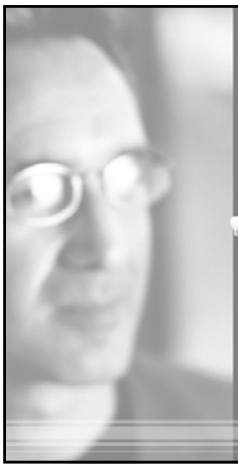


软件外包培训系列

软件测试外包

独立高级咨询师

李 健

2 0 0 4 年 7 月

Copyright © 2004 LJ3P8. All rights reserved.

自我介绍

- 3年以上对欧美软件外包和联盟软件工厂高级管理经验；
- 5年以上的软件过程改善咨询经验；
- 10年以上软件开发、测试、质量保证和项目管理经验；
- 软件工程专家网（www.51cmm.com）首席质量管理专家；
- 北京软协企业管理顾问中心首席咨询师；

针对的问题？

	测试相关技术
1	开发人员怎样做单元测试？
2	性能测试如何做（数据少，无法看出性能问题）？/压力测试？/压力测试数据分析？
3	如何自动测试？
4	回归测试如何做？
5	测试问题的功能点量化与跟踪 ？
6	如何做特殊测试？/数据库连接的测试，界面的测试
7	怎样编写测试用例？
8	怎样做内存泄漏检测、系统性能调优 ？
	测试管理
1	控制测试进度？
2	如何协调测试工作与开发工作？
3	怎样衡量和保证测试是充分的？/测试的路径多，没有耐心走完？
4	怎样进行版本控制？
5	工期紧或有限时间内，确保测试有效？/测试用例会占用时间？

课程期望

1	测试管理、设计思路和结果分析
2	能够更好的将理论应用于实践，测试方法系统化，在技术上能够有所突破
3	希望很培训做成一次深层次的测试交流，而不是单独讲一些理论上的知识。希望在交流过程中，能够真正解决测试中的问题。
4	工具使用，概念系统化，高效测试过程，量化测试进度
5	web网站系统测试工具及方法(j2ee构架)
6	希望了解测试的理论、方法，以及实践工具，以java为主。
7	希望能够得到一个系统化的理论知识，以及希望能够把讲师在测试方面的工作经验传授一些给大家。
8	如何使测试尽可能多的覆盖可能出现的情况？如何进行自动化测试？
9	白盒测试；如何让测试和具体的开发过程结合得更紧密
10	希望本次培训不光是只针对测试人员，也要从开发人员的角度来看一些问题。 希望能提高测试的理论体系，同时也明确开发人员在测试工作中需要充当那些角色。

提 纲

☞软件测试外包

- 软件测试基础
- 软件测试的有效性
- 软件测试工具技术
- 软件测试过程
- 软件测试实施要点

为什么要进行外包 ？

※ 提升公司的核心竞争力

※ 有效控制并大大降低公司业务运作成本

※ 使公司产品快速进入市场

※ 利于整合公司内部资源，扩展业务内容

※ 寻求公司内不可获取的资源

※ 获得专业的软件工程技能

软件外包生产的特征

✓用户/客户的高要求

外包市场的可控性特点:

- 客户的控制力
- 咨询公司的控制力

✓外包项目的低端特性

- 低附加值的软件生产
- 高附加值的低端生产

外包软件生产的规范化

客户的高要求需要软件生产的规范化

✓规范化的要求是工作流程化

✓规范化的要领是业务模板化

软件外包分类

❖软件外包 =（内容，时机，场所，管理）

- 内容：开发（业务建模、系统分析、系统设计、详细设计、编码），测试，维护，运营，项目管理
- 时机/项目阶段：产品初期，产品开发期，产品运营期
- 场所：离岸、本地、现场
- 管理：合同工，独立项目小组

