软件测试规范

GB/T 15532-2008

软件测试规范

- 总则
- 单元测试
- 集成测试
- 确认测试
- 系统测试
- 验收测试

总则

- 测试条目组织

  对主要的测试类别都是按“测试对象和目的”、“测试的组织和管理”、“技术要求”，“测试内容”、“测试环境”、“测试方法”、“准人条件”、“准出条件”、“测试过程”和“输出文档”等条目做出要求。

- 测试目的

  a) 验证软件是否满足软件开发合同或项目开发计划、系统／子系统设计文档、软件需求规格说明、软件设计说明和软件产品说明等规定的软件质量要求；

  b)通过测试，发现软件缺陷；

  c) 为软件产品的质量测量和评价提供依据。

总则

- 测试类别

  a） 单元测试；

  b） 集成测试；

  c） 配置项测试（也称软件合格性测试或确认测试）；

  d） 系统测试；

  e） 验收测试。

  可根据软件的规模、类型、完整性级别选择执行测试类别。

  回归测试可出现在上述每个测试类别中，并贯穿于整个软件生存周期。

总则

- 测试过程

  软件测试过程一般包括四项活动：测试策划、测试设计、测试执行、测试总结。

  – 测试策划

    测试策划主要是进行测试需求分析。

  – 测试设计

    依据测试需求，分析并选用已有的测试用例或设计新的测试用例。

  – 测试执行

    执行测试用例.

  – 测试总结

    整理和分析测试数据.

ISTQB：

计划与控制
分析与设计
实施与执行
评估出口准则和报告
测试结束活动

总则

- 测试方法
  - 静态测试方法

    静态测试方法包括检查单和静态分析方法.
  - 动态测试方法

    动态测试方法一般采用白盒测试方法和黑盒测试方法。

    在软件动态测试过程中，应采用适当的测试方法，实现测试目标。

总则

- 测试用例
  - 测试用例设计原则

    a）基于测试需求的原则。

    b）基于测试方法的原则。

    c）兼顾测试充分性和效率的原则。

    d）测试执行的可再现性原则。
  - 测试用例要素

    每个测试用例应包括以下要素：a）名称和标识。b）测试追踪。c）用例说明。d）测试的初始化要求（硬件配置、软件配置、测试配置、参数设置、其他对于测试用例的特殊说明）。e）测试的输入。f）期望的测试结果。g）评价测试结果的准则。h）操作过程。i) 前提和约束。j) 测试终止条件。

总则

- 测试管理
  - 过程管理

    软件测试应由相对独立的人员进行。根据软件项目的规模等级和完整性级别以及测试类别，软件测试可由不同机构组织实施。

    应对测试过程中的测试活动和测试资源进行管理。

    一般情况下，软件测试的人员配备见下表。

工作角色	具体职责
测试项目负责人	管理监督测试项目，提供技术指导，获取适当的资源，制定基线，技术协调，负责项目的安全保密和质量管理
测试分析员	确定测试计划、测试内容、测试方法、测试数据生成方法、测试（软、硬件）环境、测试工具，评价测试工作的有效性
测试设计员	设计测试用例·确定测试用例的优先级，建立测试环境
测试程序员	编写测试辅助软件
测试员	执行测试、记录测试结果
测试系统管理员	对测试环境和资产进行管理和维护
配置管理员	设置、管理和维护测试配置管理数据库

总则

- 测试的准入准出条件如下：
  - a) 准入条件
    1) 具有测试合同（或项目计划）；
    2) 具有软件测试所需的各种文档；
    3) 所提交的被测软件受控：
    4) 软件源代码正确通过编译或汇编。

总则

- 测试的准入准出条件如下：
  - b) 准出条件

    1) 已按要求完成了合同（或项目计划）所规定的软件测试任务；

    2) 实际测试过程遵循了原定的软件测试计划和软件测试说明；

    3) 客观、详细地记录了软件测试过程和软件测试中发现的所有问题；

    4) 软件测试文档齐全、符合规范；

    5) 软件测试的全过程自始至终在控制下进行；

    6) 软件测试中的问题或异常有合理解释或正确有效的处理；

    7) 软件测试工作通过了测试评审；

    8) 全部测试软件、被测软件、测试支持软件和评审结果已纳入配置管理。

总则

- 配置管理

  应建立配置管理库，将被测试对象和测试工作产品纳入配置管理。

- 评审
  - 测试就绪评审

    在测试执行前，对测试计划和测试说明等进行评审。评审的具体内容和要求应包括：

    a) 评审测试文档内容的完整性、正确性和规范性；

    b) 通过比较测试环境与软件真实运行的软件、硬件环境的差异，评审测试环境要求是否正确合理，满足测试要求；

    c) 评审测试活动的独立性；

    d) 评审测试项选择的完整性和合理性；

    e) 评审测试用例的可行性、正确性和充分性。

总则

- 评审(continue)
  - 测试评审

    在测试完成后，评审测试过程和测试结果的有效性，确定是否达到测试目的。主要对测试记录、测试报告进行评审，其具体内容和要求应包括：

    a) 评审文档和记录内容的完整性、正确性和规范性；

    b) 评审测试活动的独立性和有效性：

    c) 评审测试环境是否符合测试要求：

    d) 评审测试记录、测试数据以及测试报告内容与实际测试过程和结果的一致性；

    e) 评审实际测试过程与测试计划和测试说明的一致性：

    f) 评审未测试项和新增测试项的合理性；

    g) 评审测试结果的真实性和正确性；

    h) 评审对测试过程中出现的异常进行处理的正确性。

总则

- 测试文档

  软件测试文档一般包括测试计划、测试说明（需要时进一步细分为测试设计说明、测试用例说明和测试规程说明）、测试项传递报告、测试日志、测试记录、测试问题报告（也称测试事件报告）和测试总结报告，测试文档的基本内容和要求见GB./T 9386。

软件测试策略

- 测试策略：
  （1）测试从模块层开始，然后扩大延伸到整个基于计算机的系统集合中。
  （2）不同的测试技术适用于不同的时间点。
  （3）测试是由软件的开发人员和（对于大型系统而言）独立的测试组来管理的。
  （4）测试和调试是不同的活动，但是调试必须能够适应任何的测试策略。

软件测试充分性准则

- 对任何软件都存在有限的充分测试集合。
- 如果一个软件系统在一个测试数据集合上的测试是充分的，那么再多测试一些数据也应该是充分的。这一特性称为单调性。
- 即使对软件所有成分都进行了充分的测试，也并不表明整个软件的测试已经充分了。这一特性称为非复合性。
- 即使对软件系统整体的测试是充分的，也并不意味软件系统中各个成分都已经充分地得到了测试。这个特性称为非分解性。
- 软件测试的充分性应该与软件的需求和软件的实现都相关。
- 软件越复杂，需要的测试数据就越多。这一特性称为复杂性。
- 测试得越多，进一步测试所能得到的充分性增长就越少。这一特性称为回报递减率。

软件测试规范

- 总则
- 单元测试
- 集成测试
- 确认测试
- 系统测试
- 验收测试

单元测试(Unit Testing)

- 概述
- 单元测试的内容
- 单元测试的步骤
- 单元测试的执行

单元测试(Unit Testing) 概述

- 术语

  单元测试：又称模块测试，是针对软件设计的最小单位——程序模块进行正确性检验的测试工作。

  其目的是检查每个软件单元能否正确地实现设计说明中的功能、性能、接口和其他设计约束等要求，发现单元内可能存在的各种差错。

- 测试对象

  是可独立编译或汇编的程序模块（或称为软件构件或在面向对象设计中的类）。

  只测单元的内部行为，单元间接口不在此时测

  在单元测试活动中，软件的独立单元将在与程序的其他部分相隔离的情况下进行测试。

单元测试(Unit Testing)概述

- 测试的组织和管理

  一般由软件的供方或开发方组织并实施软件单元测试，也可委托第三方进行软件单元测试。

  软件单元测试的技术依据是软件设计文档（或详细设计文档）

- 技术要求
  - 软件单元测试一般应符合以下技术要求：

    a) 对软件设计文档规定的软件单元的功能、性能、接口等应逐项进行测试；

    b) 每个软件特性应至少被一个正常测试用例和一个被认可的异常测试用例覆盖；

    c) 测试用例的输入应至少包括有效等价类值、无效等价类值和边界数据值；

    （continue）

单元测试(Unit Testing)概述

- 技术要求（continue）
  - 软件单元测试一般应符合以下技术要求：（continue）

    d) 在对软件单元进行动态测试之前，一般应对软件单元的源代码进行静态测试：

    e) 语句覆盖率达到100%；

    f) 分支覆盖率要达到100%,

    g) 对输出数据及其格式进行测试。

    对具体的软件单元，可根据软件测试合同（或项目计划）以及软件单元的重要性、完整性级别等要求对上述内容进行裁剪。

单元测试(Unit Testing)概述

- 测试策略

  单元测试需要从程序的内部结构出发设计测试用例，多采用白盒测试技术为主，黑盒为辅。 多个模块可以平行地独立进行单元测试。

单元测试的内容

- 总则
  - 静态测试

    当静态测试时，所测试的内容与选择的测试方法有关。如，采用代码审查方法，通常要对寄存器的使用（仅限定在机器指令和汇编语言时考虑）、程序格式、人口和出口的连接、程序语言的使用、存储器的使用等内容进行检查；

