软件测试

张程
Email：bootan@cqu.edu.cn
QQ:80463125

自动化测试

概述

- 自动化测试是把以人为驱动的测试行为转化为机器执行的一种过程。通常，在设计了测试用例并通过评审之后，由测试人员根据测试用例中描述的规程一步步执行测试，得到实际结果与期望结果的比较。在此过程中，为了节省人力、时间或硬件资源，提高测试效率，便引入了自动化测试的概念
- Bret Pettichord 在<<自动化测试的7个步骤>>中说：“我们对自动化测试充满了希望，然而，自动化测试却经常带给我们沮丧和失望；虽然，自动化测试可以把我们从困难的环境中解放出来，在实施自动化测试解决问题的同时，又带来同样多的问题”
- 自动化测试节省人力，节省时间，得到的数据更精确些，而且操作的可重复性和Bug 的可重现性更强一些，而软件行业的测试有节约成本，提高效率的需求

手工测试的优劣

- 优点
  - 测试用例的设计
  - 界面和用户体验测试
  - 逻辑判断的正确性检查
- 局限性
  - 通过手工测试无法做到覆盖所有代码路径;
  - 许多与时序、死锁、资源冲突、多线程等有关的错误通过手工测试很难捕捉到
  - 在系统负载、性能测试时，需要模拟大量数据、或大量并发用户等各种应用场合时，也很难通过手工测试来进行
  - 在进行系统可靠性时，需要模拟系统运行十年、几十年，以验证系统能否稳定运行，也是手工测试无法模拟的。
  - 如果有大量（几千）的测试用例，需要在短时间内完成，手工测试又怎么办呢?
  - 测试可以发现错误，并不能表明程序的正确性

自动化测试的优点

- 能执行更多更频繁的测试，使某些测试任务的执行比人工测试更高效，缩短软件开发测试周期，可以更快地将软件推向市场；
- 更好地利用资源，利用整夜或周末空闲的设备执行自动化测试,测试效率高，可以充分利用硬件资源；
- 可以对程序的新版本自动执行回归测试；
- 可以执行一些人工测试有困难或不可能进行的测试如负载、性能测试；
- 自动测试具有一致性和可重复性的特点，而且测试更客观，提高了软件的信任度；
- 增强测试的稳定性和可靠性；
- 提高软件测试的准确度和精确度，增加软件信任度；
- 将任务自动化，让测试人员投入更多的精力设计出更多更好的测试用例，提高测试准确性和测试人员的积极性。

手工测试与自动化测试的情况比较

测 试 步 骤	手工测试（小时）	自动测试（小时）	改进百分率（使用工具）
测试计划制定	32	40	–25%
测试程序开发	262	117	55%
测试执行	466	23	95%
测试结果分析	117	58	50%
错误状态/纠正监视	117	23	80%
报告生成	96	16	83%
总持续时间	1090	277	75%

自动化测试的缺点

- 自动化测试不能完全代替人工测试;
- 不能立即降低测试投入，提高测试效率;
- 不能保证100%的测试覆盖率;
- 需要更长的时间去分析和隔离所发现的缺陷;
- 一种测试工具不完全适用于所有测试;
- 自动测试不一定减轻工作量;
- 测试进度可能不一定缩短;
- 自动化测试的普遍应用存在局限;
- 工具本身并没有想象力和灵活性，根据报道，自动化测试只能发现15%的缺陷，而人工测试可以发现85%的缺陷;
- 自动化测试工具在进行功能测试时，其准确的含义是回归测试工具，这时工具不能发现更多的新问题，但可以保证对已经测试过部分的准确性和客观性;
- 自动化测试不能提高测试的有效性,只是用于提高测试的效率，即减少人工测试的时间;

自动化测试的缺点（续）

- 自动化测试不具有想象力
  - *自动化测试是通过测试软件进行，自动测试工具本身不具有想象力，测试过程只是按照运行命令执行。而人工测试时可以直接判断测试结果的正确性，而自动测试许多情况下测试结果还需要人工干预判断。*
  - *人工测试可以处理意外事件，如网络连接中断，此时必须重新建立连接。手工测试时可以及时处理该意外，而自动化测试时该意外事件一般都会导致测试的中止。*
- 自动化测试不利于测试人员积累测试经验；
- 自动测试对测试质量的依赖性较大，在确保测试质量的前提下，实施自动化测试才有意义；
- 自动测试在刚开始执行时，工作效率并不一定高于手动测试，只有当整个自动测试系统成熟，且测试工程师熟练掌握测试工具后，工作效率才会随着测试执行次数的增加而提高。
- 自动测试的成本可能高于人工测试。自动测试的成本大致有以下几个部分组成：
  - *自动测试开发成本；*
  - *自动测试运行成本；*
  - *自动测试维护成本和其他相关任务带来的成本；*
  - *软件的修改带来测试脚本部分或全部修改，就会增加测试维护的开销。*

对测试自动化的误解

- 希望将所有测试活动自动化。
- 购买一个测试工具，满足所有自动化测试需求。
- 马上减轻测试工作负担。
- 在缩短进度上立竿见影。
- 期望自动产生测试计划。
- 达到100%的测试覆盖率

测试自动化的适用范围

- 不适合自动化测试情况
  - 定制型项目（一次性的）
  - 项目周期很短的项目
  - 业务规则复杂的对象
  - 美观、声音、易用性测试
  - 测试很少运行，如：一个月只运行一次
  - 软件不稳定
  - 涉及物理交互
- 适合自动化测试情况
  - 产品型项目
  - 增量式开发、持续集成项目
  - 能够自动编译、自动发布的系统
  - 回归测试
  - 多次重复、机械性动作
  - 需要频繁运行测试