❖外包服务供应商的资源：人、技术、过程

基本软件外包服务内容

（1）软件开发

- 软件项目开发
- 软件产品的客户化
- 软件产品的增值开发

（2）软件维护和再造

（3）软件测试

- 项目测试
- 第三方产品测试
- 第三方验收测试

业务运作模型

业务合同类型	计费模型	支付方式
项目合同 （工作范围和规范清晰）	标的价格和时间	基于里程碑支付
项目合同 （工作范围和规范不清晰）	实际工作内容	基于里程碑支付
“下工程”子合同	基于人力资源的计费	基于单位时间支付
人员外包	基于人力资源的计费	基于单位时间支付

计费模型

✦标的价格和时间

项目的估价是以项目固定的工作内容和时间为基础，需求在项目过程中很小变化。

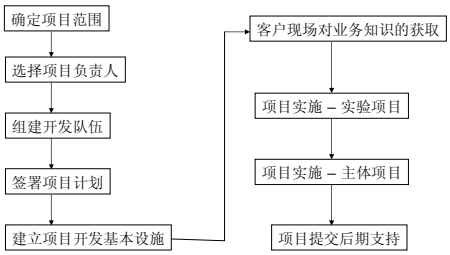
✦实际工作内容

对于需求不明确或可能发生很大的变化的项目，以实际发生的工作时间和投入作为估价的基础。这种方式，通常通过每周的时间占用表和项目进度报告来跟踪控制。

✦基于人力资源占用

计费是以每个时间单元为基础，乘上所占用的各个层次的人力资源量。

基本业务流程



客户收益

- ✓保证优质的软件产品
- ✓降低测试成本（约 40%）
- ✓降低开发成本 （约 12% ~ 30%）
- ✓降低管理成本
- ✓提升企业的核心竞争力
- ✓改善软件开发过程
- ✓享受专业的软件开发和测试服务

软件测试外包特点

- 市场特点
  - 软件测试外包的客户认可度高；
  - 国内市场存在相当的软件测试外包量；
- 技术特点
  - 不同于软件开发外包的低端特性，它具备高端内容；
  - 对业务的了解要求高于软件开发外包项目；
- 管理特点
  - 充分利用时差优势；
  - 通过合作建立长期的关系，并在此基础上评价。

软件测试外包的类型

项目测试	作为外部独立的QC资源，在软件项目系统开发过程中进行质量控制； 测试项目的持续时间较长； 可同时服务于软件开发项目的甲乙双方。
产品测试	为软件项目开发提供独立的第三方测试； 可服务于企业内部验收测试；
验收测试	服务于软件开发项目的客户方（“甲方”），帮助客户实施验收测试；

软件测试外包项目的验收

- 项目验收
  - 产品的质量指标(0.04-0.09/defects/FP)？
  - 完成的测试用例数？
  - 完成的覆盖指标？
  - 抽查测试结果？
  - 验证的验证谁来做？
- 公司评估
  - 人员评估
  - 项目评估
  - 非技术评估

软件测试外包的前提

- 客户方
  - 对外包内容的控制；
  - 对承包方的评估体系
- 承包方
  - 区分产品开发和专业服务；
  - 相对稳定的过程；
  - 相对稳定的人员

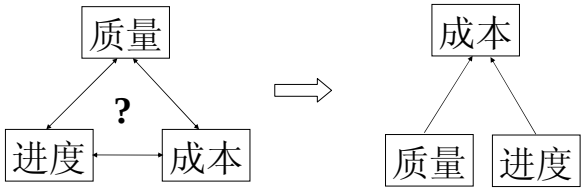
提　纲

- ☞软件测试外包
  - ☞ 软件测试外包项目管理
- 软件测试基础
- 软件测试的有效性
- 软件测试工具技术
- 软件测试过程
- 软件测试实施要点

软件测试外包项目中的甲乙双方

- “客户”—软件产品生产厂商
  - 结果管理/内容管理
  - 过程管理
- “客户” —软件产品最终客户
  - 结果管理/内容管理
  - 处理与第三方（监理方）的合作关系

