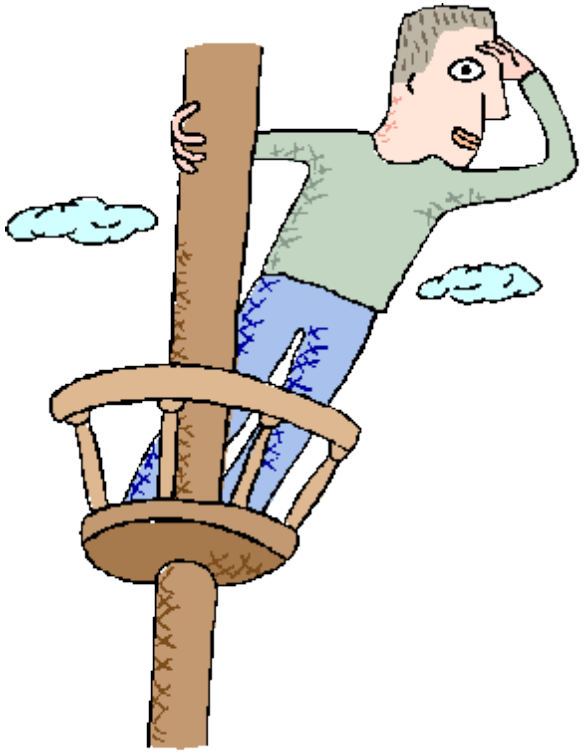
测试执行与监控

国家应用软件质量监督检验中心

课程概览



✓ 在本章中，我们将学习：

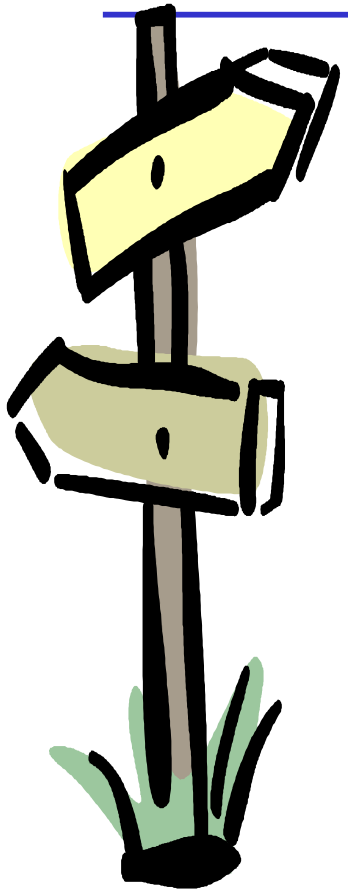
- 测试的执行
- 如何结束测试
- 测试执行的监控

课程目标

✓完成此课程之后，学员将具备以下能力：

- 执行测试用例
- 理解测试执行的监控过程
- 掌握结束测试的条件

课程目录



- ✓执行测试
- ✓监控测试执行
- ✓结束测试执行

测试执行

- ✓测试执行是执行所有或部分选定的测试用例，并对结果进行分析的过程

测试执行的前提

✓ 进入系统测试执行的要求：

- 软件实现基本已经完成
- 所有需求、设计文档均已批准、定稿
- 已经通过BVT测试或Smoke测试，检查主要功能是否能测试，表明该软件具备一定的可靠性，可以开始正式的、全面的测试
- 测试计划、用例设计已完成。测试环境已准备好

测试用例执行过程中的问题

- ✓软件是否有缺陷
- ✓填写软件缺陷报告
- ✓确定造成这些缺陷的原因
- ✓需求、设计是否有缺陷
- ✓测试环境和测试部件是否有缺陷
- ✓测试用例设计是否不合理