自动化测试的前提

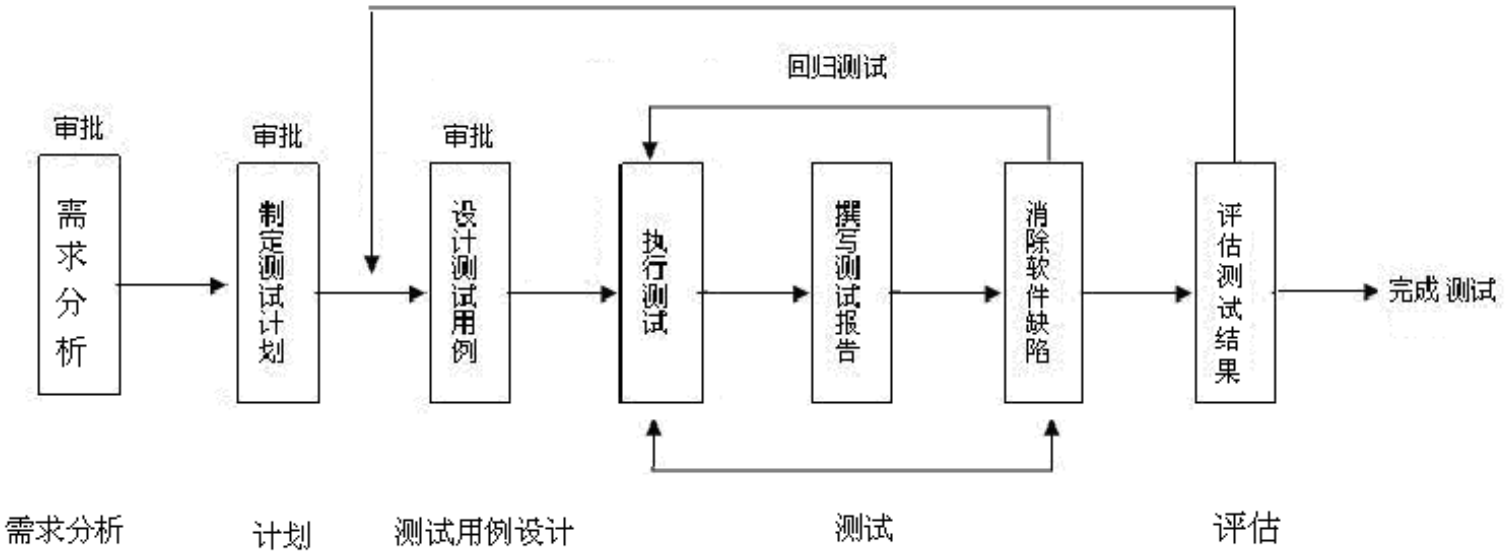
- 实施自动化测试之前需要对软件开发过程进行分析，以观察其是否适合使用自动化测试。通常需要同时满足以下条件：
  - *产品本身特征具有长期可维护性；*
  - *软件需求变动不频繁；*
    - 项目中的某些模块相对稳定，而某些模块需求变动性很大。我们便可对相对稳定的模块进行自动化测试，而变动较大的仍是用手工测试。
  - *项目周期足够长*
    - 由于自动化测试需求的确定、自动化测试框架的设计、测试脚本的编写与调试均需要较长的时间来完成。如果项目的周期比较短，没有足够的时间去支持这样一个过程，那么不便自动化测试。
  - *产品本身非紧迫的大项目；*
  - *产品结构相对复杂；*
  - *资源投入相对充裕；*
  - *在手工测试无法完成，需要投入大量时间与人力时也需要考虑引入自动化测试。比如性能测试、配置测试、大数据量输入测试等*

自动化测试的成本

- 实现成本
- 人力成本
- 新技术的风险
- 被自动化的功能是否需要大量的人工劳动
  - *有的人说：“从管理的角度来说，100% 的自动化目标只是一个从理论上可能达到的，但是实际上达到100% 的自动化的代价是十分昂贵的。一个40-60% 的利用自动化的程度已经是非常好的了。达到这个级别以上将增加测试相关的维护成本*

自动化测试的过程

- 利用自动化测试工具，经过对测试需求的分析，设计出自动化测试用例，从而搭建自动化测试的框架，设计与编写自动化脚本，测试脚本的正确性，同样需要经历需求分析、计划、测试用例设计、测试、评估