    采用静态分析方法，通常要对软件单元的控制流、数据流、接口、表达式等内容进行分析。
  - 动态测试

    当动态测试时，通常对软件单元的功能、性能、接口、局部数据结构、独立路径、出错处理、边界条件和内存使用情况进行测试。

对具体的软件单元，应根据软件测试合同（或项目计划）、软件设计文档的要求及选择的测试方法确定测试的具体内容。

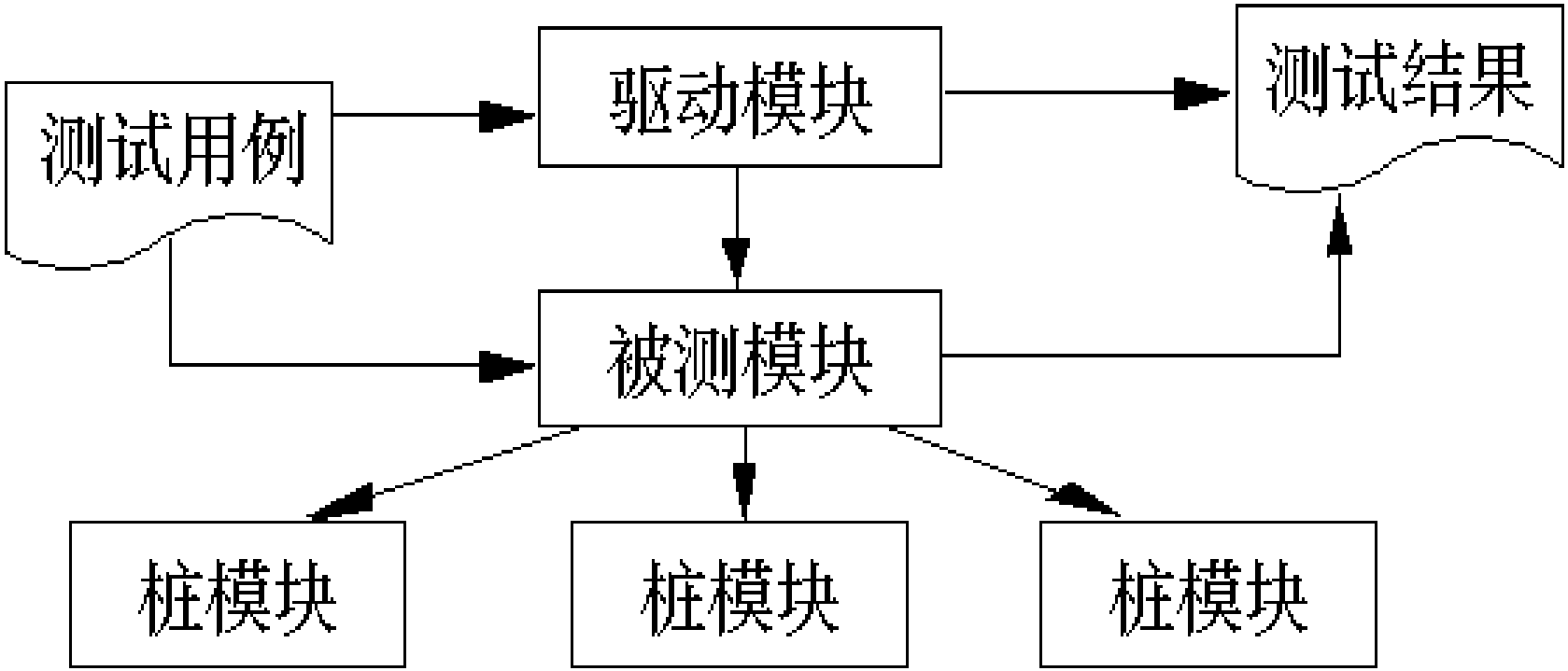
单元测试

- 驱动模块和桩模块

  模块并不是一个独立的程序，在考虑测试模块时，同时要考虑它和外界的联系，应为测试模块开发一个驱动模块（driver）和（或）若干个桩模块（stub）。下页图显示了一般单元测试的环境。

  – 驱动模块：用于模拟被测模块的上级模块。在大多数场合称为“主程序”，它接收测试数据并将这些数据传递到被测试模块，被测试模块被调用后，“主程序”显示“进入-退出”消息。
  – 桩模块：也称存根程序，用于模拟被测模块的调用模块。

单元测试



单元测试

- 驱动模块和桩模块是测试使用的软件，而不是软件产品的组成部分，但它需要一定的开发费用。

  因此开发驱动和桩模块应尽量简单。

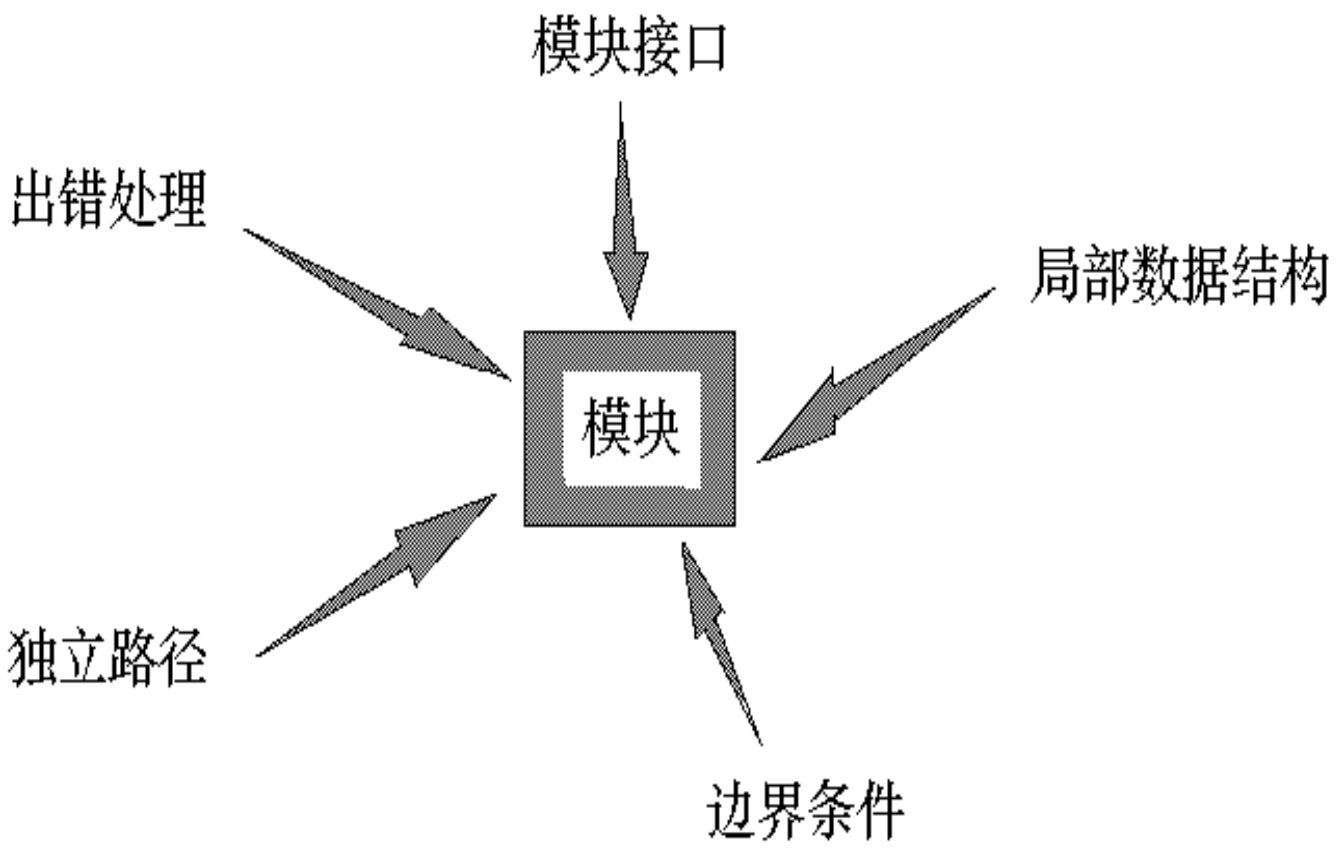
  遗憾的是，很多程序单元用简单的驱动和桩模块不能完成测试，这些模块的单元测试只能采用一些综合测试方法（常放到集成测试中再进行）。

- 提高模块的内聚度可简化单元测试。

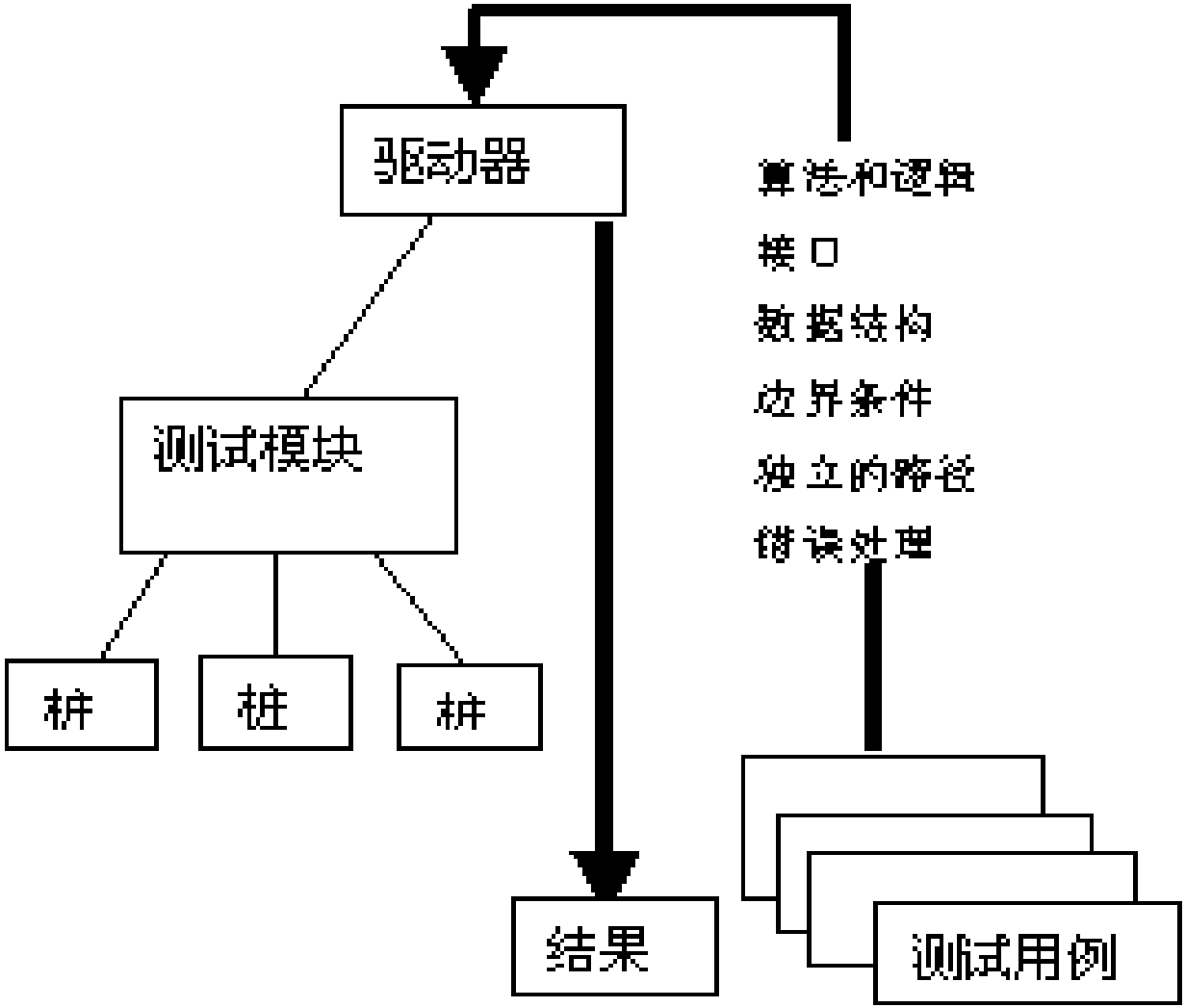
  如果每个模块只完成一个功能，所需测试用例数目将显著减少，模块中的错误也更易发现。

