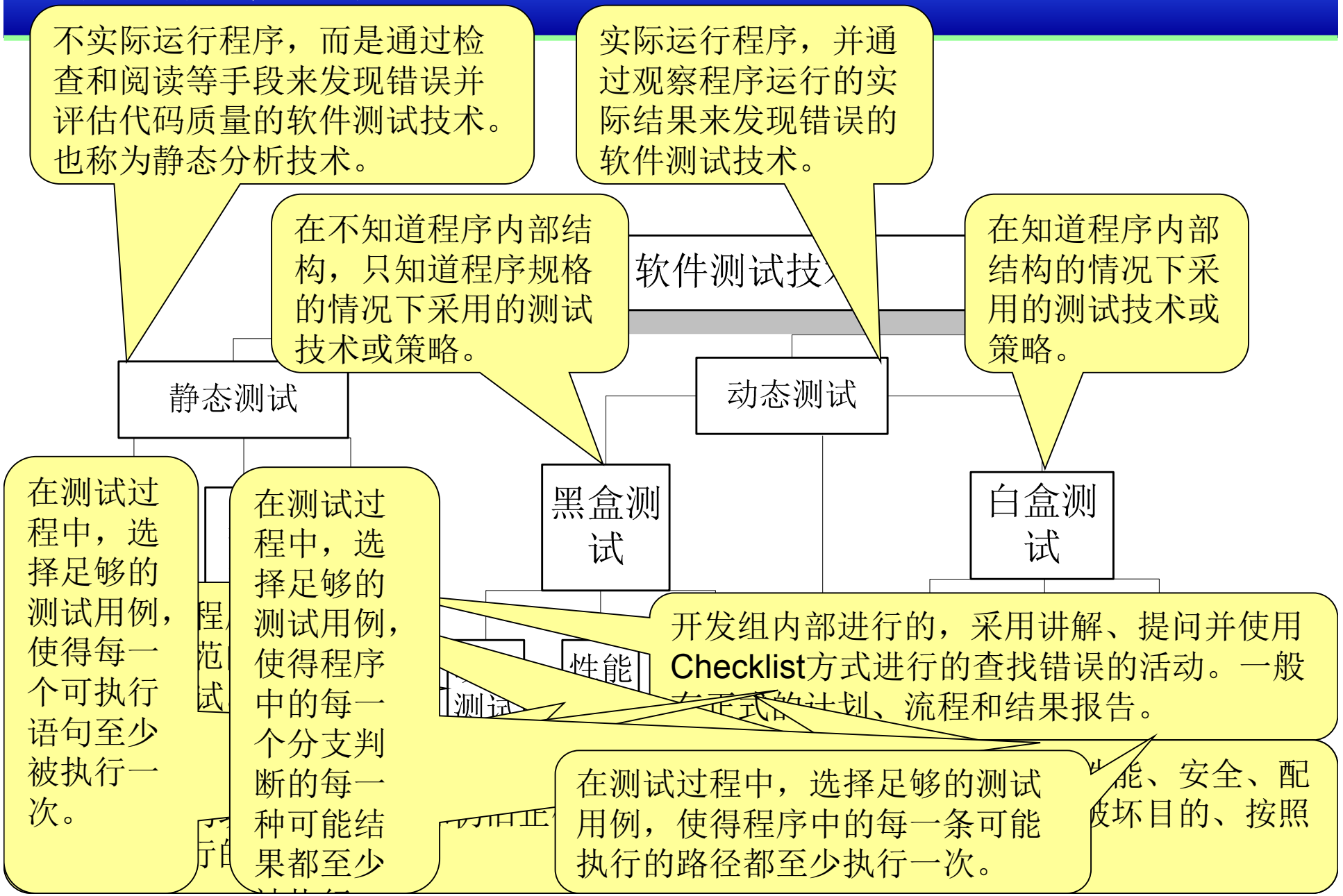
Problem report forms问题报告表

- Location地点：该问题发生在何处？
- Timing计时：它于何时发生？
- Symptom症状：观察到什么？
- End result最终结果：后果是什么？
- Mechanism机制：它是怎样发生的？
- Cause原因：它为什么会发生？
- Severity严重级别：用户或事务被影响的程度怎样？
- Cost成本：它的代价有多大