测试用例的手动执行

- ✓根据测试用例的要求人工的进行软件操作，输入数据，观测输出结果

测试用例的自动执行

- ✓使用录制回放工具
- ✓使用通用脚本

测试执行过程

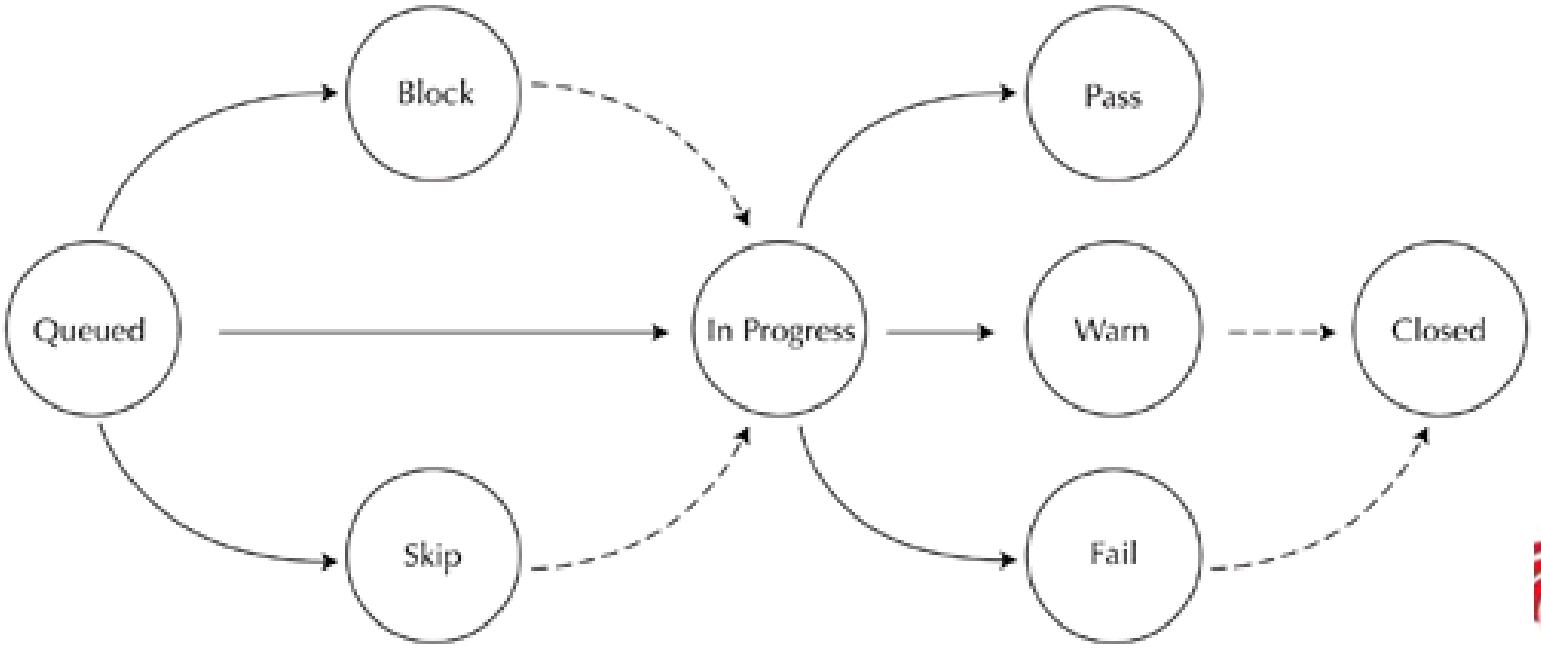
- ✓ 根据测试的阶段、任务，选择执行全部或部分测试用例
- ✓ 任务分配：将测试用例分配给测试工程师
- ✓ 执行测试，记录原始数据，报告发现的缺陷
- ✓ 执行某些测试用例时，如果需要先将被测对象置于某个特定的状态，则应保留测试环境、状态