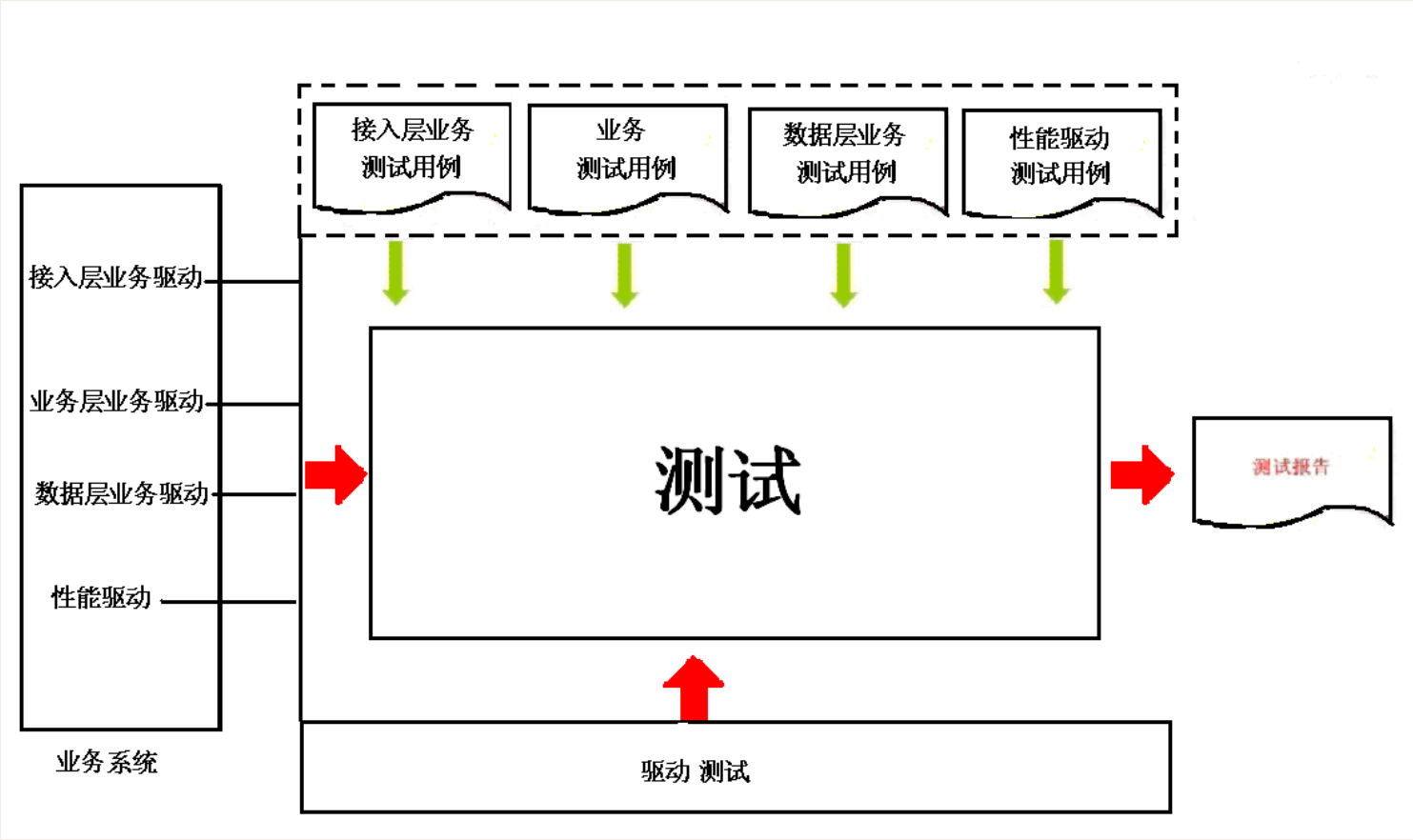


自动化测试工具选型的原则

- 目前还没有一个单一的测试工具能用来完成所有的测试需求。
- 测试工具品种不一，功能性能各异。对自动测试工具的适当选择，很大程度上决定了该工具能否获得相应的投资回报。
- 测试工具分为两类：
  - *找错工具（fault-finding）：根据既定的测试标准寻找被测程序中的缺陷，包括静态分析工具（一些白盒测试工具例如parasoft的jtest含有该功能）、动态测试工具（市面众多的测试工具robot、winrunner、qarun等）*
  - *测试支持工具：测试管理工具（如testmanager、testdirecter等）、其他支持工具（如clearquest、clearcase等）。*
- 自动化测试工具选型的参考性原则，testage.net建议：
  - *选择尽可能少的自动化产品覆盖尽可能多的平台，以降低产品投资和团队的学习成本；*
  - *测试流程管理自动化通常应该优先考虑，以满足为企业测试团队提供流程管理支持的需求；*
  - *在投资有限的情况下，性能测试自动化产品将优先于功能测试自动化被考虑；*
  - *在考虑产品性价比的同时，应充分关注产品的支持服务和售后服务的完善性；*
  - *尽量选择趋于主流的产品，以便通过行业间交流甚至网络等方式获得更为广泛的经验和支持；*
  - *应对测试自动化方案的可扩展性提出要求，以满足企业不断发展的技术和业务需求。*

自动化测试采用的技术

- 录制回放
- 脚本技术
- 数据驱动
- 关键字驱动
- 业务驱动
  - *接入层业务驱动*
  - *业务层业务驱动*
  - *数据层业务驱动*
  - *性能驱动*



自动化测试采用的技术

- 比较

	可维护性	可靠性	效率	易用性	可移植性	可复用性	健壮性
录制回放	低	低	低	高	低	低	低
脚本技术	中	高	中	低	低	中	高
数据驱动	中	高	中	中	中	中	高
关键字驱动	中	高	中	中	中	高	高
业务驱动	高	高	高	高	中	高	高

自动化测试-录制/回放技术

- 录制回放技术是以前比较流行的脚本生成技术。
- 录制回放技术可以自动录制测试执行者所做的所有操作，并将这些操作写成工具可以识别的脚本。工具通过读取脚本，并执行脚本中定义的指令，可以重复测试执行者手工完成的操作。
- 其优点在于：
  - *可以很快得到可再现的测试比较结果。*
  - *自动产生可以直接使用的测试脚本。*
  - *自动准备测试数据*
- 缺点，会随着使用的次数的增加越来越明显，主要表现在：
  - *脚本的维护性*
  - *效率问题*
  - *界面识别问题*

自动化测试-脚本技术

- 脚本技术是实现自动化测试最基本的一条要求，脚本语言具有与常用编程语言类似的语法结构，并且绝大多数为解释型语言，可以方便的在IDE中对脚本进行编辑修改。
- 任何一种脚本技术应该具备以下功能：
  - *支持多种常用的变量和数据类型。*
  - *支持数组、列表、结构，以及其它混合数据类型。*
  - *支持各种条件逻辑，（IF、CASE等语句）。*
  - *支持循环（FOR、WHILE）。*
  - *支持函数的创建和调用。*
  - *支持文件读写和数据源连接*

自动化测试-脚本技术

- 脚本技术的种类
  - *线性脚本*
  - *结构化脚本*
  - *共享脚本*
  - *数据驱动脚本*
  - *关键字驱动脚本*

自动化测试-脚本技术

- 线性脚本
  - *通过录制手工执行的测试用例时得到的脚本，这种脚本包含所有的击键（键盘和鼠标）、控制测试软件的控制键及输入数据的数字键，可以添加比较指令实现结果比较 。*
  - *如果用户只使用线性脚本技术，即录制每个测试用例的全部内容，则每个测试用例可以通过脚本完整地被回放。*
  - *几乎任何可重复的操作都可以使用线性脚本技术自动化*
  - *优点*
    - 不需要深入工作或者计划；
    - 可以快速开始自动化；
    - 对实际执行操作可以审计跟踪；
    - 用户不必是编程人员；
    - 提供良好的（软件或工具）的演示
  - *缺点*
    - 过程繁琐；
    - 一切依赖于每次捕获的内容；
    - 测试输入和比较是“捆绑”在脚本中的；
    - 无共享或重用脚本；
    - 容易受软件变化的影响；
    - 修改代价大，维护成本高；
    - 意外发生时脚本很容易与被测软件发生冲突，引起整个测试失败

自动化测试-脚本技术

- 结构化脚本
  - *类似于结构化程序设计，含有控制脚本执行的指令，支持顺序、选择和循环（叠代控制）3种基本控制结构，一个脚本可以调用另一个脚本。*
  - *由于引进其他指令改变控制结构，可以提高重用性，增加功能和灵活性，改善维护性。需要一定的编程技术*
  - *优点：*
    - 健壮性更好，可以对一些容易导致测试失败的特殊情况进行处理；
    - 可以批量执行许多类似的功能，例如需要重复的指令可以使用叠代结构；
    - 可以作为模块被其他脚本调用。
  - *缺点*
    - 脚本变得更加复杂，而且测试数据仍然“捆绑”在脚本中