软件测试外包项目管理



软件生产的特点

- 依赖人的创造性？
- 技术革新快？
- 客户需求不稳定？
- “制造”成本高？
- 产品或服务的个性化要求高？
- 多是无形资产投入？

…

“灵活性”是软件的特点

“不确定性”和“变更”是软件生产的特点

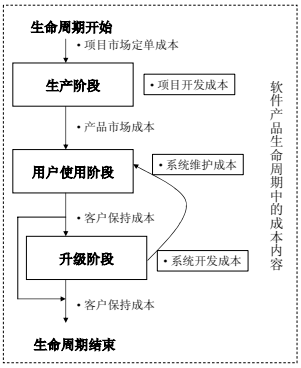
软件生产成本

✓ 软件生产成本（狭义）

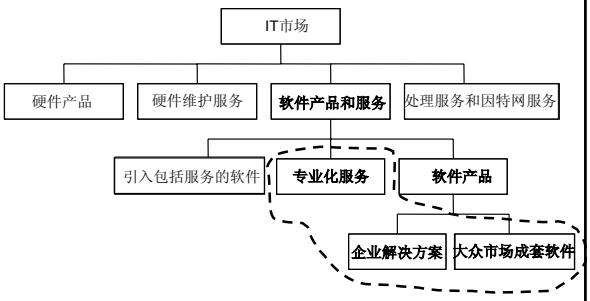
- 项目或产品开发成本
- 系统或产品维护成本

✓ 软件生产成本（广义）

- 项目市场定单成本
- 产品市场成本
- 客户保持成本

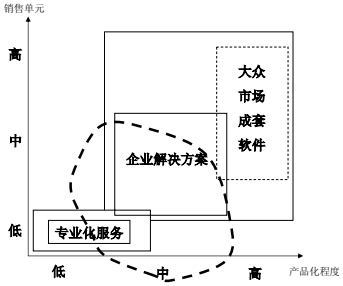


IT市场划分



分类资料来源：IDC，麦肯锡数据库，麦肯锡分析

三个市场领域的异同



企业产品	大众产品
量身定做	有数可选方案（大众版、企业版等）
安装需要几个月乃至几年，而非几个小时	几个小时安装
不是为数百万，而是为数千	寻求高市场分额

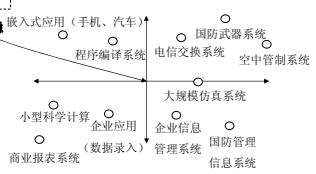
专业服务	产品行业
边际成本几乎不变	边际成本为零
市场结构高度分化	市场结构高度集中
主要是地域化，全球化趋势增强	高度全球化
客户关系一对一	客户关系一对多
生产力利用率	市场份额
1. 人力资源	1. 战略
2. 软件开发	2. 营销和销售
3. 营销和销售	3. 人力资源
4. 战略	4. 软件开发

不同项目类型特点不同

一般软件项目
- 5到10人
- 10到12个月
- 3到5个外部接口
- 需求部分不稳定

高技术复杂度
- 嵌入式、实时、分布式、容错系统
- 高性能要求、可移植性要求高
- 新系统、"架构"再工程

低管理复杂度
- 规模小
- 非正式
- 利益相关者少
- "产品"开发



高管理复杂度
- 规模大
- 正式的合同项目
- 利益相关者多
- "项目"开发

低技术复杂度
- 管理型简单系统、单一线程系统
- 交互式性能要求、单一平台
- 维护性开发，"应用"再工程

软件生产中的经济模型

工作量=（人员技能）（环境）（质量）（规模^{过程}）

COCOMO / COCOMO II：

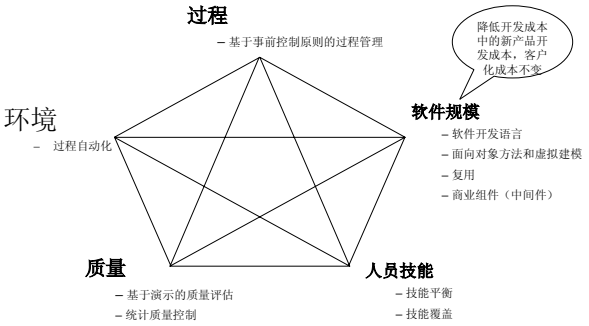
$$\text{Effort} = C1\ EAF(\text{Size}^{P1})$$