测试执行过程

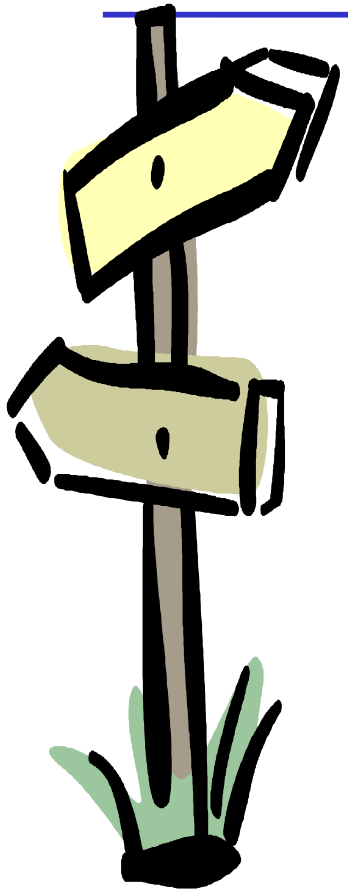
- ✓解决测试中阻碍进度的问题
- ✓向管理层报告测试的进度、发现的主要问题等等

测试执行过程

✓测试用例的状态和生命周期

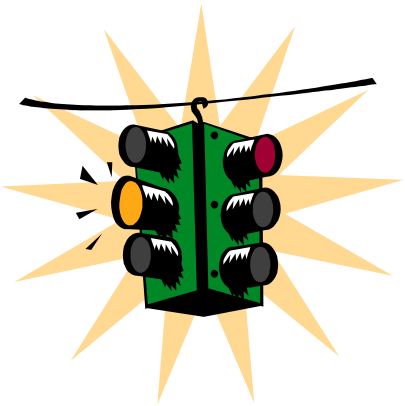


课程目录



- ✓执行测试
- ✓结束测试执行
- ✓监控测试执行

测试执行的结束



- ✓测试执行的结束
  - – 测试达到预期目的后，按计划结束
  - – 受到时间进度、资源的限制，测试被迫结束
- ✓在测试计划中明确说明测试结束的条件
  - – Good-Enough原则
- ✓结束条件的判定是在质量和成本之间的折衷

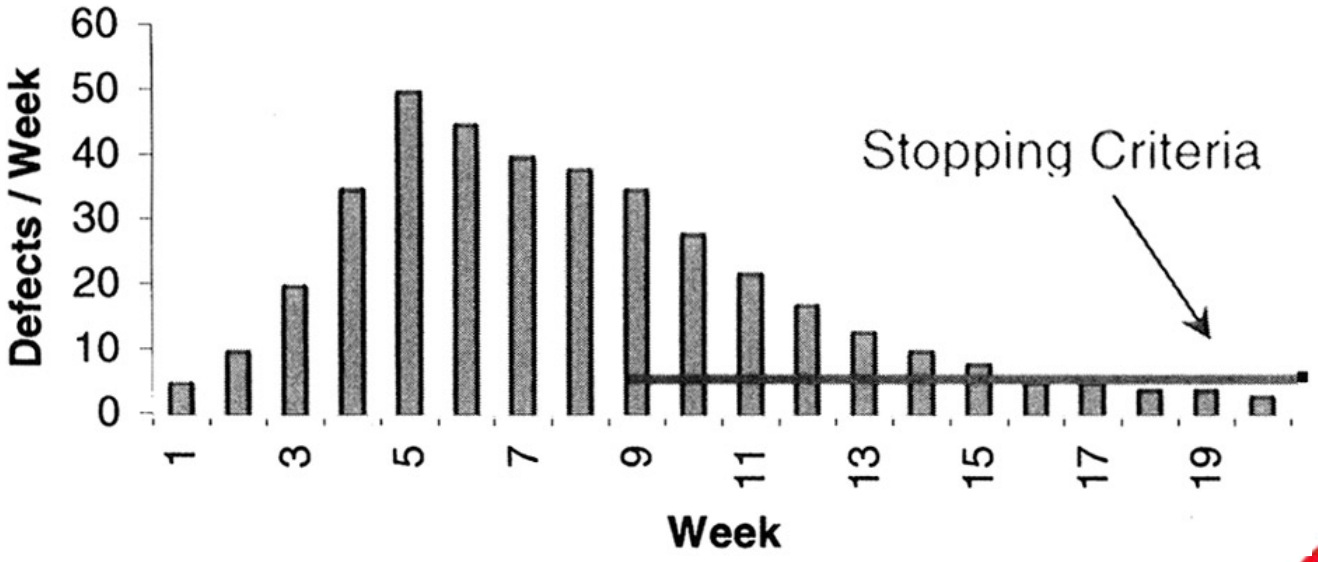
测试执行的结束条件

✓达到了覆盖率的要求

- 单元测试：语句覆盖、 …
- 集成测试：API、API/参数组合
- 系统测试：功能、用例、用例场景(scenario)
- 例如：
  - ＞100%语句覆盖
  - ＞90%用例场景覆盖