  如果一个模块要完成多种功能，可以将这个模块看成由几个小程序组成。必须对其中的每个小程序先进行单元测试要做的工作，对关键模块还要做性能测试。

单元测试的考虑



单元测试的考虑



单元测试的考虑 - 模块接口测试

- 应首先对通过被测模块的数据流进行测试。

  只有在数据能正确流入、流出模块的前提下，其他测试才有意义。

  测试接口一般应包括以下肉容：

  - 调用本模块的输入参数是否正确(参数数目、属性、类型、次序)；
  - 本模块调用子模块时输入给子模块的参数是否正确；
  - 调用内部函数的参数是否正确；
  - 是否修改了只读型参数
  - 全局量的定义在各模块中是否一致；
  - 在单元有多个入口的情况下，是否引用了与当前入口无关的参数；
  - 常数是否当作变量来传递；

单元测试的考虑 - 模块接口测试

- 如果模块内包括外部输入输出，还应该考虑下列因素：
  - 文件属性是否正确；
  - OPEN与CLOSE语句是否正确；
  - 规定的输入 / 输出格式说明与输入 / 输出语句是否匹配；
  - 缓冲区容量与记录长度是否匹配；
  - 在进行读写操作之前是否打开了文件；
  - 在结束文件处理时是否关闭了文件；
  - 正文书写 / 输入错误，
  - I / O错误是否检查并做了处理。

单元测试的考虑 - 局部数据结构测试

- 检查局部数据结构是为了保证临时存储在模块内的数据在程序执行过程中完整、正确。

  局部数据结构往往是错误的根源，应仔细设计测试用例，力求发现下面几类错误：

  – 不正确或不一致的数据类型说明
  – 使用尚未赋值或尚未初始化的变量
  – 错误的初始值或错误的缺省值
  – 变量名拼写错或书写错
  – 上溢、下溢（数组越界）或地址异常（非法指针）

  除了局部数据结构外，如可能，单元测试时还应该查清全局数据（例如FORTRAN的公用区）对模块的影响。

单元测试的考虑 - 路径测试

- 在模块中应尽可能地对每一条独立执行路径进行测试，满足某覆盖标准。

  独立路径是指在程序中至少引进一个新的处理语句集合或一个新条件的任一路径。在程序的控制流图中，一条独立路弪是至少包含有一条在其他独立路径中从未有过的边的路径。

  应设计适当的测试用例，对软件单元中的独立路径进行测试，特别是对独立路径中的基本路径进行测试。

  基本路径指在程序控制流图中，通过对控制构造的环路复杂性分析而导出的基本的、可执行的独立路径集合。

  此时设计测试用例是为了发现因错误计算、不正确的比较和不适当的控制流造成的错误。

单元测试的考虑 - 路径测试

由于不能穷举测试，所以基本路径测试和循环测试是最常用且最有效的测试技术。计算中常见的错误包括：

- 死代码
- 误解或用错了算符优先级；
- 混合类型运算；
- 变量初值错；
- 精度不够；
- 表达式符号错。

单元测试的考虑 - 路径测试

比较判断与控制流常常紧密相关，测试用例还应注意发现下列错误：

- 不同数据类型之间的比较；
- 错误地使用逻辑运算符或优先级；
- 因计算机表示的局限性，期望理论上相等而实际上不相等的两个量相等；
- 关系表达试中比较运算或变量出错；
- 循环终止条件或不可能出现；
- 迭代发散时不能退出；
- 错误地修改了循环变量。

单元测试的考虑 - 错误处理测试

- 一个好的设计应能预见各种出错条件，并预设各种出错处理通路，出错处理通路同样需要认真测试，测试应着重检查下列问题：
  - 出错的描述是否难以理解
  - 在对错误进行处理之前，错误条件是否已经引起系统的干预。
  - 所提供的差错描述信息不足以确定造成差错的位置或原因
  - 显示的错误与实际的错误是否相符
  - 对错误条件的处理正确与否
  - 意外的处理不当
  - 联机条件处理（即交互处理等）不正确

单元测试的考虑 - 边界测试

- 边界测试是单元测试中最后，也是最重要的一项任务。

  软件经常在边界上失效，应采用边界值分析技术，注意数据流、控制流中边界出错的可能性。

  如果对模块有性能要求的话，还要专门进行关键路径测试，以确定最坏情况下和平均意义下影响模块性能的因素。

  – 普通合法的数据是否正确处理
  – 普通非法的数据是否正确处理
  – 边界内最接近边界的（合法）数据是否正确处理
  – 边界外最接近边界的（非法）数据是否正确处理
  – N次循环的第0、1、n次是否有错
  – 运算或判断中取最大最小值时是否有错
  – 数据流、控制流中刚好等于、大于或小于确定的比较值时是否出错

单元测试的考虑 - 功能、性能、内存使用

- 功能

  应对软件设计文档规定的软件单元的功能逐项进行测试。

- 性能

  按软件设计文档的要求，对软件单元的性能（如精度、时间、容量等）进行测试。

- 内存使用

  检查内存的使用情况．特别是动态申请的内存在使用上的错误（如指针越界、内存泄露等）。

单元测试的文档

- 文档

  软件单元测试完成后形成的文档一般应有：

  a) 软件单元测试计划；

  b) 软件单元测试说明；

  c) 软件单元测试报告；

  d) 软件单元测试记录；

  e) 软件单元测试问题报告。

  可根据需要对上述文档及文档的内容进行裁剪。

集成测试

- 概述
- 集成测试的目的
- 集成测试的方式
- 集成测试内容

集成测试

- 术语
  - 集成测试

    也称为组装测试、联合测试，是将模块按照设计要求组装起来进行测试，主要目标是发现与接口有关的问题。
  - 部件测试

    子系统的组装测试特别称为部件测试，它所做的工作是要找出组装后的子系统与系统需求规格说明之间的不一致。
- 测试对象

  软件集成测试的对象包括：

  a) 任意一个软件单元集成到计算机软件系统的组装过程；

  b) 任意一个组装得到的软件系统。

集成测试

- 为什么进行集成测试？
  - 一个模块可能对另一个模块产生不利的影响
  - 将子功能合成时不一定产生所期望的主功能
  - 独立可接受的误差在组装后可能会超过可接受的误差限度
  - 可能会发现单元测试中未发现的接口方面的错误
  - 在单元测试中无法发现时序问题（实时系统）
  - 在单元测试中无法发现资源竞争问题

集成测试

- 集成测试的目的
  - 在模块组装后查找模块间接口的错误
  - 模块间数据交换是否丢失、有错（通讯、数据内容、时序等）
  - 各个子功能组合起来，能否达到预期要求的父功能；
  - 全局数据结构是否有问题；
  - 单个模块的误差累积，是否会放大，从而达到不能接受的程度。

在单元测试的同时可进行组装测试，发现并排除在模块连接中可能出现的问题，最终构成要求的软件系统。

集成测试

- 测试的组织和管理

  软件集成测试一般由软件供方组织并实施，测试人员与开发人员应相对独立；也可委托第三方进行软件集成测试。软件集成测试的工作产品一般应纳入软件的配置管理中。

  软件集成测试的技术依据是软件设计文档（软件结构设计文档）。待集成的软件单元已通过单元测试。

- 技术要求(可裁剪)

  软件集成测试一般应符合以下技术要求：

  a)应对已集成软件进行必要的静态测试，并先于动态测试进行；

  b) 软件要求的每个特性应被至少一个正常的测试用例和一个被认可的异常测试用例覆盖；

  c) 测试用例的输入应至少包括有效等价类值、无效等价类值和边界数据值；

  （continue）

集成测试

- 技术要求

d) 应采用增量法，测试新组装的软件；

e) 应逐项测试软件设计文档规定的软件的功能、性能等特性；

f) 应测试软件之间、软件和硬件之间的所有接口；

g) 应测试软件单元之间的所有调用，达到100%的测试覆盖率；

h) 应测试软件的输出数据及其格式；

i) 应测试运行条件（如数据结构、输入／输出通道容量、内存空间、调用频率等）在边界扶态下，进而在人为设定的状态下，软件的功能和性能；

j) 应按设计文档要求，对软件的功能、性能进行强度测试；

k)对完整性级别高的软件，应对某进行安全性分析，明确每一个危险状态和导致危险的可能原因，并对此进行针对性的测试。

集成测试

- 集成策略

  通常，把模块组装成为系统的方式有两种：

  – 一次性组装方式(big bang，大爆炸)