$$\text{Time} = C2\ (\text{Effort})^{P2}$$

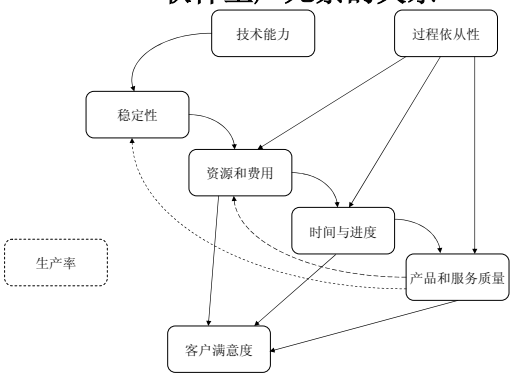
软件过程是以指数形式影响软件生产的成本。

软件开发规模越大，过程质量所产生的影响就越大。

软件项目成本改善途径



软件生产元素的关系



软件生产特点对项目成本管理的影响

有效的软件项目成本管理

II

有效的软件项目变更管理

软件测试外包项目变更

编号	变更类型	责任	80/20
1	测试项目目标变化	非责任变更	?
2	测试项目范围变化	非责任变更	?
3	测试技术要求变化	责任变更和非责任变更	?
4	测试完成指标变化	责任变更和非责任变更	?
5	测试过程变化	责任变更和非责任变更	?
6	测试组成员变更	责任变更	?
7	测试支持环境变更	责任变更和非责任变更	?

成功的软件测试外包项目

- 客户满意度
  - 外包项目实施过程质量
  - 外包项目实施结果质量
- 项目利润
  - 降低项目成本
  - 控制项目成本
- 公司利润
  - 外包项目实施过程质量（QA在其中的作用？）

软件测试外包项目的风险管理

编号	典型风险	风险
1	性能测试技术	?
2	回归测试技术	?
3	对测试对象业务逻辑的理解	?
4	明确的测试目标	?
5	与客户顺畅的沟通	?
6	测试组成员变更	?
7	测试支持环境	?

提　纲

- 软件测试外包
- ☞软件测试基础
- 软件测试的有效性
- 软件测试工具技术
- 软件测试过程
- 软件测试实施要点

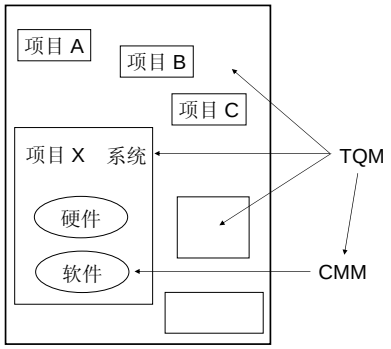
基本质量概念

- 软件质量

  质量是一个产品、组件或过程的特征满足特定需求的程度。质量也是一个系统、组件或过程满足客户或用户的需求和期望的程度。[IEEE-STD-610]

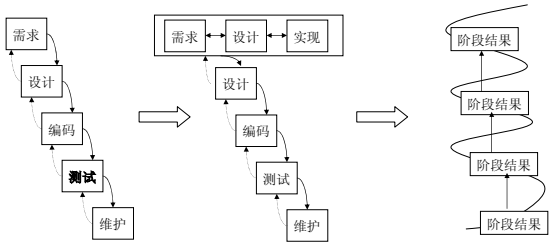
- 全面质量管理思想（TQM）
  - 软件生命周期模型的转变
  - 软件开发方法的转变