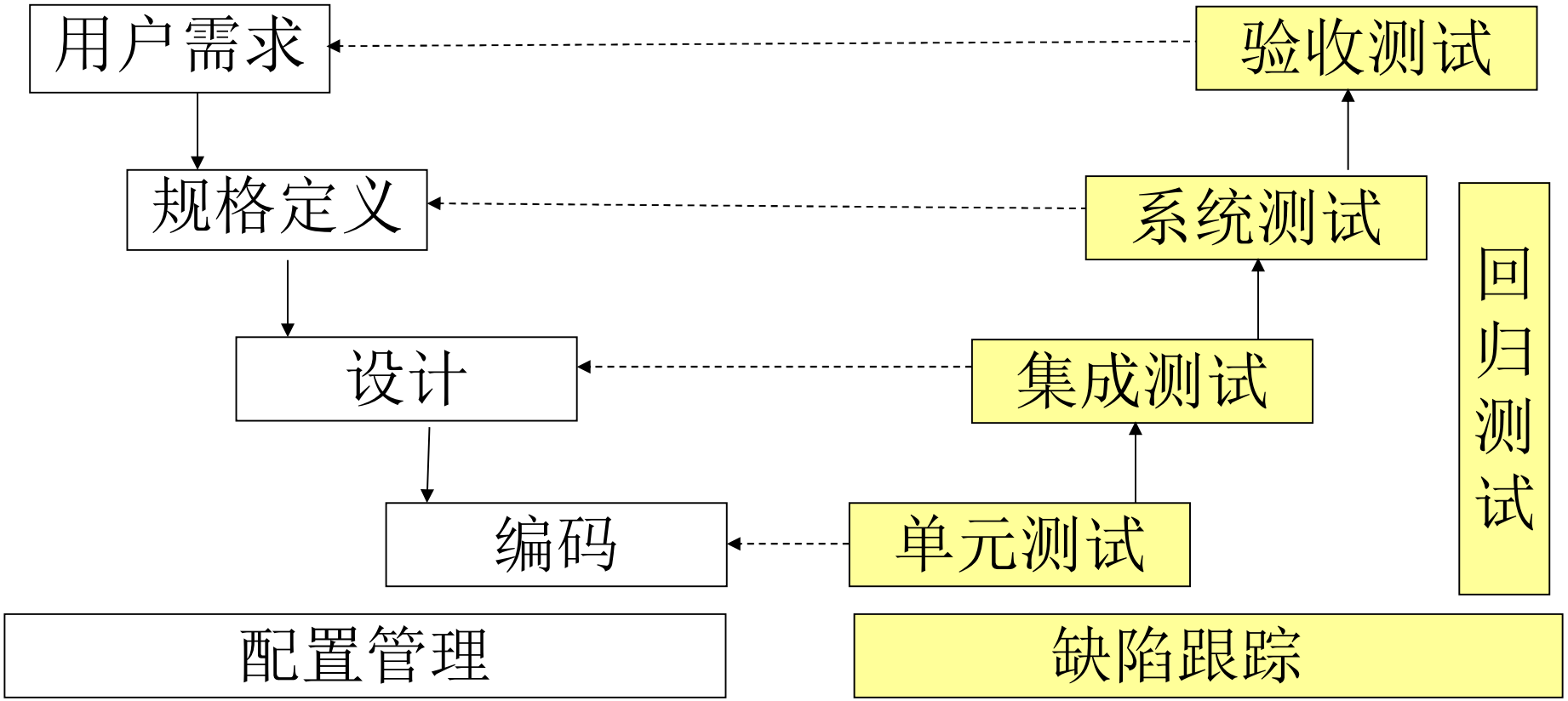
Testing safety-critical systems测试安全攸关的系统