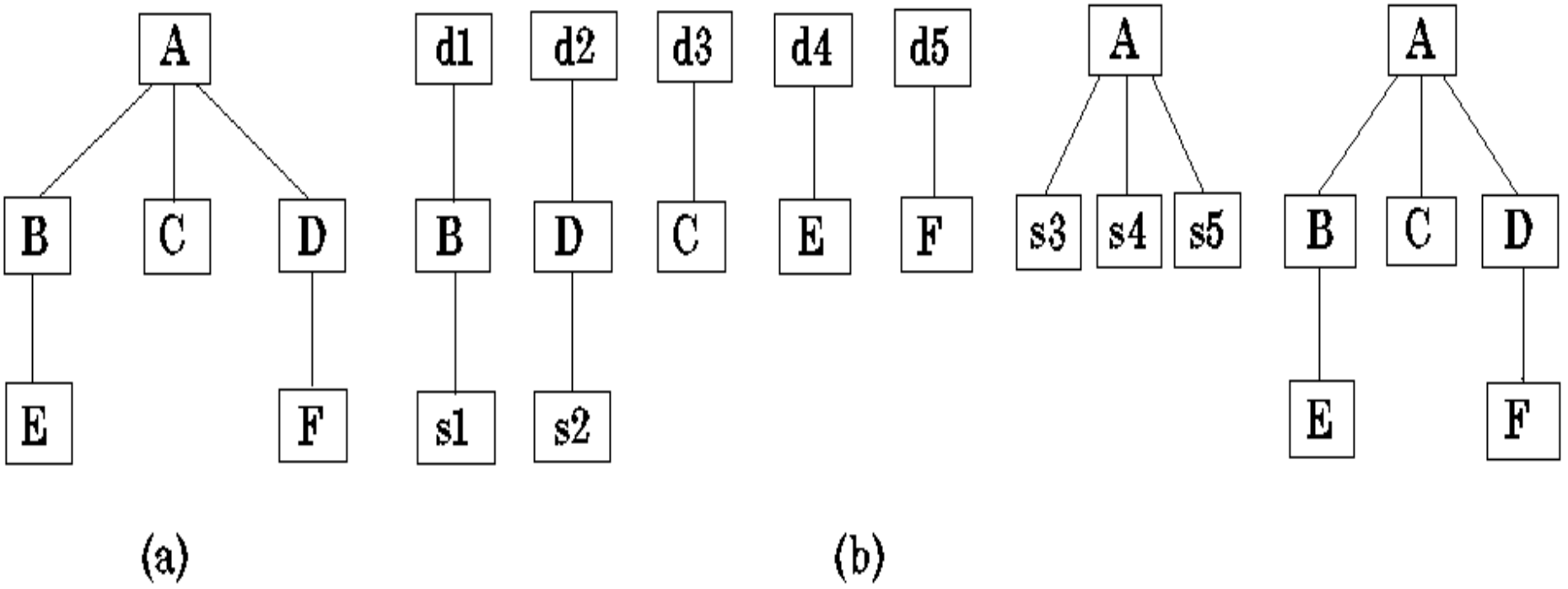
  – 增殖式组装方式

- 一次性组装方式

  它是一种非增殖式组装方式。也叫做整体拼装。

  非增式测试方法采用一步到位的方法来构造测试：对所有模块进行个别的单元测试后，按程序结构图将各模块联接起来，把联接后的程序当作一个整体进行测试。

集成测试 - 一次性组装方式



集成测试

- 一次性组装方式特点

  一次性组装方式的做法是先分散测试，再集中起来一次完成集成测试。因此其特点是：

  – 可并行调试所有模块，因此充分利用人力，加快工作进度。
  – 如果在模块的接口处存在差错，只会在最后的集成测试时一下子暴露出来。
  – 错误定位困难。

- 应避免一次性组装方式

  最坏情况下，甚至连单元测试都被省略。

  使用的必要条件：高度模块化且模块间的耦合极小，且详尽说明了接口且接口错较少时，可考虑使用一次性组装方式。

集成测试

- 增殖式组装方式
  - 这种组装方式又称渐增式组装
  - 首先对一个个模块进行模块测试，然后将这些模块逐步组装成较大的系统；
  - 在组装的过程中边连接边测试，以发现连接过程中产生的问题
  - 通过增殖逐步组装成为要求的软件系统。

  增式测试把单元测试与集成测试结合起来进行，将模块逐步集成起来，逐步完成集成测试。

集成测试 - 增殖式组装方式

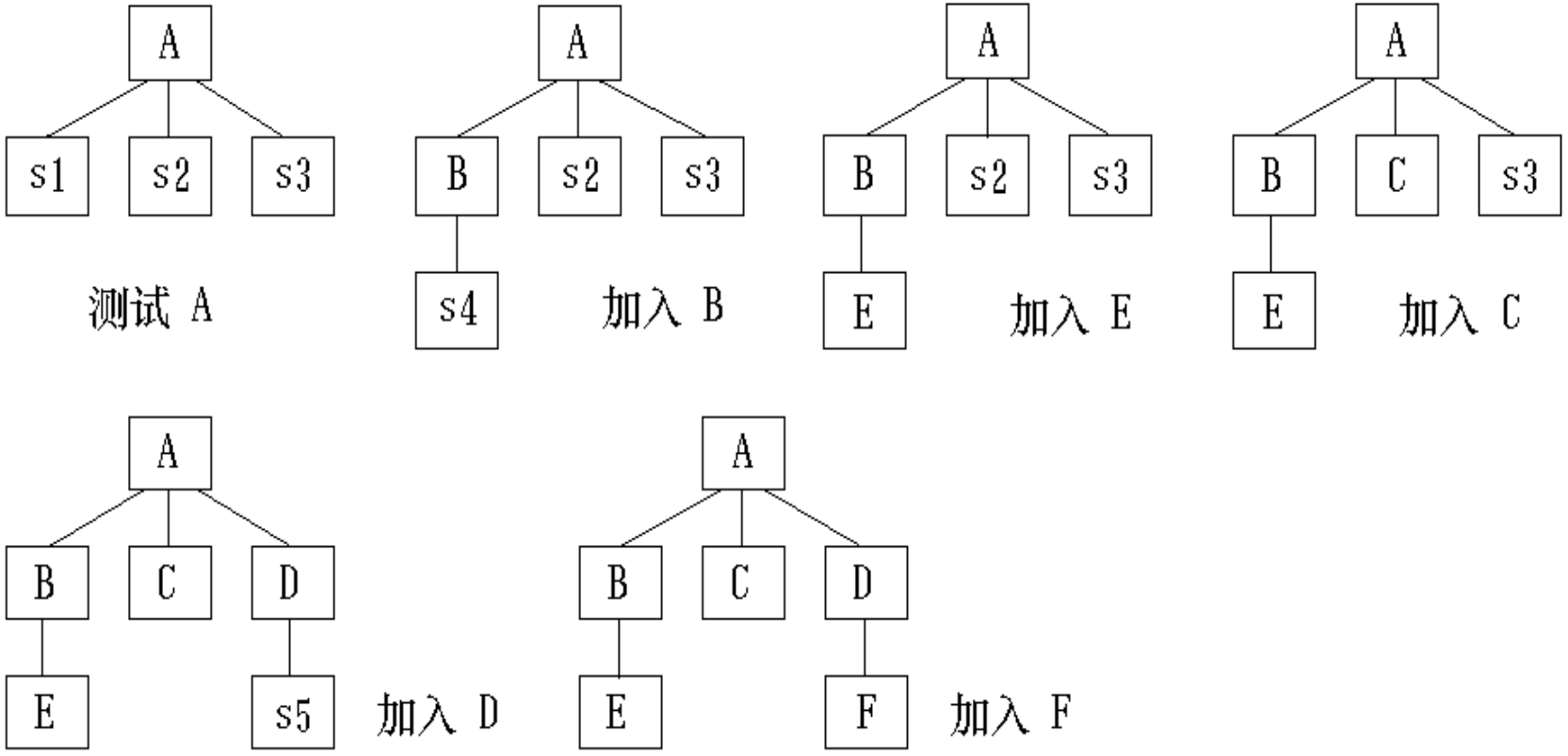
- 增殖式组装方式优点
  - 把可能出现的差错分散暴露出来，便于找出问题和修改 ；
  - 一些模块在逐步集成的测试中，得到了较为频繁的考验，因而可能取得较好的测试效果
- 实施方法：
  - 自顶向下结合
  - 自底向上结合
  - 混合增值式

自顶向下增式测试

- 集成步骤：
  - 主控模块作为测试驱动，所有与主控模块直接相连的模块用桩模块替换，并对主模块进行测试；
  - 根据集成的方式（深度或广度），每次用一个实际模块替换相应的桩模块；
  - 在每个模块被集成时，都必须已经进行了单元测试；
  - 进行回归测试以确定集成新模块后没有引入错误

  上述过程从第2步重复进行，直到整个系统结构被集成完成。

自顶向下增式测试



按深度方向组装的例子

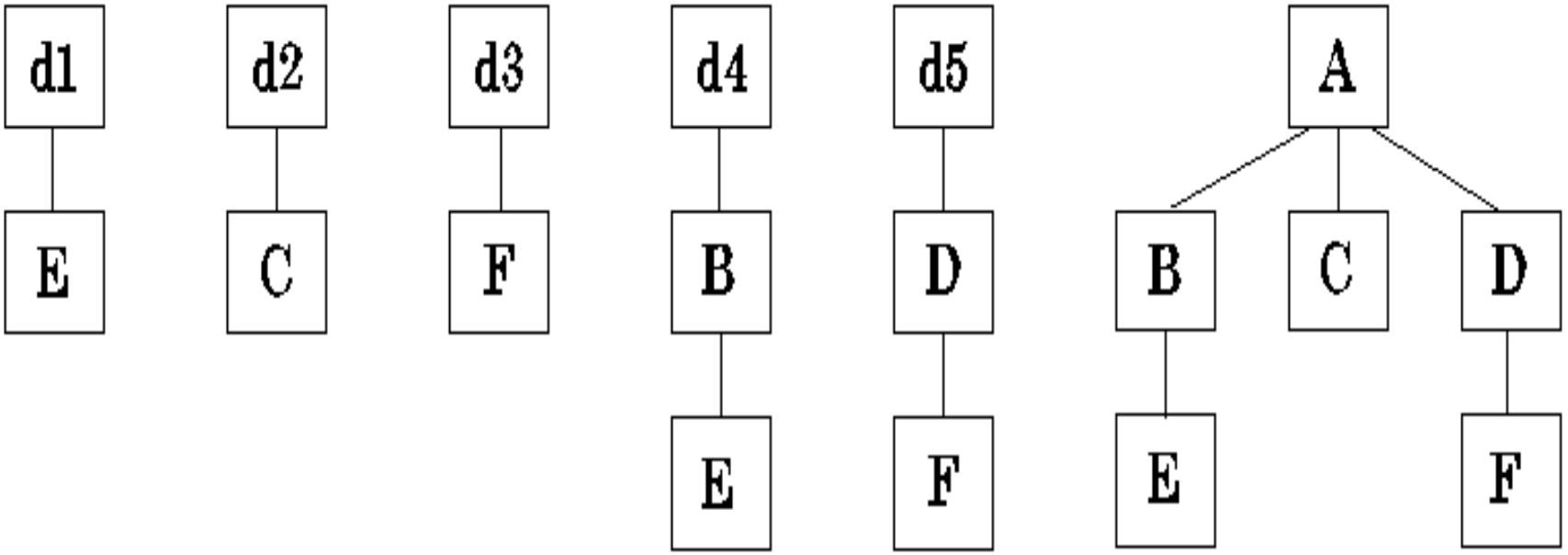
自顶向下增式测试

- 特点：
  - 这种组装方式将模块按系统程序结构，沿控制层次自顶向下进行组装。
  - 在测试过程中较早地验证了主要的控制和判断点。
  - 选用按深度方向组装的方式，可首先实现和验证一个完整的软件功能。功能的可行性较早地得到证实，给开发者和用户带来成功的信心。
  - 减少驱动模块开发费用
  - 支持故障隔离
  - 随着底层模块的不断增加，系统越来越复杂，易导致底层模块测试不充分。
  - 桩的开发和维护占据较大成本。