全面质量管理（TQM）



- 管理层介入
- 目的和可度量的目标
- 教育和培训
- 员工介入和实施小组
- 客户关注
- 认证和培训合作伙伴
- 有效的激励

软件生命周期模型的转变



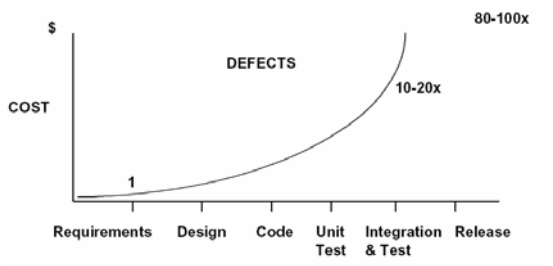
质量控制事后转变为事前和事中，单角色转变为多角色

软件开发方法的转变

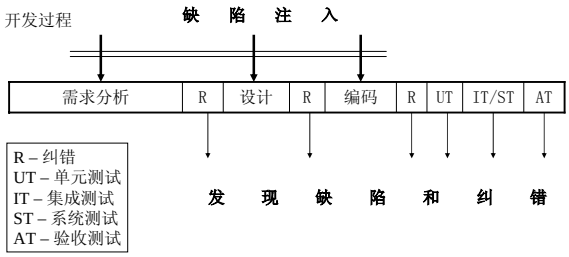
- ➔ 结构化方法
- ➔ 面向对象方法
  - ❶易于对现实应用进行建模
  - ❷支持重用

质量控制事后转变为事前和事中，质量内涵的转变

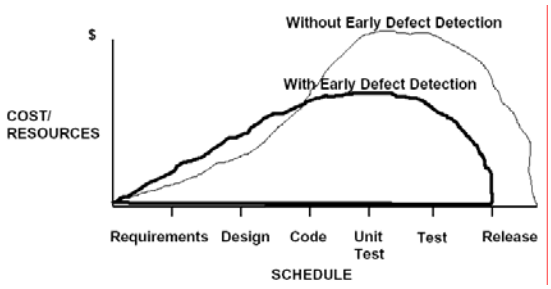
质量代价比率变化



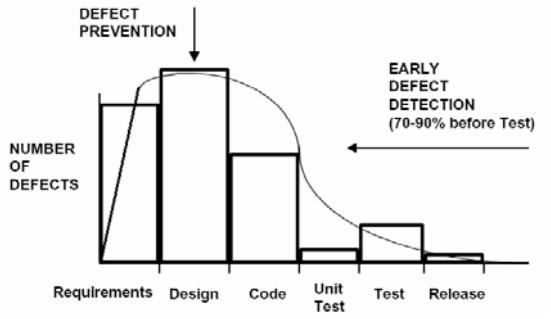
产品质量问题的产生与更正



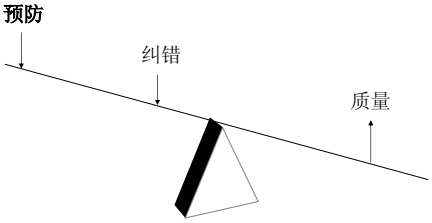
“早”发现“早”收益



优秀的质量管理策略



基本质量策略



预防为主：事前 / 事中

QC、QA、测试、验证、确认

- 质量控制（QC）

  用于满足质量需求的技术和活动。[ISO 8402-1994]

- 质量保证（QA）

  用于管理已定义的标准、规范、规程、方法、过程能够在项目过程中得到有效应用的一套有计划的系统化手段。

- 软件测试

  软件测试是为了发现错误而执行程序的过程。发现错误？？

- 验证（Verification） -“You built it right”