自动化测试-脚本技术

- 共享脚本
  - *共享脚本是脚本可以被多个测试用例使用。*
  - *这种脚本技术的思想是将一些常见任务单独编制脚本，当要执行这些任务的时候，只需要在测试用例适当的地方调用这些脚本即可*
  - *优点：*
    - 以较少的开销实现类似的测试；
    - 维护开销低于线性脚本；
    - 减少了重复的脚本；
    - 可以在共享脚本中添加更智能的功能；
  - *缺点：*
    - 需要跟踪更多的脚本，文档、名字以及存储，如果管理得不好，很难找出适当的脚本；
    - 对于每个测试仍需要一个特定的测试脚本，因此维护脚本开销仍然比较高；
    - 共享脚本通常针对被测软件的某一部分

自动化测试-脚本技术

- 数据驱动脚本
  - *数据驱动脚本技术将测试输入存储到独立的（数据）文件中，而不是存储在脚本中。脚本中存放控制信息。执行测试时，从文件而不是直接从脚本中读取测试输入*
  - *优点:*
    - 可以快速增加类似的测试;
    - 测试者增加新测试不必掌握工具脚本语言的技术;
    - 对第二个及以后类似的测试无额外的维护开销。
  - *缺点:*
    - 初始建立的开销较大
    - 需要专业（编程）支持
    - 必须易于管理

自动化测试-脚本技术

- 关键字驱动脚本
  - *关键字驱动脚本技术实际上是较复杂的数据驱动技术的逻辑扩展。用关键字的形式将测试逻辑封装在数据文件中，测试工具只要能够解释这些关键字即可对其应用自动化。*
  - *关键字驱动脚本有如下特征：*
    - 测试脚本由控制脚本、测试文件、支持脚本组成；
    - 控制脚本不再受被测软件或特殊应用的约束；
    - 测试文件中使用关键字描述测试事例；
    - 控制脚本依次读取测试文件中的每个关键字并调用相关的支持脚本

自动化测试-自动比较

- 自动比较就是让测试工具自动完成实际输出和预期输出之间的比较任务。
- 自动比较的类型：
  - *简单比较：又称无智能比较，在实际输出与预期输出之间寻求完全相同的匹配*
  - *复杂比较：又称智能比较，允许用已知的差异来比较实际输出和预期输出*
  - *动态比较：在执行测试用例时进行的比较*
  - *执行后比较：在测试用例运行后执行的比较*

自动化测试-前处理和后处理

- 在大多数测试用例中，开始测试之前要具备一些适当的先决条件。这些先决条件应该在测试之前实现，称为自动化的前处理。
- 每次测试执行过后需要进行恢复工作，称为自动化的后处理
- 前处理和后处理的特征
  - *数量多*
  - *成批量出现*
  - *类型重复多*
  - *容易自动化*
- 前处理和后处理的自动化实现方式
  - *脚本*
    - 前处理和后处理任务可以在脚本程序里执行，所以它们可以直接由测试执行工具来实现。鉴于许多任务都很简单，因此可更有效地用共享脚本程序来执行它们。
  - *命令行文件*
    - 大多数前处理和后处理任务能用一些形式的命令文件来执行(像是命令程序、外壳脚本或批处理文件等)。

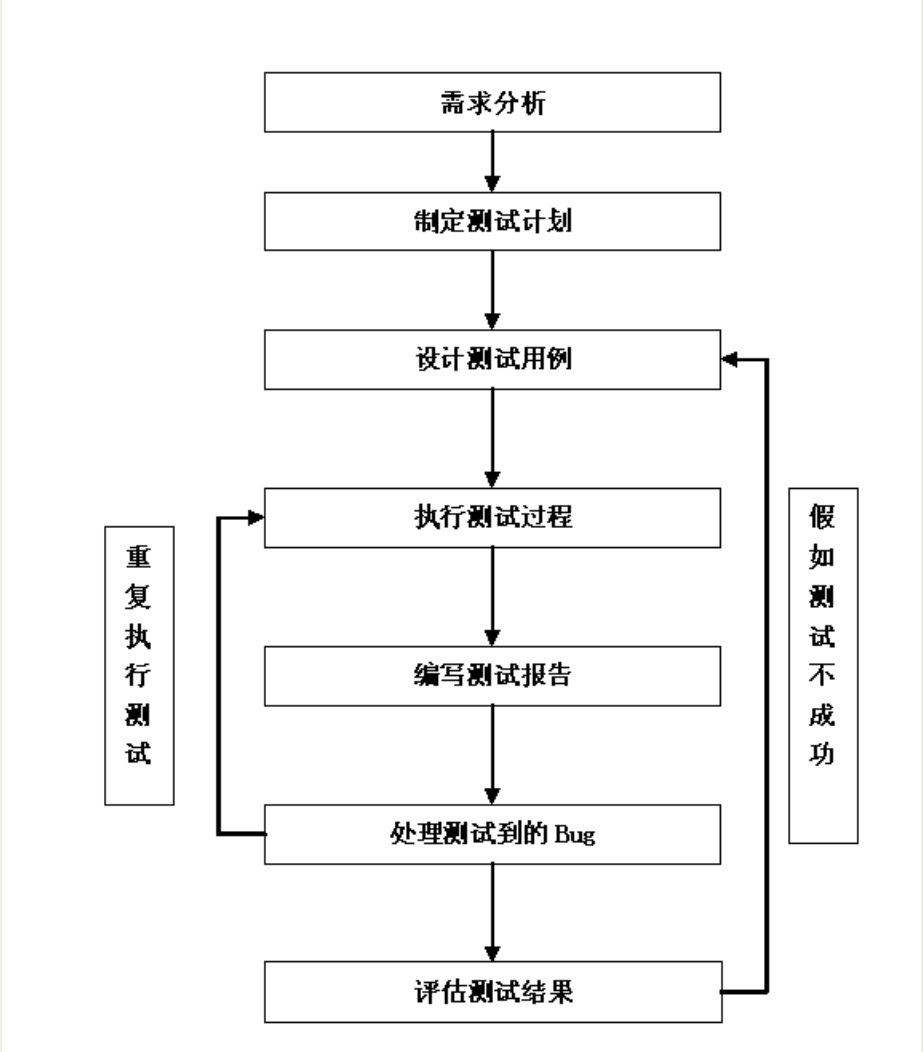
自动化测试方案选择需要考虑的因素

- 项目的影响：自动化测试能否帮助你的项目进度、覆盖率、风险?
- 复杂度：自动化是否容易实现，包括数据和其他环境的影响。
- 时间：自动化测试的实现需要多少时间?
- 早期需求和代码的稳定性：需求或早期的代码是否能证明是在范围内变化的?
- 维护工作量：代码是否能长期保持相对稳定？功能特性是否会进化?
- 覆盖率：自动化测试能否覆盖程序的关键特性和功能?
- 资源：测试组是否拥有足够的人力资源、硬件资源和数据资源来运行自动化测试。
- 自动化测试的执行：负责执行自动化测试的小组是否拥有足够的技能和时间去运行自动化测试?
- 自动化测试管理
  - *自动化测试管理测试过程的资源有：测试脚本、测试操作、库函数、场景恢复、测试数据、对象库等。*