- Design diversity设计多样性: use different kinds of designs, designers使用不同“种类”的设计方案、设计人员
- Software safety cases软件安全因果分析:
  - failure modes and effects analysis失效模式和后果分析（FMEA）
  - hazard and operability studies危险和可操作性研究
- Cleanroom净室方法: certifying software with respect to the specification根据说明证明软件

测试技术小结


不实际运行程序，而是通过检查和阅读等手段来发现错误并评估代码质量的软件测试技术。也称为静态分析技术。
实际运行程序，并通过观察程序运行的实际结果来发现错误的软件测试技术。
在不知道程序内部结构，只知道程序规格的情况下采用的测试技术或策略。
在知道程序内部结构的情况下采用的测试技术或策略。
软件测试技
静态测试
动态测试
黑盒测试
白盒测试
在测试过程中，选择足够的测试用例，使得每一个可执行语句至少被执行一次。
在测试过程中，选择足够的测试用例，使得程序中的每一个分支判断的每一种可能结果都至少
性能
开发组内部进行的，采用讲解、提问并使用Checklist方式进行的查找错误的活动。一般
在测试过程中，选择足够的测试用例，使得程序中的每一条可能执行的路径都至少执行一次。
能、安全、配
破坏目的、按照

测试过程

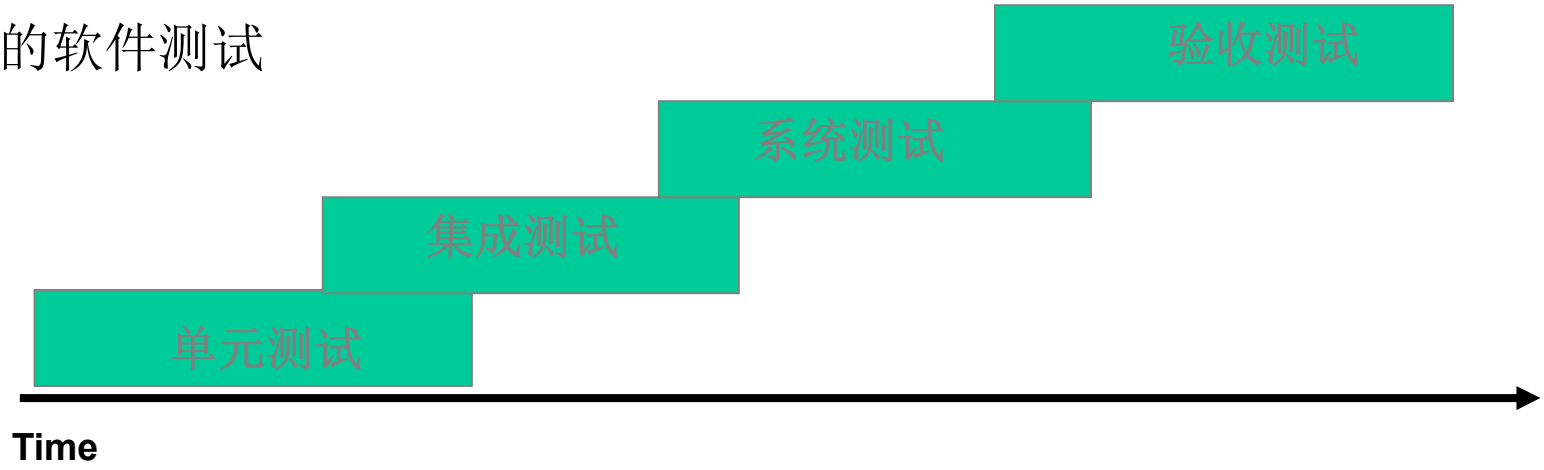


当前软件测试面临的挑战

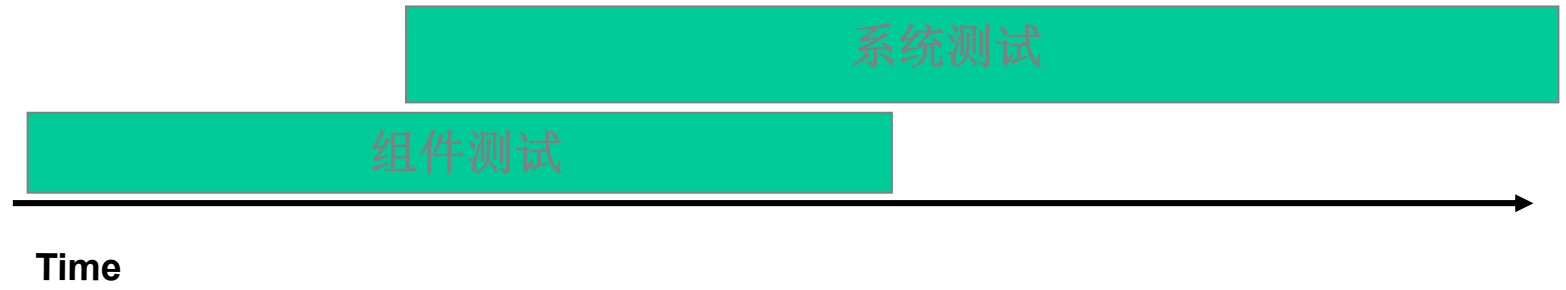
- 如何适应由于软件新技术、新架构的应用导致测试工作量增大？
- 如何进行软件测试工作的分工？
- 如何提高开发团队的进行组件测试的质量？
- 如何提高系统测试团队的士气？
- 如何评价系统测试过程的进度？
- 如何评价系统测试的完备性？
- 如何评价软件质量？

软件测试阶段

传统的软件测试



现代软件测试



组件测试—以测试为驱动的开发

- 目的
  - 尽可能发现早的软件缺陷
  - 保证系统测试的效率
- 测试方法—黑盒和白盒相结合
  - 黑盒测试：基于软件设计规范设计测试用例
  - 白盒测试：基于代码覆盖情况设计测试用例
- 利用Xunit测试框架提高测试用例实现效率
  - http://www.xprogramming.com/
- 利用Coverage工具获得代码覆盖情况