  在软件生命周期的各个阶段对阶段产出的检查或评价以确保满足各自的需求。

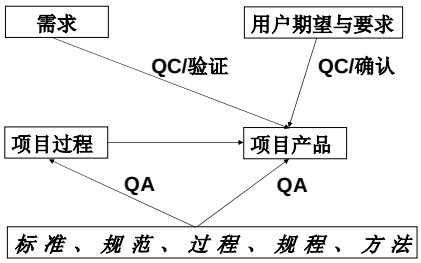
- 确认（ Validation ）-“You built the right thing”

  以用户使用的方式对软件产品或系统进行检查或评价以确定满足用户的需求。

QC、QA、测试、验证、确认（续）

	定位	对象	角色	技术手段
QC	质量策略	项目产品	项目组	评审、测试、模拟
QA	质量策略	项目产品和过程	QA组	评审、审核、评价/评估
测试	质量手段	可执行项目产品	测试组	
验证	质量策略	项目产品	项目组	评审、测试、Demo、模拟、原型
确认	质量策略	项目产品	项目组	评审、测试、Demo、模拟、原型

QC、QA、验证、确认



QC与QA

- 为什么要区分QA与QC？

  QC的目的是控制；QA的目的是改善；

  QC关注的是项目近期目标，QA更应该关注长期目标；

- 职责范围不同

  QC和QA全员的职责；

  QC由工程人员推动，QA由PM和质量保证人员推动。

- QA与QC的联系

  QC活动、过程是QA的工作对象；

  QA自身的过程、活动需要进行QC。

验证（VER）与确认（VAL）

- 为什么要区分验证和确认？

  “验证”强调对需求的满足；

  “确认”强调客户参与，更直接提升客户满意度；

  “验证”与“确认”涉及的角色不同，发现问题和偏差后处理优先级不同；

  “验证”结果衡量的是项目组中某个职能过程的性能；“确认”结果衡量的是整个项目过程的性能。

- 验证与确认的联系

  验证和确认可以采用相同的实施手段、技术；

  验证的过程输出可以作为确认的输入。

软件测试的目的

软件测试是为了发现缺陷

软件测试是为了发现重要的缺陷

软件测试是为了发现对客户重要的缺陷

测试的原则

- 程序员或程序设计机构不应测试自己设计的程序，？？
- 测试用例设计中，不仅要有确定的输入数据，而且要确定预期输出的数据
- 测试用例的设计不仅要有合理的输入数据，还要有不合理的输入数据
- 除了检查程序是否作完了它应做的事外，还要检查它是否做了不应做的事
- 保留全部的测试用例，并作为软件的组成部分之一
- 程序中存在错误的概率与在该段程序中已发现的错误数成比例（程序中错误产生的集群现象）

系统\产品的问题形式

错 误 → 缺陷/Bug → 故 障 → 失 效

指在软件生存周期内的不希望或不可接受的人为错误，其结果是导致软件缺陷（Defect/Bug）的产生。软件错误是一种人为过程，相对软件本身是一种外部行为。

软件缺陷存在于软件（文档、数据、程序）之中的那些不希望或不可接受的偏差，其结果是软件运行于某一特定条件时出现软件故障。软件缺陷存在于软件内部、静态形式。

软件故障是指软件运行过程中出现的一种不希望或不可接受的内部状态。此时若无适当措施（容错）加以及时处理，便产生软件失效（Failure）。软件故障是一种动态行为。

软件失效是指软件运行时产生的一种不希望或不可接受的外部行为结果。

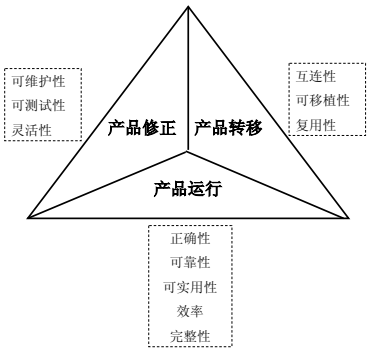
提　纲

- 软件测试外包
- ☞软件测试基础
  - ☞ 测试的类型
- 软件测试的有效性
- 软件测试工具技术
- 软件测试过程
- 软件测试实施要点

产品质量目标

功能性	可靠性	效率	可移植性	易使用性	可维护性
适合性 *准确性* *互用性* *依从性* *安全性*	*成熟性* *容错性* *可恢复性*	*时间特性* *资源特性*	*适应性* *可安装性* *一致性* *可替换性*	*可理解性* *易学习性* *操作性*	*可分析性* *可变更性* *稳定性* *可测试性*