自底向上增式测试

- 工作程序：
  - 组装从最底层的模块开始，组合成一个构件，用以完成指定的软件子功能
  - 编制驱动程序，协调测试用例的输入与输出；
  - 测试集成后的构件
  - 按程序结构向上组装测试后的构件，同时去掉驱动程序

自底向上增式测试



自底向上增式测试

- 特点
  - 是从程序模块结构的最底层的模块开始组装和测试。
  - 桩模块开发成本减少，在模块的测试过程中需要从子模块得到的信息可以直接运行子模块得到。但有时为了模拟中断或异常，还要设计少量地桩模块。
  - 支持故障隔离
  - 允许对底层模块进行早期验证
  - 驱动模块的开发量较大
  - 高层的验证在最后，不易早期发现设计上、整体架构上的错误。

两种实施方法的比较

	优点	缺点
自顶向下测试	可以自然地做到逐步求精，一开始便能让测试者看到系统的框架	☐需要提供桩模块 ☐在输入/输出模块接入系统以前，在桩模块中表示测试数据有一定困难 ☐由于桩模块不能模拟数据，如果模块间的数据流不能构成有向的非环状图，一些模块的测试数据难于生成； ☐观察和解释测试输出往往也是困难的
自底向上测试	☐由于驱动模块模拟了所有调用参数，即使数据流并未构成有向的非环状图，生成测试数据也没有困难 ☐特别适合于关键模块在结构图的底部的情况	☐直到最后一个模块被加进去之后才能看到整个程序（系统）的框架 ☐只有到测试过程的后期才能发现时序问题和资源竞争问题

混合增殖式测试

- 自顶向下增殖的方式和自底向上增殖的方式各有优缺点。

  一般来讲，一种方式的优点是另一种方式的缺点。因此，通常是把以上两种方式结合起来进行组装和测试。下面简单介绍几种常见的综合的增殖方式：

  – 衍变的自顶向下的增殖测试
  – 自底向上–自顶向下的增殖测试
  – 回归集成测试

混合增殖式测试

- 衍变的自顶向下的增殖测试

  基本思想是强化关键模块的测试，并自底向上组装成子系统，然后由主模块开始自顶向下：

  – 首先对输入 / 输出模块和引入新算法模块进行测试;
  – 再自底向上组装成为功能相当完整且相对独立的子系统;
  – 然后由主模块开始自顶向下进行增殖测试。

混合增殖式测试

- 自底向上–自顶向下的增殖测试
  - 首先对含读操作的子系统自底向上直至根结点模块进行组装和测试;
  - 然后对含写操作的子系统做自顶向下的组装与测试。
- 回归集成测试
  - 这种方式采取自顶向下的方式测试被修改的模块及其子模块;
  - 然后将这一部分视为子系统，再自底向上测试，以检查该子系统与其上级模块的接口是否适配。

关键模块问题

在组装测试时，应当确定关键模块，对这些关键模块及早进行测试。

- 关键模块的特征：
  - ① 满足某些软件需求；
  - ② 在程序的模块结构中位于较高层次（高层控制模块）；
  - ③ 较复杂、较易发生错误；
  - ④ 有明确定义的性能要求。

在做回归测试时，也应该集中测试关键模块的功能。

集成测试内容

- 总则

  进行必要的静态测试。

  当动态测试时．从全局数据结构及软件的适合性、准确性、互操作性、容错性、时间特性、资源利用性这几个软件质量子特性方面考虑．确定测试内容。

- 全局数据结构

  可测试全局数据结构的完整性，包括数据的内容、格式．并对内部数据结构对全局数据结构的影响进行测试。

- 适合性方面

  应对设计文档分配给已集成软件的每一项功能逐项进行测试。

集成测试内容

- 准确性方面

  从准确性方面考虑．可对软件中具有准确性要求的功能和精度要求的项（如，数据处理精度、时间控制精度、时间测量精度）进行测试。

- 互操作性方面

  在互操作性方面，可考虑测试以下两种接口：

  1）所加入的软件单元与已集成软件之间的接口；

  2）已集成软件与支持其运行的其他软件、例行程序或硬件设备的接口。

  对接口的输入和输出数据的格式、内容、传递方式、接口协议等进行测试。

  测试 被测软件的控制信息，如：信号或中断的来源、目的、优先级、表示格式或表示值、最小、最大和平均频率，响应方式和响应时间等。

集成测试内容

- 容错性方面

  在容错性方面，可考虑测试已集成软件对差错输入、差错中断、漏中断等情况的容错能力，并考虑通过仿真平台或硬件测试设备形成一些人为条件，测试软件功能、性能的降级运行情况。

- 时间特性方面

  在时间特性方面，可考虑测试已集成软件的运行时间，算法的最长路径下的计算时间。

- 资源利用性方面

  在资源利用性方面，可考虑测试软件运行占用的内存空间和外存空间。

集成测试实施中的注意点

- 测试环境

  测试环境应包括测试的运行环境和测试工具环境。

  测试的运行环境一般应符合软件测试合同（或项目计划）的要求，通常是开发环境或仿真环境。

  测试工具一般要求是经过认可的工具

- 制定测试计划时，应考虑的因素

  1）采用何种系统组装方法来进行组装测试；

  2）组装测试过程中连接各个模块的顺序；

  3）代码编制和单元测试进度是否与组装测试的顺序一致

  4）测试过程中是否需要专门的硬件设备；

  5）测试所需软件（驱动模块、桩模块、测试用例生成程序等）的准备情况。

集成测试完成的标志

- 判定集成测试过程完成了,可按以下几个方面检查:
  1）成功地执行了测试计划中规定的所有集成测试;
  2）修正了所发现的错误;
  3）测试结果通过了专门小组的评审。