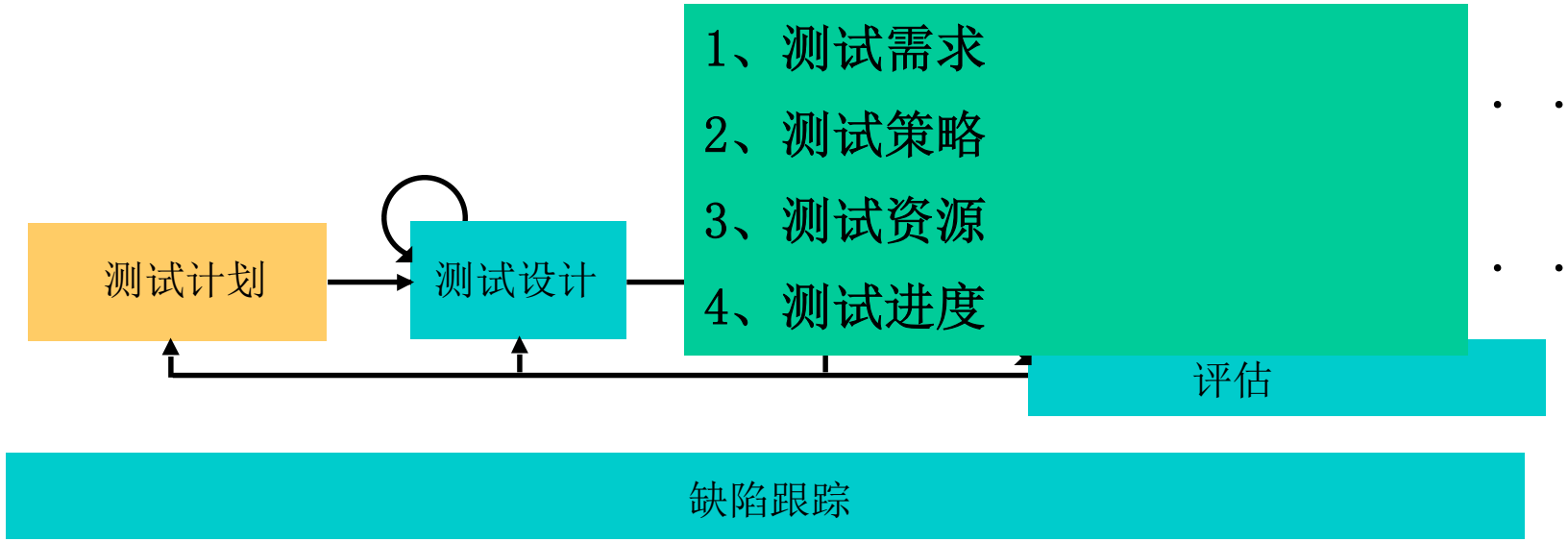
组件测试—Rational解决方案

- 组件测试理念
  - 边开发边测试
  - 减少对软件开发自身的影响
  - 无需学习测试脚本语言
- 软件测试工具
  - Purify：自动定位内存相关错误
  - Quantify：发现程序的性能瓶颈
  - Coverage：发现未被测试的代码