测试执行的结束条件

✓指定的时间段内没有发现新的缺陷



测试执行的结束条件

- ✓基于成本的考虑
  - 测试执行到一定阶段时，查找未发现的缺陷的成本逐渐增大，如果超过了由潜在缺陷所引起的代价，则可以停止测试
  - 不适用于要求高可靠性的软件：武器、医学设备、…

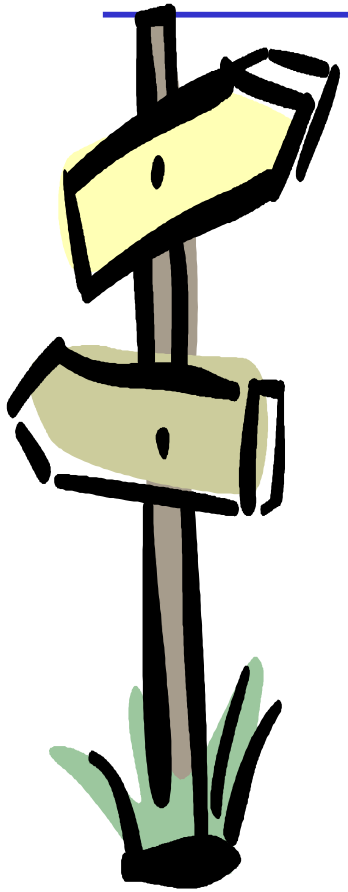
测试执行的结束条件

- ✓项目组达成一致
  - 基于技术、资金、开销等各种因素，项目组（包括管理层、开发、测试、市场销售）意见一致，认为可以停止测试

测试执行的结束条件

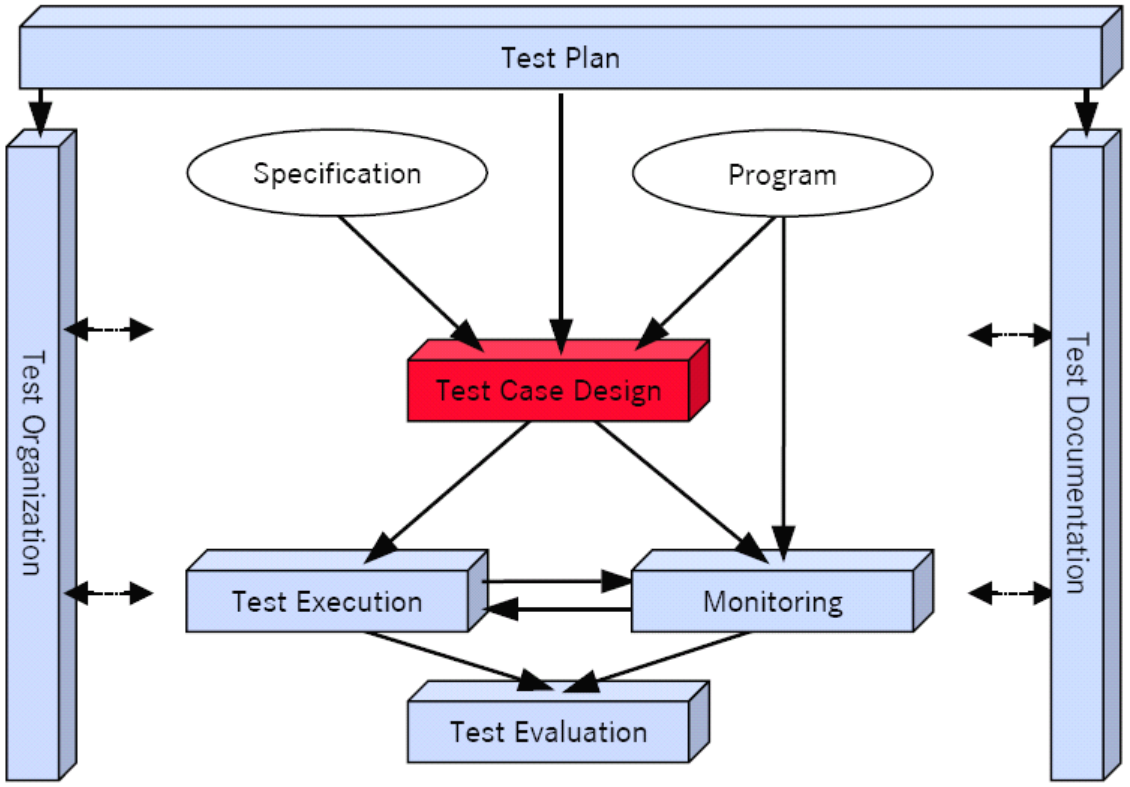
- ✓因时间进度、资源的限制必须结束
  - 一般是为了按计划尽快发布软件，抢占市场
  - 可能存在潜在的严重缺陷
  - 已知的缺陷可能还未修改