自动化测试要点与重点

- 要点
  - 测试的范围
  - 测试时间的准备工作
  - 录制/回放
  - 对测试脚本的维护
- 重点
  - 搭建测试环境、测试场景
  - 测试用例
  - 测试结果验证
  - 自动化测试的基本流程

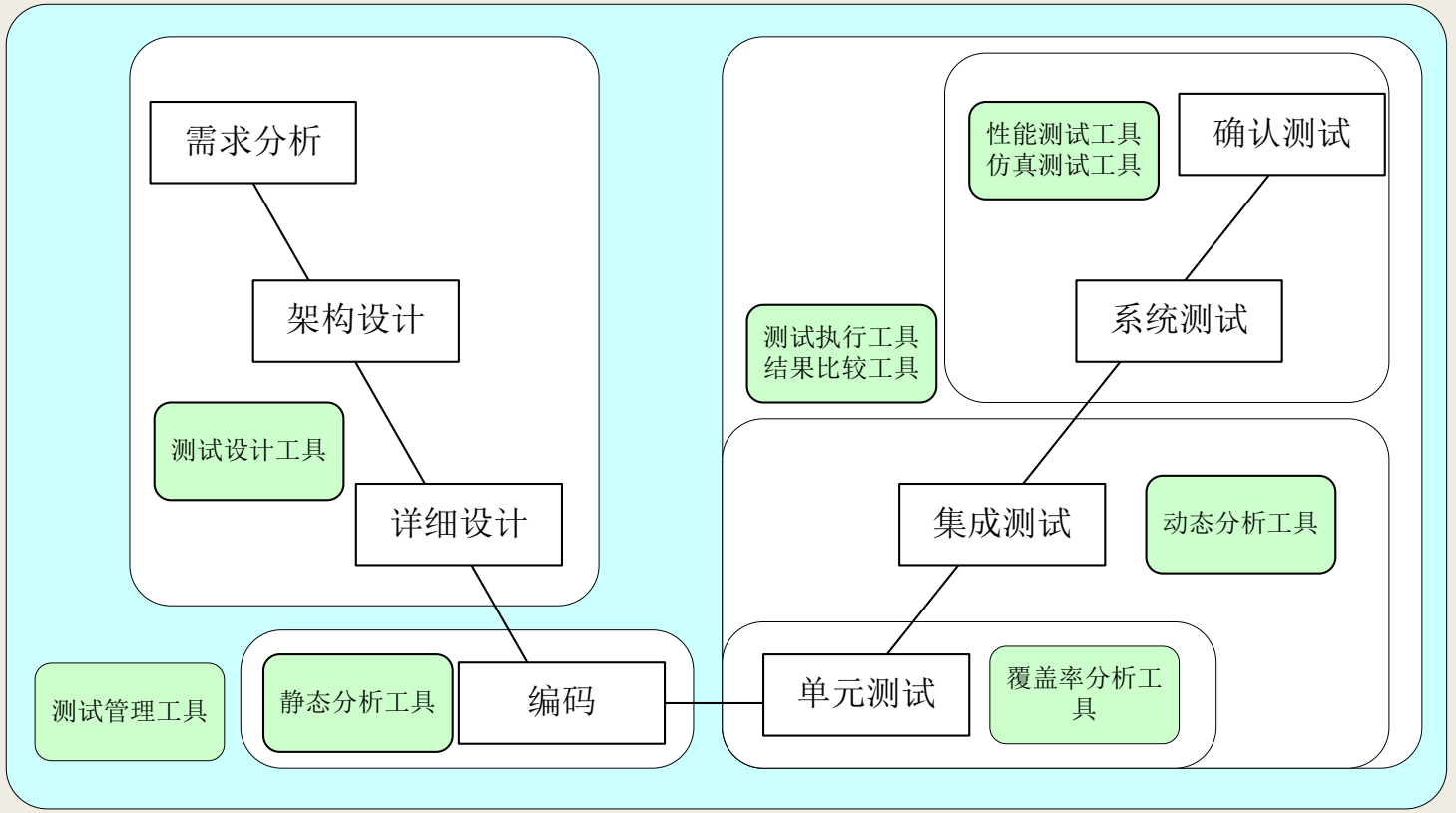
自动化测试的基本流程


需求分析
制定测试计划
设计测试用例
执行测试过程
编写测试报告
处理测试到的 Bug
评估测试结果
重复执行测试
假如测试不成功

自动化测试工具

- 测试工具可以从两个不同的方面去分类。
  - *根据测试方法不同，自动化测试工具可以分为：*
    - 白盒测试工具、黑盒测试工具
  - *根据测试的对象和目的，自动化测试工具可以分为：*
    - 单元测试工具、功能测试工具、负载测试工具、性能测试工具、Web测试工具、数据库测试工具、回归测试工具、嵌入式测试工具、页面链接测试工具、测试设计与开发工具、测试执行和评估工具、测试管理工具等

自动化测试工具与软件开发周期的关系



自动化测试工具-白盒测试工具

- 白盒测试工具一般是针对被测源程序进行的测试，测试所发现的故障可以定位到代码级。根据测试工具工作原理的不同，白盒测试的自动化工具可分为静态测试工具和动态测试工具。
- 静态测试工具——是在不执行程序的情况下，分析软件的特性。静态分析主要集中在需求文档、设计文档以及程序结构方面。
- 按照完成的职能不同，静态测试工具包括以下几种类型：
  - *代码审查*
  - *一致性检查*
  - *错误检查*
  - *接口分析*
  - *输入输出规格说明分析检查*
  - *数据流分析*
  - *类型分析*
  - *单元分析*
  - *复杂度分析*