  - Test RealTime：针对嵌入式系统软件组件测试

系统测试

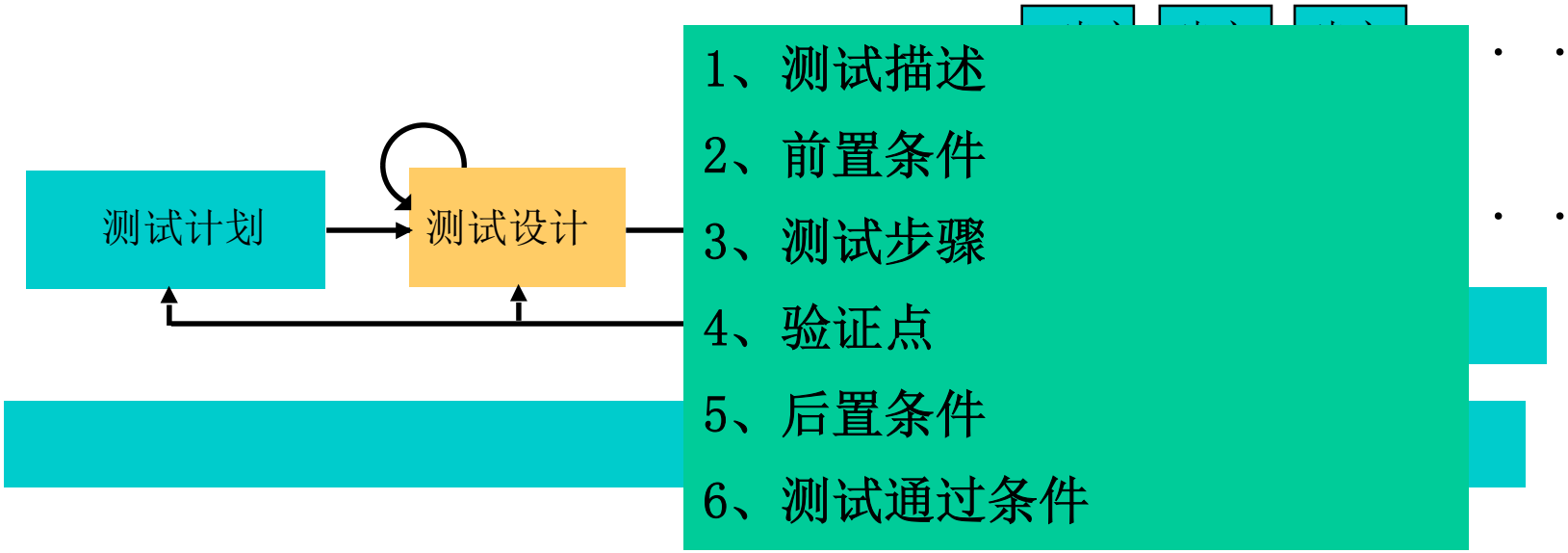
- 主要由测试部门进行
- 为测试工程师提供良好的职业发展道路
  - 测试经理
  - 测试分析员
  - 测试员
- 增强测试团队和开发团队的沟通
- 明确定义并贯彻的测试过程是测试自动化的重要前提