集成测试

- 文档

  软件集成测试完成后形成的文档一般应有：

  a) 软件集成测试计划；

  b) 软件集成测试说明；

  c) 软件集成测试报告：

  d) 软件集成测试记录和／或测试日志；

  e) 软件集成测试问题报告。

  可根据需要对上述文档及文档的内容进行裁剪。

确认测试

- 术语

  确认测试的任务是验证软件的功能和性能及其特性是否与用户的要求一致。

  对软件的功能和性能要求在软件需求规格说明中已经明确规定。它就是软件确认测试的基础。

- 测试对象

  是软件配置项。

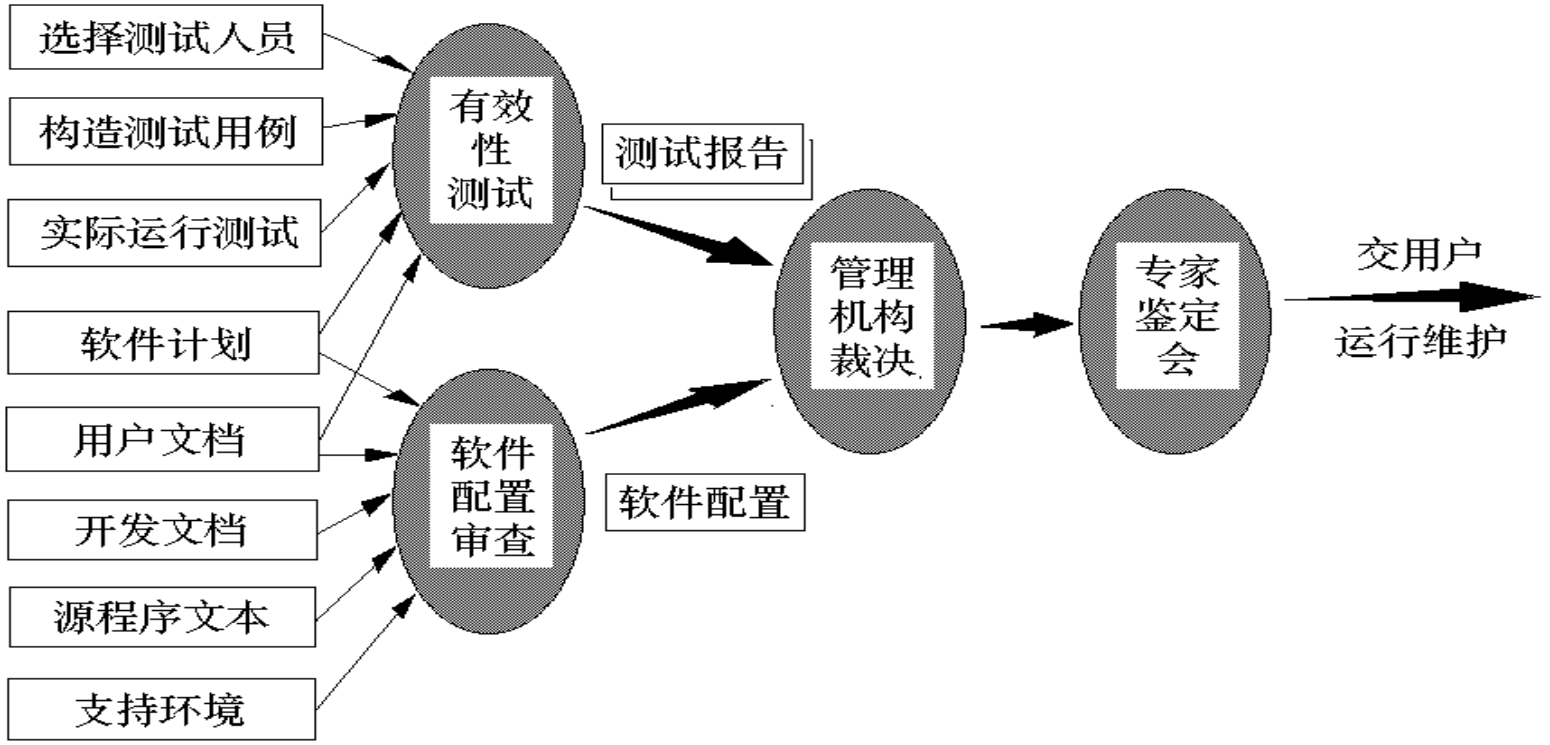
  软件配置项是为独立的配置管理而设计的并且能满足最终用户功能的一组软件。

- 测试目的

  目的是检验软件配置项与软件需求规格说明的一致性。

确认测试

- 确认测试包括有效性测试和软件配置审查。



确认测试

- 技术要求

  一般应符合以下技术要求：

  a) 必要时，在高层控制流图中作结构覆盖测试；

  b) 较件配置项的每个特性应至少被一个正常测试用例或一个被认可的异常测试用例所覆盖：

  c) 测试用例的输入应至少包括有效等价类值、无效等价类值和边界数据值；

  d) 应逐项测试软件需求规格说明规定的软件配置项的功能、性能等特性；

  e) 应测试软件配置项的所有外部输入、输出接口（包括和硬件之间的接口）；

  f) 应测试软件配置项的输出及其格式；

  g) 应按软件需求规格说明的要求，测试软件配置项的安全保密性，包括数据的安全保密性；

  （continue）

确认测试

- 技术要求（continue）

  h) 应测试人机交互界面提供的操作和显示界面，包括用非常规操作、误操作、快速操作测试界面的可靠性；

  i) 应测试运行条件在边界状态和异常状态下，或在人为设定的状态下，软件配置项的功能和性能：

  j) 应测试软件配置项的全部存储量、输入／输出通道和处理时间的余量；

  k) 应按需求规格说明的要求，对软件配置项的功能、性能进行强度测试；

  l) 应测试设计中用于提高软件配置项安全性、可靠性的结构、算法、容错、冗余、中断处理等方案；

  m) 对完整性级别高的软件配置项，应对其进行安全性分析，明确每一个危险状态和导致危险的可能原因，并对此进行针对性的测试；

  n) 对有恢复或重置功能需求的软件配置项，应测试其恢复或重置功能和平均恢复时间，并且对每一类导致恢复成重置的情况进行测试；

  o) 对不同的实际问题应外加相应的专门测试。

确认测试

- 测试内容

  GB/T15532-2008中，确认测试内容主要依据GB/T 16260.1规定的质量特性来进行，有别于传统的测试内容，测试内容主要从：功能性（适合性、准确性、互操作性、安全保密性）、可靠性（成熟性、容错性、易恢复性）、易用性（易理解性、易学性、易操作性、吸引性）、效率（时间特性、资源利用性）、维护性（易分析性、易改变性、稳定性、易测试性）、可移植性（适应性、易安装性、共存性、易替换性）和依从性等方面（可剪裁）来考虑。标准给出了各特性的测试内容。

  对具体的软件配置项，可根据软件合同（或项目计划）及软件需求规格说明的要求对标准给出的内容进行裁剪。

  下面是传统的有效性测试

确认测试 - 有效性测试

- 有效性测试（功能测试）

  任务

  验证被测软件是否满足需求规格说明书列出的需求。

  对软件的功能和性能要求在软件需求规格说明书中已经明确规定。它包含的信息就是软件确认测试的基础。

  测试内容

  大体可归为界面、数据、操作、逻辑、接口等几方面。

  - 程序安装、启动正常，有相应的提示框、错误提示等
  - 每项功能符合实际要求
  - 系统的界面清晰、美观
  - 菜单、按钮操作正常、灵活，能处理一些异常操作

（continue）

确认测试 - 有效性测试

测试内容

大体可归为界面、数据、操作、逻辑、接口等几方面。

- 能接受正确的数据输入，对异常数据的输入有提示、容错处理等
- 数据的输出结果准确，格式清晰，可以保存和读取
- 功能逻辑清楚，符合使用者习惯
- 系统的各种状态按照业务流程而变化，并保持稳定
- 支持各种应用的环境
- 能配合多种硬件周边设备
- 软件升级后，能继续支持旧版本的数据
- 与外部应用系统的接口有效

确认测试 - 有效性测试

- 有效性测试（功能测试）使用技术

  是在模拟的环境 (可能就是开发的环境) 下，运用黑盒测试技术：

  等价类划分法，边界值分析法，错误推测法，因果图法，组合分析法

确认测试 - 有效性测试

- 有效性测试结果

  在全部软件测试的测试用例运行完后，所有的测试结果可分为两类：

  1）测试结果与预期的结果相符。

  说明软件的功能或性能特征与需求规格说明书相符，从而软件被接受。

  2）测试结果与预期的结果不符。

  说明软件的功能或性能特征与需求规格说明不一致，这时需列一张软件各项缺陷表或软件问题报告，通过与用户的协商，解决所发现的缺陷和错误。

  这阶段发现的问题往往和需求分析阶段的差错有关，涉及面广，解决起来较困难。

确认测试 - 软件配置复查

- 软件配置复查

  是确认测试的另一个重要环节，其目的是保证：

  – 软件配置的所有成分都齐全；
  – 各方面的质量都符合要求；
  – 具有维护阶段所必需的细节；
  – 已经编排好分类的目录。

  软件配置复查的主要内容是文件资料。

- 文件资料：
  – 用户所需资料(用户手册、操作手册)
  – 设计开发资料(各开发阶段产品：需求规格说明书、设计说明书)
  – 源程序及测试资料(测试说明书，测试报告)

确认测试 - 软件配置复查

- 文档资料的完整性和正确性

  除了按合同规定的内容和要求，由人工审查软件配置之外，在确认测试的过程中，应当严格遵守用户手册和操作手册中规定的使用步骤，以便检查这些文档资料的完整性和正确性。必须仔细记录发现的遗漏和错误，并且适当地补充和改正。

- 确认测试完成后形成的文档

  a) 软件配置项测试计划；
  b) 软件配置项测试说明；
  c) 软件配置项测试报告；
  d) 软件配置项测试记录和/或测试日志；
  e) 软件配置项测试问题报告。

  可根据需要对上述文档及文档的内容进行裁剪

系统测试

- 术语

  系统测试，是将通过确认测试的软件，作为整个基于计算机系统的一个元素，与计算机硬件、外设、某些支持软件、数据和人员等其它系统元素结合在一起，在实际运行环境下，对计算机系统进行一系列的测试。

- 为什么要进行系统测试？

  由于软件只是计算机系统中的一个组成部分，软件开发完成之后，最终还要和系统中的硬件系统、某些支持软件、数据信息等其他部分配套运行。因此，在投入运行前要完成系统测试，以保证各组成部分不仅能单独的得到检验，而且在系统各部分协调工作的环境下也能正常工作。

系统测试

- 测试对象

  系统测试的对象是完整的、集成的计算机系统．重点是新开发的软件配置项的集合。

- 测试目的

  在于通过与系统的需求定义作比较，发现软件与系统的定义不符合或与之矛盾的地方。

  系统测试的测试用例应根据需求分析规格说明来设计，并在实际使用环境下来运行。

系统测试

- 意义

保证各组成部分不仅能单独地受到检验，而且在系统各部分协调工作的环境下也能正常工作。

系统组成部分：除去软件外，还可能包括计算机硬件及其相关的外围设备、数据及其收集和传输机构、掌握计算机系统运行的人员及其操作等，甚至还可能包括受计算控制的执行机构。

显然，系统测试已超出了软件工作的范围。其意义：

1）软件在系统中占有重要的位置，软件测试的好坏与能否顺利、成功地完成系统测试关系极大。

2）系统测试实际上是针对系统中各个组成部分进行的综合性检验。尽管每一个检验有着特定的目标，然而所有的检测工作都要验证系统中每个部分均已得到正确的集成，并能完成指定的功能。

系统测试

- 技术要求

  系统测试一般应符合以下技术要求：

  a) 系统的每个特性应至少被一个正常测试用例和一个被认可的异常测试用例所覆盖：

  b) 测试用例的输入应至少包括有效等价类值、无效等价类值和边界数据值；

  c) 应逐项测试系统／子系统设计说明规定的系统的功能、性能等特性：

  d) 应测试软件配置项之间及软件配置项与硬件之间的接口；

  e) 应测试系统的输出及其格式；

  f) 应测试运行条件在边界状态和异常状态下，或在人为设定的状态下，系统的功能和性能；

  g) 应测试系统访问和数据安全性；

系统测试

- 技术要求

h) 应测试系统的全部存储量、输入 / 输出通道和处理时间的余量；

i) 应按系统或子系统设计文档的要求，对系统的功能、性能进行强度测试；

j) 应测试设计中用于提高系统安全性、可靠性的结构、算法、容错、冗余、中断处理等方案；

k) 对完整性级别高的系统，应对其进行安全性、可靠性分析，明确每一个危险状态和导致危险的可能原因，并对此进行针对性的测试；

l) 对有恢复或重置功能需求的系统，应测试其恢复或重置功能和平均恢复时间，并且对每一类导致恢复或重置的情况进行测试；

m) 对不同的实际问题应外加相应的专门测试。

系统测试

- 系统测试的内容

  GB/T15532-2008中，系统测试内容主要依据GB/T 16260.1规定的质量特性来进行，有别于传统的测试内容，测试内容主要从：功能性、可靠性、易用性、效率、维护性、可移植性和依从性等方面（可剪裁）来考虑。

- 功能性

  1 适合性方面

  从适合性方面考虑，应测试系统 / 子系统设计文档规定的系统的每一项功能。

  2 准确性方面

  从准确性方面考虑，可对系统中具有准确性要求的功能和精度要求的项（如数据赴理精度、时间控制精度、时间测量精度）进行测试。

系统测试

- 功能性

3 互操作性方面

从互操作性方面考虑，可测试系统 / 子系统设计文档、接口需求规格说明文档和接口设计文档规定的系统与外部设备的接口、与其他系统的接口。测试其格式和内容，包括数据交换的数据格式和内容；测试接口之间的协调性；测试软件对系统每一个真实接口的正确性；测试软件系统从接口接收和发送数据的能力；测试数据的约定、协议的一致性；测试软件系统对外围设备接口特性的适应性。