自动化测试工具-白盒测试工具（续）

- 动态测试工具——是直接执行被测程序以提供测试活动。它需要实际运行被测系统，并设置断点，向代码生成的可执行文件中插入一些监测代码，掌握断点这一时刻程序运行数据（对象属性、变量的值等），具有功能确认、接口测试、覆盖率分析、性能分析等性能。动态测试工具可以分为以下几种类型：
  - *功能确认与接口测试*
  - *覆盖测试*
  - *性能测试*
  - *内存分析*
- 常用的动态工具有：
  - *Compuware公司的DevPartner*
  - *IBM公司的 Rational Purify*

自动化测试工具-黑盒测试工具

- 黑盒测试工具是在明确软件产品应具有的功能的条件下，完全不考虑被测程序的内部结构和内部特性，通过测试来检验软件功能是否按照软件需求规格的说明正常工作。
- 按照完成的职能不同，黑盒测试工具可以分为：
  - *功能测试工具——用于检测程序能否达到预期的功能要求并正常运行。*
  - *性能测试工具——用于确定软件和系统的性能。*
- 常用的黑盒测试工具有：
  - *Compuware公司的QACenter*
  - *IBM公司的Rational TeamTest*

自动化测试工具-测试设计与开发工具

- 测试设计是说明被测软件特征或特征组合的方法，并确定选择相关测试用例的过程。
- 测试开发是将测试设计转换成具体的测试用例的过程。
- 测试设计和开发需要的工具类型有：
  - *测试数据生成器*
  - *基于需求的测试设计工具*
  - *捕获/回放*
  - *覆盖分析*

自动化测试工具-测试执行和评估工具

- 测试执行和评估是执行测试用例并对测试结果进行评估的过程，包括选择用于执行的测试用例、设置测试环境、运行所选择的测试用例、记录测试执行过程、分析潜在的故障，并检查测试工作的有效性。
- 评估类工具对执行测试用例和评估测试结果过程起到辅助作用。
- 测试执行和评估类工具有：
  - *捕获/回放*
  - *覆盖分析*
  - *存储器测试*

自动化测试工具-测试管理工具

- 测试管理工具用于对测试过程进行管理，帮助完成制定测试计划，跟踪测试运行结果。通常，测试管理工具对测试计划、测试用例、测试实施进行管理，还包括缺陷跟踪管理等。
- 常用的测试管理工具有：
  - *IBM公司的Rational Test Manager*
- 测试管理工具包括以下内容：
  - *测试用例管理*
  - *缺陷跟踪管理（问题跟踪管理）*
  - *配置管理*

自动化测试-选择自动化测试工具

- 测试人员在选择和使用自动化测试工具时，可以从以下角度来考虑：
  - *按照用途选择匹配的测试工具*
  - *在适当的生命周期选择测试工具*
  - *按照测试人员的实际技能选择匹配的测试工具*
  - *选择一个可提供的测试工具*

自动化测试-基本测试工具

- 配置管理工具
- 缺陷跟踪工具
- 监控工具
- 功能测试工具
- 性能测试工具

自动化测试-基本测试工具

- 配置管理工具
  - *配置管理工具提供了全面的配置管理功能——包括版本控制、工作空间管理、Build管理和过程控制，而且无需软件开发者改变他们现有的环境、工具和工作方式。*
  - *主要功能：*
    - 版本控制
    - 工作空间管理
    - Build管理
    - 过程控制

自动化测试-基本测试工具

- 缺陷跟踪工具
  - *缺陷跟踪工具用于帮助公司和团队跟踪工作中的问题，管理和记录这些问题的处理过程，并为用户提供事务分配和自动通知的平台。*
  - *功能优点：*
    - 配合使用者的工作方式
    - 针对整个生命周期的缺陷跟踪
    - 设计一次就可以到处使用
    - 将分散的团队整合起来

自动化测试-基本测试工具

- 监控工具
  - *监控工具用来标明未测试代码并提供代码覆盖分析工具，是一个面向VC、VB或者Java 开发的测试覆盖程度检测工具，可以自动检测测试完整性和那些无法达到的部分。*
  - *具体功能：*
    - 即时代码测试百分比显示；
    - 未测试，测试不完整的函数，过程或者方法的状态表示；
    - 在源代码中定位未测试的特定代码行；
    - 为执行效率最大化定制数据采集；
    - 为所需要的焦点细节定制显示方式；
    - 从一个程序的多个执行合成数据覆盖度；
    - 和其他团队成员共享覆盖数据或者产生报表

自动化测试-基本测试工具

- 功能测试工具
  - *功能测试工具实现了功能测试和回归测试的自动化，它具有一个包含多种自定义选项的、健壮的用户动作记录器，并具备智能脚本维护能力，使得测试创建和执行过程在应用程序变更时是可恢复的，可以降低功能测试上的人力和物力的投入和风险。*
  - *具体功能：*
    - 为Java、Web、Microsoft Visual Studio. Net WinForm程序提供健壮的测试支持。
    - 可以定制生成Java或Visual Basic.Net语言的测试脚本。
    - 使用Script Assure技术支持频繁的用户界面变更。
    - 自动化的数据关联和数据驱动测试，可以消除手工编码。
    - 多点验证，支持正则表达式的模式匹配。
    - 先进的对象映射维护能力。
    - 支持Linux测试的编辑和执行。

自动化测试-基本测试工具

- 性能测试工具
  - *性能测试工具用来提高应用程序的性能和质量，它为那些需要进行创建和配置可靠的应用程序的开发者设计。可以创建、修改和实现自动化的衰减、冒烟测试。*
  - *具体功能：*
    - 对当前的开发环境的影响达到了最小化；
    - 提供了树型关系调用图，及时反映了影响性能的关键数据；
    - 功能列表详细窗口，显示了大量与性能有关的数据；
    - 精确记录了源程序执行的指令数，正确反映了时间数据，在调用函数中正确传递这些记录，使关键路径一目了然；
    - 可以控制所收集到的数据，通过过滤器显示重要的程序执行过程

自动化测试-常用工具类型总结

	测试管理	功能测试	压力测试	白盒测试
Compuware	**QADirector**	**QARun TestPartner**	**QALoad**	**DevPartner**
MI	**TestDirector**	**WinRunner**	**LoadRunner**	**--**
Rational	**TestManager**	**Robot**	**LoadTest**	**Purify Purecoverage Quantify**
其他	--	--	**WAS**	**JUnit CppUnit**