系统测试过程—测试计划



- 输入：软件需求书
- 输出：测试计划书

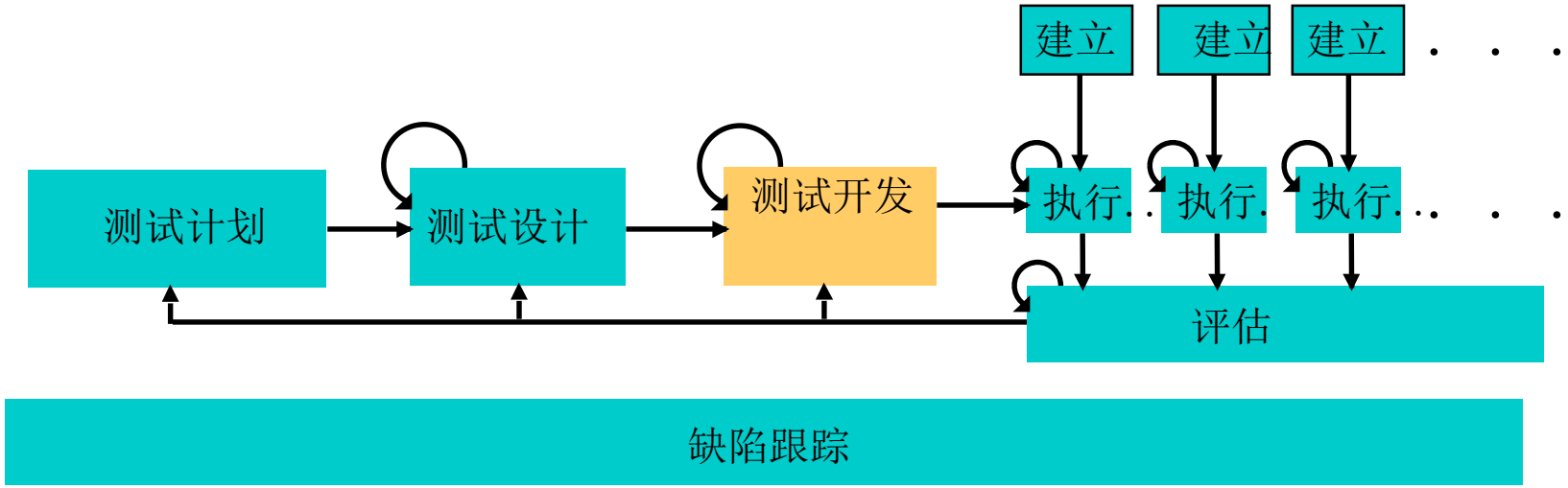
系统测试过程—测试设计



- 输入：软件测试计划书
- 输出：软件测试大纲

良好的测试设计是测试自动化的重要保证！

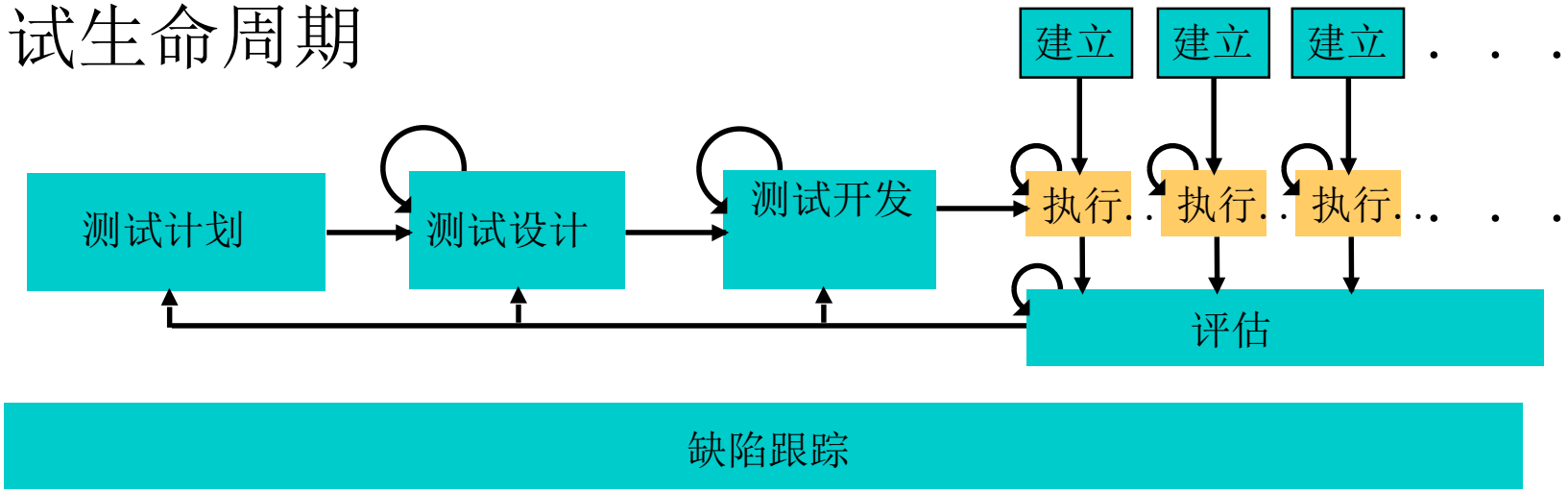
系统测试过程—测试开发



- 测试脚本
  - 手工测试脚本
  - 自动化测试脚本

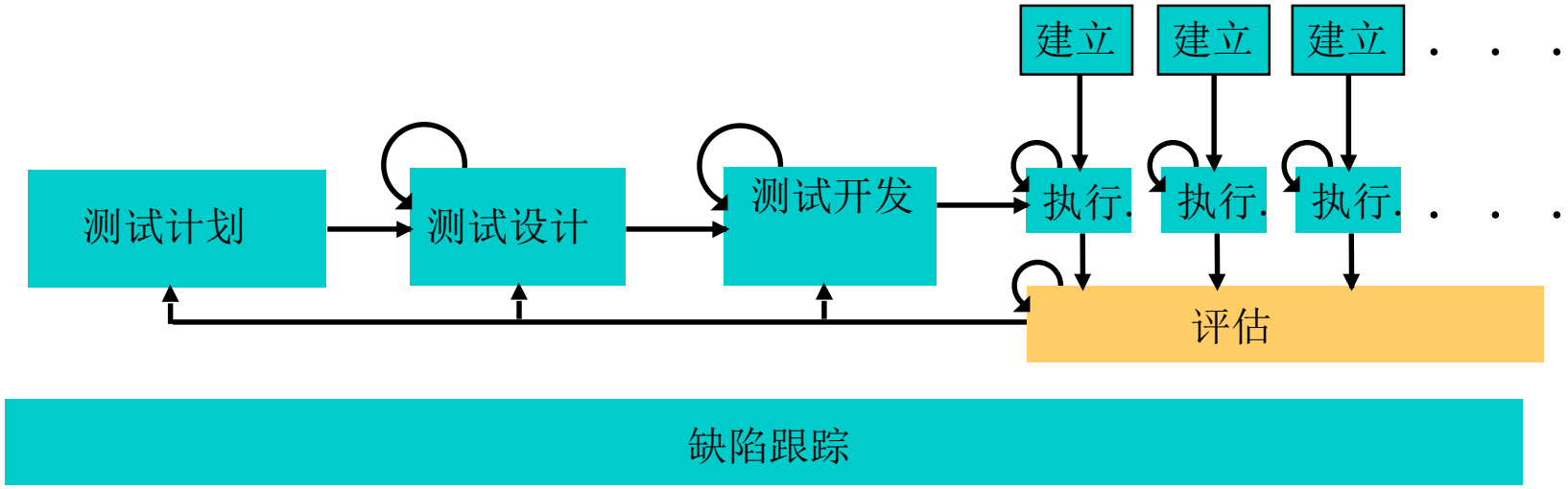
系统测试过程—测试执行