4 安全保密性方面

从安全保密性方面，可测试系统及其数据访问的可控制性。

测试系统防止非法操作的模式，包括防止非授权的创建、删除或修改程序或信息，必要时做强化异常操作的测试。

测试系统防止数据被讹误和被破坏的能力。

测试系统的加密和解密功能

系统测试

- 可靠性

  1 成熟性方面

  在成熟性方面，可基于系统运行剖面设计测试用例，根据实际使用的概率分布随机选择输入，运行系统，测试系统满足需求的程度并获取失效数据，其中包括对重要输入变量值的覆盖、对相关输入变量可能组合的覆盖、对设计输入空间与实际输入空间之间区域的覆盖、对各种使用功能的覆盖、对使用环境的覆盖。应在有代表性的使用环境中、以及可能影响系统运行方式的环境中运行软件，验证系统的可靠性需求是否正确实现。对一些特殊的系统，如容错软件、实时嵌入式软件等，由于在一般的使用环境下常常很难在软件中植入差错，应考虑多种测试环境。

  测试系统的平均无故障时间。

  选择可靠性增长模型．通过检测到的失效数和故障数，对系统的可靠性进行预测。

系统测试

- 可靠性

  2 容错性方面

  从容错性方面考虑，可测试：

  a) 系统对中断发生的反应：

  b) 系统在边界条件下的反应：

  c) 系统的功能、性能的降级情况；

  d) 系统的各种误操作模式：

  e) 系统的各种故障模式（如数据超范围、死锁）；

  f) 测试在多机系统出现故障需要切换时系统的功能和性能的连续平稳性。

  注：可用故障树分析技术检测误操作模式和故障模式。

系统测试

- 可靠性

  3 易恢复性方面

  从易恢复性方面考虑，可测试：

  a) 具有自动修复功能的系统的自动修复的时间；
  b) 系统在特定的时间范围内的平均宕机时间；
  c) 系统在特定的时间范围内的平均恢复时间；
  d) 系统的可重启动并继续提供服务的能力；
  e) 系统的还原功能的还原能力。

系统测试

- 易用性

1 易理解性方面

从易理解性方面考虑，可测试：

a) 系统的各项功能，确认它们是否容易被识别和被理解；

b) 要求具有演示能力的功能，确认演示是否容易被访问、演示是否充分和有效；

c) 界面的输入和输出，确认输入和输出的格式和含义是否容易被理解。

2 易学性方面

从易学性方面考虑，可测试系统的在线帮助，确认在线帮助是否容易定位，是否有效；还可对照用户手册或操作手册执行系统，测试用户文档的有效性。

系统测试

- 易用性

3 易操作性方面

从易操作性方面考虑，可测试：

a) 输入数据，确认系统是否对输入数据进行有效性检查；

b) 要求具有中断执行的功能．确认它们能否在动作完成之前被取消：

c) 要求具有还原能力（数据库的事务回滚能力）的功能，确认它们能否在动作完成之后被撤消；

d)包含参数设置的功能，确认参数是否易于选择、是否有缺省值；

e) 要求具有解释的消息，确认它们是否明确；

f) 要求具有界面提示能力的界面元素，确认它们是否有效；

g) 要求具有容错能力的功能和操作，确认系统能否提示出错的风险、能否容易纠正错误的输入、 能否从差错中恢复：

（continue）

系统测试

- 易用性

3 易操作性方面

h) 要求具有定制能力的功能和操作，确认定制能力的有效性；

i) 要求具有运行状态监控能力的功能，确认它们的有效性。

注：以正确操作、误操作模式、非常规操作模式和快速操作为框架设计测试用例，误操作模式有错误的数据类型作参数、错误的输入数据序列、错误的操作序列等。如有用户手册或操作手册，可对照手册逐条进行测试。

4 吸引性方面

从吸引性方面考虑，可测试系统的人机交互界面能否定制。

系统测试

- 效率

  1 时间特性方面

  从时间特性方面考虑，可测试系统的响应时间、平均响应时间、响应极限时间．系统的吞吐量、平均吞吐量、极限吞吐量，系统的周转时间、平均周转时间、周转时间极限。（这些术语定义见下页）

  在测试时，应标识和定义适合于软件应用的任务．并对多项任务进行测试．而不是仅测一项任务。

  软件应用任务的例子，如在通信应用中的切换、数据包发送，在控制应用中的事件控制，在公共用户应用中由用户调用的功能产生的一个数据的输出等。

效率方面的术语

响应时间：指系统为完成一项规定任务所需的时间：

平均响应时间：指系统执行若干并行任务所需的平均时间；

响应极限时间：指在最大负载条件下，系统完成某项任务需要时间的极限；

吞吐量：指在给定的时间周期内系统能成功完成的任务数量：

平均吞吐量：指在一个单位时间内系统能处理并发任务的平均数：

极限吞吐量：指在最大负载条件下．在给定的时间周期内，系统能处理的最多并发任务数；

周转时间：指从发出一条指令开始到一 组相关的任务完成的时间；

平均周转时间：指在一个特定的负载条件下，对一些并发任务．从发出请求到任务完成所需要的平均时间；

周转时间极限：指在最大负载条件下，系统完成一项任务所需要时间的极限。

系统测试

- 效率

2 资源利用性方面

可测试系统的输入/输出设备、内存和传输资源的利用情况：

a) 执行大量的并发任务，测试输入 / 输出设备的利用时间；

b) 在使输入 / 输出负载达到最大的系统条件下，运行系统，测试输入，输出负载极限；

c) 并发执行大量的任务，测试用户等待输入 / 输出设备操作完成需要的时间；

注：建议调查几次测试与运行实例中的最大时间与时间分布。

d) 在规定的负载下和在规定的时间范围内运行系统，测试内存的利用情况；

e)在最大负载下运行系统，测试内存的利用情况；

（continue）

系统测试

- 效率

  2 资源利用性方面

  f) 并发执行规定的数个任务．测试系统的传输能力；

  g) 在系统负载最大的条件下和在规定的时间周期内，测试传输资源的利用情况；

  h) 在系统传输负载最大的条件下，测试不同介质同步完成其任务的时间周期。

- 维护性

  1 易分析性方面

  从易分析性方面考虑，可设计各种情况的测试用例运行系统，并监测系统运行状态数据，检查这些数据是否容易获得、内容是否充分。如果软件具有诊断功能，应测试该功能。

系统测试

- 维护性

  2 易改变性方面

  从易改变性方面考虑，可测试能否通过参数来改变系统。

  3 稳定性方面

  （暂不推荐软件稳定性方面的测试内容。）

  4 易测试性方面

  从易测试性方面考虑，可测试软件内置的测试功能，确认它们是否完整和有效。

系统测试

- 可移植性

  1 适应性方面

  从适应性方面考虑，可测试：

  a) 软件对诸如数据文件、数据块或数据库等数据结构的适应能力；

  b) 软件对硬件设备和网络设施等硬件环境的适应能力；

  c) 软件对系统软件或并行的应用软件等软件环境的适应能力；

  d) 软件是否易于移植。

系统测试

- 可移植性

2 易安装性方面

从易安装性方面考虑，可测试软件安装的工作量、安装的可定制性、安装设计的完备性、安装操作的简易性、是否容易重新安装。

注1：安装设计的完备性可分为三级：

a)最好：设计了安装程序，并编写了安装指南文档；

b) 好：仅编写了安装指南文档；

c) 差：无安装程序和安装指南文档。

注2：安装操作的简易性可分为四级：

a) 非常容易：只需启动安装功能并观察安装过程：

b) 容易：只需回答安装功能中提出的问题；

c) 不容易：需要从表或填充框中看参数；

d) 复杂：需要从文件中寻找参数，改变或写它们。

系统测试

- 可移植性

  3 共存性方面

  从共存性方面考虑，可测试软件与其他软件共同运行的情况。

  4 易替换性方面

  当替换整个不同的软件系统和用同一软件系列的高版本替换低版本时，在易替换性方面，可考虑测试：

  a) 软件能否继续使用被其替代的软件使用过的数据；

  b) 软件是否具有被其替代的软件中的类似功能。

- 依从性方面

  当软件在功能性、可靠性、易用性、效率、维护性和可移植性方面遵循了相关的标准、约定、风格指南或法规时，应酌情进行测试。

系统测试

以上基于特性的系统测试对应到传统的测试类型一般有：

功能测试、性能测试、负载测试、压力测试（强度测试）、疲劳测试、易用性（用户界面）测试、安装与卸载测试、配置测试、文档测试、安全性测试、恢复测试、回归测试、健壮性测试、交付测试（稳定期测试）、演练测试、背靠背测试、度量测试、比较测试等。

上述测试最好由第三方进行。

系统测试

- 文档

  系统测试完成后形成的文档一般应有：

  a) 系统测试计划；
  b) 系统测试说明；
  c) 系统测试报告：
  d) 系统测试记录和／或测试日志；
  e) 系统测试问题报告。

验收测试（Acceptance Testing）

- 验收测试概述、目的
- 验收测试与系统测试的区别
- 验收测试的时机、范围
- 验收测试主要内容
- 验收测试过程
- 验收测试形式

验收测试（Acceptance Testing）

- 验收测试(也称为交付测试)

  检查软件能否按合同要求进行工作，即是否满足软件合同中的确认标准。

- 测试对象

  是完整的、集成的计算机系统。

- 测试目的

  验收测试的目的是在真实的用户（或称系统）工作环境下检验完整的软件系统．是否满足软件开发技术合同（或软件需求规格说明）规定的要求。

  其结论是软件的需方确定是否接收该软件的主要依据。

验收测试（Acceptance Testing）

- 测试的组织和管理

  验收测试应由软件的需方组织，由独立于软件开发的人员实施。如果验收测试委托第三方实施．～一般应委托国家认可的第三方测试机构。

  应加强验收测试的配置管理，已通过测试的验收状态和各项参数应详细记录，归档保存．未经测试负责人允许，任何人无权改变。

  软件验收测试的技术依据是软件研制合同（或用户需求或系统需求）。

- 技术要求、测试内容

  同系统测试。

验收测试（Acceptance Testing）

- 验收测试与系统测试的区别
  - 组织机构：
  - 测试地点：
  - 覆盖范围：
  - 实施人员：
- 验收测试的时机
  - 在通过系统测试后。
  - 商业现货软件产品可在安装或集成时进行。
  - 组件的可用性验收测试可在组件测试时进行。