自动化测试-一些开源测试工具

- 单元测试工具：
  - *JUNIT（CppUnit）：JUnit是一个开源的java测试框架，它是Xuint测试体系架构的一种实现。在JUnit单元测试框架的设计时，设定了三个总体目标，第一个是简化测试的编写，这种简化包括测试框架的学习和实际测试单元的编写；第二个是使测试单元保持持久性；第三个则是可以利用既有的测试来编写相关的测试。使用环境: Windows, OS Independent, Linux*
- 功能测试工具：
  - *Linux Test Project（http://ltp.sourceforge.net/ ）：Linux Test Project是一个测试Linux内核和内核相关特性的工具集合。该工具的目的是通过把测试自动化引入到Linux内核测试，提高Linux的内核质量。 使用环境: Linux*
  - *WebInject（http://www.webinject.org/）：WebInject是一个针对Web应用程序和服务的免费测试工具。 它可以通过HTTP接口测试任意一个单独的系统组件。可以作为测试框架管理功能自动化测试和回归自动化测试的测试套。使用环境: Windows, OS Independent, Linux*
  - *MaxQ（http://maxq.tigris.org/）：MaxQ是一个免费的功能测试工具。它包括一个HTTP代理工具，可以录制测试脚本，并提供回放测试过程的命令行工具。测试结果的统计图表类似于商用测试工具，比如Astra QuickTest和Empirix e-Test，这些商用工具都很昂贵。MaxQ希望能够提供一些关键的功能，比如HTTP测试录制回放功能，并支持脚本。使用环境: Java 1.2以上版本*

自动化测试-一些开源测试工具（续）

- 性能测试工具：
  - *Apache JMeter （http://jakarta.apache.org/jmeter/ ）：Apache JMeter是100%的Java桌面应用程序，它被设计用来加载被测试软件功能特性、度量被测试软件的性能。设计Jmeter的初衷是测试Web应用，后来又扩充了其它的功能。Jmeter可以完成针对静态资源和动态资源（Servlets, Perl脚本, Java对象, 数据查询s, FTP服务等）的性能测试。Jmeter可以模拟大量的服务器负载、网络负载、软件对象负载，通过不同的加载类型全面测试软件的性能。Jmeter提供图形化的性能分析。使用环境: Solaris, Linux, Windows (98, NT, 2000). JDK1.4以上*
  - *OpenSTA (Open System Testing Architecture)（http://portal.opensta.org/index.php）：基于CORBA的分布式软件测试构架。使用OpenSTA，测试人员可以模拟大量的虚拟用户。OpenSTA的结果分析包括虚拟用户响应时间、web服务器的资源使用情况、数据库服务器的使用情况，可以精确的度量负载测试的结果。使用环境: OS Independent*

自动化测试-一些开源测试工具（续II）

- 性能测试工具：
  - *TPTEST（http://tptest.sourceforge.net/about.php）：工具描述: TPTest的提供测试Internet连接速度的简单方法。使用环境: MacOS/Carbon、 Win32*
  - *DBMonster （http://dbmonster.kernelpanic.pl/ ）：DBMonster是一个生成随机数据，用来测试SQL数据库的压力测试工具。使用环境: OS Independent*
  - *Web Application Load Simulator（http://www.openware.org/loadsim/index.html）：LoadSim是一个网络应用程序的负载模拟器。使用环境: JDK 1.3以上*

自动化测试-一些开源测试工具（续III）

- 缺陷管理工具：
  - *Mantis（http://mantisbt.sourceforge.net/）：Mantis是一款基于WEB的软件缺陷管理工具，配置和使用都很简单，适合中小型软件开发团队。使用环境: MySQL, PHP*
  - *Bugzilla（http://www.mozilla.org/projects/bugzilla/）：一款软件缺陷管理工具。使用环境: TBC*
- 测试管理工具：
  - *TestLink（http://testlink.sourceforge.net/docs/testLink.php）：基于WEB的测试管理和执行系统。测试小组在系统中可以创建、管理、执行、跟踪测试用例，并且提供在测试计划中安排测试用例的方法。使用环境: Apache, MySQL, PHP*
  - *Bugzilla Test Runner（http://sourceforge.net/projects/testrunner/）：Bugzilla Test Runner基于Bugzilla缺陷管理系统的测试用例管理系统。使用环境: Bugzilla 2.16.3 or above（bugzilla是一个可以发布bug以及跟踪报告bug进展情况的开源软件）*

自动化测试成熟度

- 自动化测试工具以前只是被看作是一种捕获和回放的工具
- 按照成熟度，自动化测试是一种捕获、编辑、编程、数据驱动、使用动作词和回放的工具，可划分为 5 个级别。
  - *级别 1 捕获和回放*
  - *级别 2 捕获、编辑和回放*
  - *级别 3 编程和回放*
  - *级别 4 数据驱动的测试*
  - *级别 5 使用动作词的测试自动化*

自动化测试成熟度-级别1

- 捕获和回放：这是使用自动化测试的最低的级别，同时这并不是自动化测试最有用的使用方式
- 优点
  - *自动化的测试脚本能够被自动的生成，而不需要有任何的编程知识*
- 缺点
  - *你会拥有大量的测试脚本，同时当需求和应用发生变化时相应的测试脚本也必须被重新录制*
- 用法
  - *当测试的系统不会发生变化时 — 小规模的自动化*

自动化测试成熟度-级别2

- 捕获和回放：捕获、编辑和回放：在这个级别中，使用自动化的测试工具来捕获想要测试的功能。将测试脚本中的任何写死的测试数据，比如名字、帐号等等，从测试脚本的代码中完全删除，并将他们转换成为变量
- 优点
  - *测试脚本开始变得更加的完善和灵活，并且可以大大的减少脚本的数量和维护的工作*
- 缺点
  - *需要一定的编知识。频繁的变化可能会引起"意大利面条式的代码"，并且变更和维护几乎是不可能的*
- 用法
  - *当进行回归测试时，被测试的应用有很小的变化，比如仅仅是针对计算的代码变化，但是没有关于GUI 界面的变化*