测试生命周期



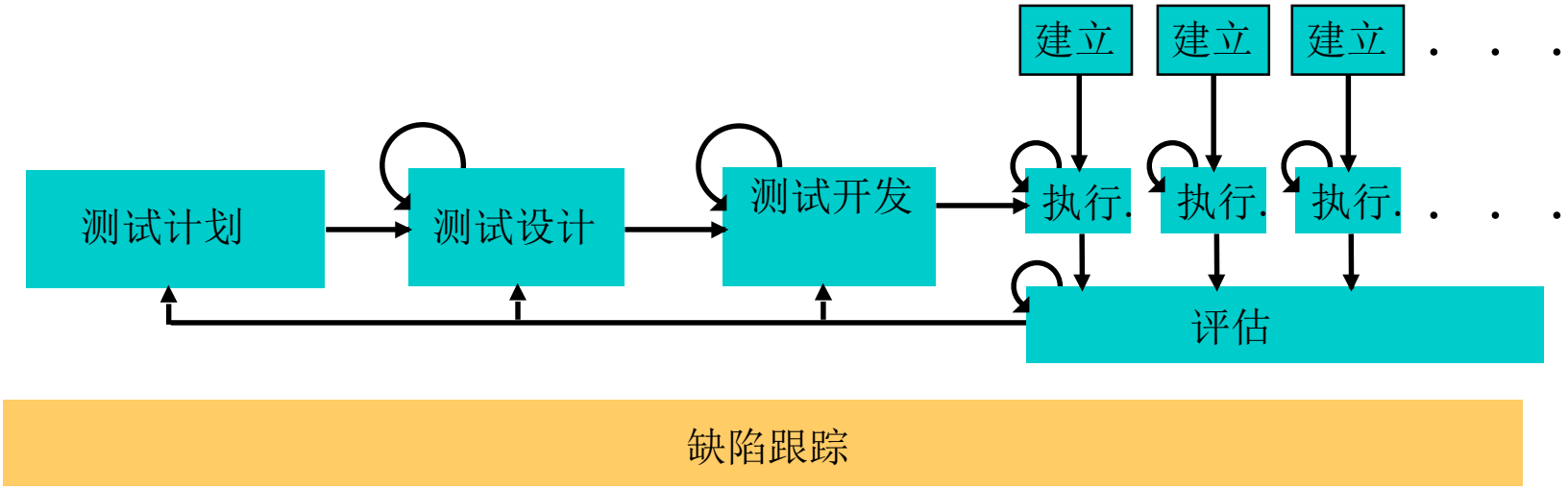
- 测试执行
  - 进行测试执行管理
  - 运行测试
  - 记录测试结果，包括缺陷报告和测试日志

系统测试过程—测试评估



- 测试评估
  - 统计和分析测试结果，确定是否达到软件发布的标准

系统测试过程—缺陷跟踪



- 缺陷跟踪
  - 记录测试发现的缺陷或用户问题，并且跟踪、管理缺陷的状态变更

系统测试度量

- 测试覆盖率：评价测试的完备性
  - 需求覆盖率
  - 代码覆盖率
- 测试报告
  - Defect Density缺陷密度
  - Defect Aging缺陷老化
  - Defect Trend缺陷趋势
- 性能指标
  - 动态监控
  - 响应时间/吞吐量报告
  - 百分比报告

Rational系统测试方案

- 软件工具是完美过程得以成功实施的重要保证
- Rational TeamTest
  - TestManager： 集中、可伸缩的测试管理平台
  - Robot：传统应用自动化测试工具
  - RobotJ：Web/Java自动测试工具
  - ClearQuest：缺陷跟踪工具

Rational系统测试方案