验收测试（Acceptance Testing）

- 验收测试的范围

  验收测试范围广，取决于被测程序。

  – 如是用户定制的开发软件，需全面验收测试。

  – 如是一个标准产品，该产品已在类似环境运行了很长时间，则验收测试包括系统安装，运行一些典型用例（检测主要功能、性能）。

  – 如是系统集成，重点测互操作性

验收测试（Acceptance Testing）

- 文档

  验收测试完成后的文档一般直有：

  a) 验收测试计划；

  b) 验收测试说明；

  c) 验收测试报告：

  d) 验收测试记录：

  e) 验收测试问题报告。

  可根据需要对上述文档及文档的内容进行裁剪。

回归测试

- 什么是回归测试？
  - 在集成测试策略的环境中，回归测试是对某些已经进行过的测试的某些子集再重新进行一遍，以保证上述改变不会传播无法预料的副作用或引发新的问题。
  - 在更广的环境里，回归测试就是用来保证（由于测试或其他原因的）改动不会带来不可预料的行为或另外的错误。

回归测试

- 测试对象

  回归测试的对象包括：

  a) 未通过单元测试的软件，在变更之后，应对其进行单元测试：

  b) 未通过配置项测试的软件，在变更之后，首先应对变更的软件单元进行测试，然后再进行相关的集成测试和配置项测试；

  c) 未通过系统测试的软件，在变更之后，首先应对变更的软件单元进行测试．然后再进行相关的集成测试、软件配置项和系统测试：

  d) 因其他原因进行变更之后的软件单元，也首先应对变更的软件单元进行测试，然后再进行相关的软件测试。

回归测试

- 测试目的

  回归测试的测试目的是：

  a) 测试软件变更之后，变更部分的正确性和对变更需求的符合性：

  b) 测试软件变更之后，软件原存的、正确的功能、性能和其他规定的要求的不损害性。

- 单元回归测试
  - 技术要求

    一股应符合原软件单元测试的技术要求，可根据变更情况酌情裁剪。当回归测试结果和原软件单元测试的正确结果不一致时，应对软件单元重新进行回归测试。

回归测试

- 单元回归测试
  - 测试内容

    一般应根据软件单元的变更情况确定软件单元回归测试的测试内容。

    a) 仅重复测试原软件单元测试做过的测试内容；

    b) 修改原软件单元测试做过的测试内容；

    c) 在前两者的基础上增加新的测试内容。
  - 测试方法

    当未增加新的测试内容时，软件单元回归测试应采用原软件单元测试的测试方法；否则，应根据情况选择适当的测试方法；

回归测试

- 单元回归测试
  - 准入条件

    进入单元回归测试一般应县备以下条件：

    a) 被测软件单元完成变更且已经置于软件配臀管理之下；

    b) 软件变更报告单齐全；

    c) 具有测试相关的全部文档及资源；

    d) 具备相关测试的设施环境。
  - 准出条件

    软件单元回归测试的准出条件用来评价软件单元同归测试的工作是否达到要求。软件单元回归测试的准出条件与原软件单元测试的准出条件一致；

    另外，软件单元回归测试的文档应齐全、符合规范。

回归测试

- 单元回归测试
  - 文档

    a) 软件单元回归测试计划；

    b) 软件单元回归测试说明；

    c) 软件单元回归测试报告：

    d) 软件单元回归测试记录；

    e) 软件单元回归测试问题报告。

    可根据需要对上述文档及文档的内容进行裁剪。

    上述文档也可分别作为软件单元测试产生的文档的补充件。

回归测试

- 配置项回归测试
  - 技术要求

    软件配置项回归测试的技术要求一般应符合以下原则：

    a) 对变更的软件单元的测试，应符合原软件单元测试的技术要求，可根据变更情况进行裁剪；

    b) 对变更的软件单元和受变更影响的软件进行集成的测试，应符合原软件集成测试的技术要求，可根据受影响情况进行裁剪；

    c) 对变更后的软件配置项的测试，应符合原软件配置项测试的技术要求．可根据受影响情况进行 裁剪；

    d) 当回归测试结果和原软件单元测试、软件集成测试和软件配置项测试的正确结果不一致时，应对出现问题的软件单元、受该单元影响的已集成软件和软件配置项重新进行回归测试。

回归测试

- 配置项回归测试
  - 测试内容

    软件配置项回归测试的测试内容分三种情况考虑：

    a) 对变更的软件单元的测试可能存在以下三种情况：一是仅重复测试原软件单元测试做过的测试内容：二是修改原软件单元测试做过的测试内容：三是在前两者的基础上增加新的测试内容；

    b) 对于变更的软件单元和受变更影响的软件进行集成的测试，测试分忻员应分析变更对软件集成的影响域，并据此确定回归测试内容。可能存在以下三种情况：一是仅重复测试与变更相关的、并已在原软件集成测试中做过的测试内容；二是修改与变更相关的、并已在原软件集成测试中做过的测试内容；三是在前两者的基础上增加新的测试内容：

回归测试

- 配置项回归测试
  - 测试内容

    软件配置项回归测试的测试内容分三种情况考虑：

    C) 对于变更后的软件配置项的测试，测试分析员应分析变更对软件配置项的影响域．并据此确定回归测试内容。可能存在以下三种情况：一是仅重复测试与变更相关的、并已在原软件配置项测试中做过的测试内容：二是修改与变更相关的、并已在原软件配置项测试中做过的测试内容；三是在前两者的基础上增加新的测试内容。

回归测试

- 配置项回归测试
  - 测试环境

    软件配置项回归测试的测试环境要求分三种情况：

    a) 对于变更的软件单元的测试．其测试环境要求应与原软件单元测试的测试环境要求一致：

    b) 对于变更的软件单元和受变更影响的软件进行集成的测试，其测试环境要求应与原软件集成 测试的测试环境要求一致：

    c) 对于变更后的软件配置项的测试，其测试环境要求应与原软件配置项测试的测试环境要求 一致。

回归测试

- 配置项回归测试
  - 测试方法

    软件配置项回归测试不排除使用标准测试集和经认可的系统功能测试方法。是重复软件配置项开发各阶段的相关工作的方法。这种方法分三种情况：

    a) 对于变更的软件单元的测试．当未增加新的测试内容时，对变更的软件单元的测试采用原单元测试的测试方法；否则，根据情况选择适当的测试方法。

    b) 对于变更的软件单庇和受变更影响的软件进行集成的测试，当未增加新的测试内容时．对受影响的软件集成测试采用原软件集成测试的测试方法：否则，根据情况选择适当的测试方法。

    c) 对于变更后的软件配置项的测试．当呆增加新的测试内容时，对软件配茂项的测试采用j泉软件配置项测试的测试方法：否则．根据情况选择适当的测试方法。

回归测试

- 配置项回归测试
  - 准入条件

    进入配置项回归测试，一般应县备以下条件：

    a) 被测软件完成变更且已经置于软件配置管理之下；

    b) 相关的软件测试报告、软件变更报告单齐全；

    c) 具有相关测试的全部文档及资源；

    cl) 具备相关测试的设施环境。
  - 准出条件

    软件配置项回归测试的准出条件用来评价回归测试的工作是否达到要求，一般应符合以下原则：

    a) 按照软件集成测试的要求（见前），完成了对变更的和受变更影响的软件的集成测试．并且无新问题出现：

回归测试

- 配置项回归测试
  - 准出条件

b) 对变更的软件配置项的回归测试应符合原软件配置项测试的准出条件，并且无新问题出现。

另外，软件配置项回归测试的史档应齐全、符合规范。

  - 文档

a) 软件配置项同归测试计划：

b) 软件配置项同归测试说明：

c) 软件配置项回归测试报告；

d) 软件配置项回归测试记录和或测试日志：

e) 软件配置项回归测试问题报告。

可裁剪。 上述文档也可分别作为软件单元测试、软件集成测试和软件配置项测试产生的文档的补充件。

回归测试

- 系统回归测试
  - 技术要求

    a) 对变更的软件单元的测试，应符合原软件单元测试的技术要求，可根据变更情况进行裁剪；

    b) 对变更的软件单元和受变更影响的软件进行集成的测试，应符合原软件集成测试的技术要求，可根据受影响情况进行裁剪；

    c) 对变更的和受变更影响的软件配置项的测试，应符合原软件配置项测试的技术要求，可根据受影响情况进行裁剪：

    d) 对变更的系统的测试．应符合原系统测试的技术要求．可恨据受影响情况进行裁剪；

    e) 当回归测试结果和原软件单元测试、软件集成测试、软件配置项测试和系统测试的正确结果不 一致时，应对出现问题的软件单元和受该单元影响的已集成软件、软件配置项和系统重新进行回归测试。

回归测试

- 系统回归测试
  - 测试内容

分四种情况考虑：

a) 对于变更的软件单元的测试的三种情况。

b) 对于变更的软件单元和受变更影响的软件进行集成的测试，测试分析员应分析变更的软件单元对软件集成的影响域．并据此确定回归测试内容。

c) 对于变更的和受变更影响的软件配置项的测试．测试分析员应分析变更对软件配置项的影响域，并据此确定回归测试内容。

d) 对于变更的系统的测试，测试分析员应分析软件系统受变更影响的范围，并据此确定同归测试内容。

回归测试

- 系统回归测试
  - 测试环境

    系统回归测试的测试环境要求分四种情况：

    a) 对于变更的软件单元的测试，其测试环境要求应与原软件单元测试的测试环境要求一致；

    b) 对于变更的软件单元和受变更影响的软件进行集成的测试，其测试环境要求应与原软件集成测试的测试环境要求一致；

    c) 对于变更的和受变更影响的软件配置项的测试．其测试环境要求麻与原软件配置项测试的测试环境要求一致：

    d) 对于变更的系统的测试，其测试环境要求应与原系统测试的测试环境要求一致。

回归测试

- 系统回归测试
  - 测试方法

    系统回归测试不排除使用标准测试集和经认可的系统功能测试方法。是重复软件系统开发各阶段的相关工作的方法：这种测试方法分四种情况：

    a) 对于变更的软件单元的测试；

    b) 对于变更的软件单元和受变更影响的软件进行集成的测试；

    c) 对于变更的和受变更影响的软件配置项的测试；

    d) 对于变更的系统的测试。

回归测试

- 系统回归测试
  - 文档

    系统回归测试完成后形成的文档一般应有：

    a) 系统回归测试汁划：

    l，) 系统回归测试说明；

    c) 系统回归测试报告：

    d) 系统回归测试记录和／或测试曰志；

    e) 系统回归测试问题报告。

    可根据需要对上述文档及文档的内容进行裁剪。

    上述文档也可分别作为软件单元测试、软件集成测斌、软件配置项测试和系统测试产生的文档的补充件。