产品质量目标（续）



测试策略

功能性	可靠性	效率	可移植性	易使用性	可维护性
适合性 *准确性* *互用性* *依从性* *安全性*	*成熟性* *容错性* *可恢复性*	*时间特性* *资源特性*	*适应性* *可安装性* *一致性* *可替换性*	*可理解性* *易学习性* *操作性*	*可分析性* *可变更性* *稳定性* *可测试性*

功能测试
数据与数据库测试
安全与访问测试
配置测试

失效与恢复测试

性能测试
负载测试
容量测试
压力测试

安装测试

用户界面测试

测试的类型

编号	测试类型	测试实施时机	说明
1	功能测试	单元测试、集成测试	验证测试对象的功能是否满足规格说明；
2	结构测试	单元测试、集成测试	检查软件的内部活动是否符合设计要求的方法，以及软件的接口是否符合规范
3	可靠性测试	集成测试、系统测试	检测系统无故障运行时间。
4	强度测试	系统测试	是在高负荷紧张的情况下进行的测试。
5	性能测试	系统测试	是检查系统是否满足指定的性能指标，协调整个系统并判断软件和硬件中的因素是否限制了系统的性能。
6	恢复测试	系统测试	证明出现程序错误、硬件失效及数据错误之后，整个系统恢复工作的功能是否正常。测试方法是把一些程序错误故意输入到系统中，观察是否能在出错后恢复。模拟硬件错误，模拟/制造数据错误，来分析系统的反应。
7	启动/停止测试	系统测试	验证系统的初始化功能和断点处理功能。
8	配置测试	系统测试	证明程序所需配置的各种情况的可达到性。对特定类型的硬件和最大最小配置都要测试到。
9	安全性	系统测试	力图确定系统的安全性问题。测试方法是确定类似的其它系统中出现过的一些安全性问题，然后设法列出一些测试用例来破坏程序的保密性，证明系统也有类似问题。

 性能测试指标

测试的类型（续）

编号	测试类型	测试实施时机	说明
10	可使用性测试	集成测试、系统测试	验证系统的易使用性。
11	可支撑性测试	系统测试、验收测试	测试支持系统运行所有的过程、方法、文档、规程是否可行。
12	安装测试	系统测试、验收测试	验证系统在应用环境下的各种安装选择的正确性。
13	互连测试	系统测试	验证系统与客户其他系统之间的互操作性。此互操作性在项目需求规格说明中已明确。
14	兼容性测试	集成测试、系统测试	测试系统不同版本之间的信息兼容性和运行兼容性。
15	容量测试	系统测试	要检测系统的能力最高能达到什么程度。最大处理能力、处理规模，而且表现在持续的处理时间上，可以是在最大事务处理规模下的持续时间，也可以是一定事务规模下的持续时间。
16	文档测试	系统测试、验收测试	验证所有提交给用户的文档，文档的无二义性和精确性。