课程目录



- ✓执行测试
- ✓结束测试执行
- ✓监控测试执行

测试执行的监控


Test Plan
Specification
Program
Test Organization
Test Case Design
Test Documentation
Test Execution
Monitoring
Test Evaluation

测试监控的任务和目的

- ✓记录和管理测试用例的执行状态
- ✓根据当前的执行状态，判定测试用例的设计质量和效率
  - 使用脚本进行自动测试
- ✓根据发现的缺陷分布，判定结束测试的条件是否成熟

测试监控的任务和目的

- ✓ 评估测软件的质量
  - 缺陷的数量、种类、…
- ✓ 评估开发过程的质量
  - 缺陷的分布、修复缺陷的时间、回归测试时发现的缺陷数量、…
- ✓ 评估测试工程师的表现
  - 是否按计划完成任务
  - 发现缺陷的数量

测试监控的内容

- ✓测试用例执行的进度 = 已执行的数目/总数目
  - 此数据只表明执行进度，不表示测试的成功率
  - 为了得到更精确的进度数据，可计算测试步骤数

测试监控的内容

- ✓缺陷的存活时间 =缺陷从open到closed的时间
  - 表明修改缺陷的效率

测试监控的内容

- ✓ 缺陷的趋势分析 --- 按照测试执行的时间顺序(以月、周、天为时间单位)，被发现的缺陷数量的分布
  - – 如果越来越少，趋近于0，则考虑结束测试执行
  - – 相反，则说明存在以下的问题:
    - > 代码修改引发新的缺陷
    - > 前一版本的测试存在覆盖率的问题，新的测试发现了原先未发现的缺陷
    - > 必须先修改某些缺陷后才能继续测试，然后才发现其他的缺陷

测试监控的内容

- ✓ 缺陷分布密度 =对应于一项需求的总缺陷数/对应于该项需求的测使用例总数
  - – 需要考虑缺陷的优先级和严重程度
  - – 如果过多的缺陷集中在某项需求上，可能表明以下问题:
    - > 该项功能需求是否过于复杂?
    - > 该项的需求设计、实现是否有问题?
    - > 分配给该项的开发资源是否不足?
    - > ……

测试监控的内容

- ✓缺陷修改质量＝每次修改后发现的缺陷数量（包括重现的缺陷和由修改所引起的新缺陷）
  - 评价开发部门修复缺陷的质量
  - 如果修改某项功能后，此数值较高，测试部门应当及时通知开发部门

改进测试执行过程

- ✓基于质量风险分析，先测试最容易出现缺陷、对软件影响最大的部分
- ✓正确分析测试结果
- ✓……