自动化测试成熟度-级别3

- 编程和回放：这个级别是面对多个构建版本的有效使用测试自动化的第一个级别
- 优点
  - *确定了测试脚本的设计，使用与开发中相同的编码习惯。搭建起测试和开发之间的桥梁。*
  - *在项目的早期就可以开始自动化的测试。能够在项目的早期就开始进行测试脚本的设计。与开发人员交并调查他们认为可能会存在问题的区域。确保了开发人员关注在获得能够被测试的方案上*
- 缺点
  - *要求测试人员具有很好的软件技能，包括设计、开发等*
- 用法
  - *大规模的测试套件被开发、执行和维护的专业自动化测试*

自动化测试成熟度-级别4

- 数据驱动的测试：对于自动化测试来说这是一个专业的测试级别。拥有一个强大的测试框架，这个测试框架是基于能够根据被测试系统的变化快速创建一个测试脚本的测试功能库。维护的成本相对是比较低的。在测试中会使用到大量真实的数据
- 优点
  - *能够维护和使用良好的并且有效的模拟真实生活中数据的测试数据*
- 缺点
  - *软件开发的技能是基础，并且需要访问相关的测试数据*
- 用法
  - *大规模的测试套件被开发、执行和维护的专业自动化测试*

自动化测试成熟度-级别5

- 数据驱动的测试：对于自动化测试来说这是一个专业的测试级别。拥有一个强大的测试框架，这个测试框架是基于能够根据被测试系统的变化快速创建一个测试脚本的测试功能库。维护的成本相对是比较低的。在测试中会使用到大量真实的数据
- 优点
  - *能够维护和使用良好的并且有效的模拟真实生活中数据的测试数据*
- 缺点
  - *软件开发的技能是基础，并且需要访问相关的测试数据*
- *用法*
  - *大规模的测试套件被开发、执行和维护的专业自动化测试*

自动化测试与单元测试

- 单元自动化测试是测试最低层次——保证每个函数/方法，或者说最小功能模块的正确性的一种测试。我们要清楚了这样的2件事情：
  - *单元测试是最低层级的测试，它只保证函数（子程序）的可靠性，不保证其它；*
  - *单元测试应该能保证每一个函数（子程序）的可靠性。*
- 理论上来说，单元测试和其他测试一样，也是可以纯人工完成的（我们可以写一段某函数的测试代码，然后输入我们的测试输入，观察测试输出，并跟期望值做比较——事实上这种人工测试）。但是，单元测试有一点特殊性，就是在一个系统中，函数（子程序模块）会非常非常的多，变化也比软件的功能频繁的多。面对这么多的函数（子程序模块），这么频繁的变化，进行自动化测试

自动化单元测试的重点

- 如何搭建测试环境、测试场景?
- 如何选择测试用例?
- 任何测试用例的数据准备都是一个令人头疼的问题，进行白盒测试时一般都需要根据路径、边界值、编码流程来进行测试用例数据的准备。假如，您所选择的测试工具能够自动生成测试用例推荐使用工具自动生成，然后再其自动生成用例的基础上进行编辑强化测试脚本。
- 如何校验测试结果?
- 对于测试代码本身，应该尽可能的简单，如果测试本身过于复杂，我们不能保证测试的正确性，测试这个工作就白做了。
- 单元测试的编写，是由开发人员做的。开发人员最清楚函数（子程序模块）需求，在这种情况下当然就应该是开发人员自己编写测试用例。
- 在编写测试用例的时候绝不能假定任何函数（子程序模块）的实现，而应该完全按照它应该有的需求来做。
- 如何准备自动化测试脚本，自动化测试用例通常称为“测试脚本”。
- 单元测试中单元模块的选择

自动化单元测试的优点

- 自动化：节省了我们的时间，免去了重复的操作。
- 测试数据重用性：多用于回归测试，也可用于相同功能点的测试数据的准备工作中。
- 测试的可控性：我们完全可以按测试计划来控制单元测试的进度，我们可以选择在合适的时间，使用合适的方法及合适的数据来进行单元测试

单元测试自动化工具

- 选择单元测试自动化工具我们需要了解我们的测试环境、待测系统类型、开发语言等等
  - *基于java语言开发的软件（包括C/S 及B/S架构的软件），我们可以选择免费开源的Junit 测试工具来进行测试。Junit用于编写和运行可重复的测试。*
  - *基于C++ 语言开发的软件我们可以选择适用C++ Test 进行测试。C++ Tes 是Parasoft针对C/C++的一款自动化测试工具。C++test支持编码策略增强，静态分析，全面代码走查，单元与组件的 测试，为用户提供一个实用的方法来确保其C/C++代码按预期运行。*
  - *基于微软C# .net框架开发的软件系统我们可以使用.net unit 测试工具来测试.netunitye 是基于Nunit系列的单元测试软件之一，如果您有兴趣了解，请自行查阅资料*

自动化测试与功能测试

- 功能测试在以下几种情况下引入自动化测试
  - *测试时间相对长，且存在大量重复性、机械性人工测试的项目；*
  - *产品型软件，每发布一个新的版本或打补丁都需要对其他模块执行相同的测试；*
  - *项目型软件，需求变更频繁，每变更一次，需要对原有的无争议的功能做测试；*
  - *经常需要更换应用程序部署站点的软件，每更换一次需要对所有功能做验证测试；*
  - *测试时间相对长，且存在大量需要执行回归测试的软件项目；*
  - *系统界面稳定，需要对业务流程进行验证测试的软件；*
  - *采用增量开发持续集成的项目，需要对频繁更新的程序执行验证测试；*
  - *软件项目采用主流开发平台技术，且不存在物理交互的测试，如刷卡测试；*
  - *项目工期紧、测试周期短的项目不应采取自动化测试；*
  - *界面的美观、声音的体验和易用性的测试不应采取自动化测试。*

功能自动化测试需要解决的问题

- 开始之前：在功能自动化测试项目开始之前，应该全面地调查和了解
  - *测试过程自动化的成本是多少?*
  - *其投资回报率是什么?*
  - *哪些应用/过程适合做自动化测试，哪些不合适?*
  - *是否需要新的培训，这将对当前的开发计划安排产生怎样的影响?*
  - *自动化测试得正确地方法论是什么?*
  - *自动化测试时涉及到哪些情况?*
  - *当比较自动化测试产品时，哪些功能最重要?*
- 开始功能自动化测试需要解决的问题
  - *准备数据*
  - *复杂操作*
  - *测试太脆弱*
  - *测试比较麻烦*
  - *执行速度比较慢*
  - *带验证码的页面没法测*