提　纲

- 软件测试外包
- ☞软件测试基础
  - ☞ 测试的基本方法
- 软件测试的有效性
- 软件测试工具技术
- 软件测试过程
- 软件测试实施要点

软件测试方法

	测试方法	说明
1	静态测试方法	不执行被测程序的情况下扫描源程序，检查程序模块间的接口是否一致及变量的定义和使用情况。通过对被测试软件的源程序的控制流(如函数调用关系、函数的控制流程图)和数据流(如外部变量使用情况等)的分析来检验程序。
2	动态测试方法	在程序执行的过程中跟踪程序的运行情况，得到程序运行的覆盖率、数据流的变化和程序的执行动态行为。
3	“白盒子”测试方法	检查软件的内部活动是否符合设计要求的方法。它检查软件内部的逻辑结构，是以仔细检查过程的细节为基础，通过提供一组指定条件和循环的测试用例，对穿过软件的逻辑路径进行测试，可以在不同点检查程序的状态，以确定实际状态与预期状态是否一致。白盒子测试也称为结构测试。
4	“灰盒子”测试方法	测试系统部件之间的接口。
5	“黑盒子”测试方法	着眼于软件的外部特性，而不考虑软件的内部逻辑结构。黑盒子测试指在软件的接口上进行测试，即看它能否满足功能要求，输入能否被正确地接收，并正确地输出结果，以及能否保持外部信息的完整性等。黑盒子测试也称为功能测试。

静态分析结果

提　纲

- 软件测试外包
- ☞软件测试基础
  - ☞ 测试用例的设计
- 软件测试的有效性
- 软件测试工具技术
- 软件测试过程
- 软件测试实施要点

软件测试用例

- 测试用例

测试用例是实施一次测试而向被测系统提供的输入数据、操作或各种环境设置。

- 测试用例设计的基本原则
  - 测试用例的代表性：能够代表并覆盖各种合理的和不合理的、合法的和非法的、边界的和越界的，以及极限的输入数据、操作和环境设置等。
  - 测试结果的可判定性：即测试执行结果的正确性是可判定的。
  - 测试结果的可再现性：即对同样的测试用例，系统的执行结果应当是相同的。
  - 测试用例执行的自动化特性：测试用例的自行可以工具或批处理执行。

软件测试用例设计的方法

	测试用例设计方法	说明
1	等价类划分	把所有可能的输入数据，即程序的输入域划分为若干部分，然后从每一部分中选取少数有代表性的数据作为而是用例。
2	边界值分析	针对大量错误是发生在输入或输出范围的边界上，针对此情况设计的测试用例是边界值分析。
3	因果图	通过分析输入之间的因果关系，生成不同类型的输入组合，形成组合测试用例。
4	结构覆盖类用例	语句覆盖，分支覆盖，判定覆盖，条件覆盖和路径覆盖。
5	工具捕获	通过记录/回放工具捕获的系统行为，并在此基础上修改得到的测试用例。提升了测试用例设计的效率，但从覆盖上不易掌握。

覆盖情况 ***等价划分***

提 纲

- 软件测试外包
- ☞软件测试基础
  - ☞ 测试的阶段
- 软件测试的有效性
- 软件测试工具技术
- 软件测试过程
- 软件测试实施要点

测试阶段

- 测试阶段

  测试阶段描述的是测试执行的时机，与测试对象的成熟程度密切相关。

- 测试阶段划分的意义

  （1）各个阶段的测试对象不同；

  （2）各个阶段测试的内容和侧重不同；

  （3）各个阶段实施的人员不同。

典型测试阶段

	测试阶段	说明
1	单元测试	对程序设计中的最小单元进行测试，进行正确性检测。
2	集成测试	在单元测试基础上，需要对按照设计要求集成起来的系统或子系统进行测试，着重各个模块之间的结构和系统的功能。
3	系统测试	集成完成后的系统放入相应的运行软、硬件环境中进行测试，测试侧重整个系统的功能，特别是性能的表现。
4	回归测试	回归测试是在系统已经基本稳定，一轮测试中发现的错误已十分有限的情况下进行的。主要复查以知错误的纠正情况，确认未引发任何新的错误。最后进行终结回归测试。
5	验收测试	验收测试对软件质量保证无直接贡献。验收测试包括对测试的验收，有测试的过程、测试的方法、测试的内容以及测试的结果。
6	Alpha测试	在开发场所下，用户进行的体验测试。
7	Beta测试	用户进行的产品测试，即在用户使用环境中进行使用并测试

问题和讨论