软件自动化测试基础

软件自动化测试基础

- 自动化测试基础
- 软件自动化测试工具简述
- 自动化测试解决方案举例

自动化测试基础

- 自动化测试的定义、意义
- 正确认识自动化测试
- 自动化测试的原理和方法

自动化测试

- 什么是自动化测试？

  一般是指软件测试的自动化。自动化测试可理解为测试过程自动化和测试结果分析自动化，包括测试活动的管理与实施、测试脚本的开发与执行。

  软件自动化测试就是模拟手动测试步骤，控制被测软件的执行，完成全自动或半自动测试的过程。

  全自动测试：指不需人工干预，由程序自动完成测试的全过程。

  半自动测试：指在测试过程中，需手动输入测试用例或选择测试路径，再由自动测试程序按照人工指定的要求完成自动测试。

自动化测试的意义

- 软件测试工作量大，重复性高

  通常，软件测试的工作量很大（据统计，测试会占用到40%的开发时间；一些可靠性要求非常高的软件，测试时间甚至占到开发时间的60%）。而测试中的许多操作是重复性的、非智力性的和非创造性的，并要求做准确细致的工作，计算机就最适合于代替人工去完成这样的任务。

自动化测试的意义

- 自动化测试带来的好处
  - 缩短软件开发测试周期，可让产品更快投放市场。
  - 提高测试效率，充分利用硬件资源。
  - 节省人力资源，降低测试成本。
  - 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。
  - 测试工具使测试工作相对较容易，且能产生更高质量的测试结果。
  - 手工不能做的事情，自动化测试能做，如负载、性能测试。

自动化测试的引入

- 正确的认识观

  软件测试实行自动化进程，绝不是因为厌烦了重复的测试工作，而是因为测试工作的需要，更准确地说是回归测试和系统测试的需要。

  软件测试工具能提高测试效率、覆盖率和可靠性等，自动化测试虽然具有很多优点，但它只是测试工作的一部分，是对手工测试的一种补充。

自动化测试存在的一些问题

- 不正确的观念或不现实的期望
- 缺乏具有良好素质和有经验的测试人才
- 没有进行有效、充分的培训
- 测试工具本身的问题影响测试的质量
- 自动测试的维护开销问题，不考虑公司的实际情况、盲目引入测试工具
- 其他技术问题和组织问题

测试自动化限制

测试自动化可以带来非常明显的收益，但也有以下限制：

- 不能取代手工测试
- 手工测试比自动测试发现的缺陷更多
- 对测试质量的依赖性极大
- 测试自动化不能提高有效性
- 测试自动化可能会制约软件开发。
- 工具本身并无想象力

另外，人工测试比测试工具更优越的另一个方面是可以处理意外事件。虽然工具也能处理部分异常事件，但是对真正的突发事件和不能由软件解决的问题就无能为力。

建立正确的自动化测试目标

克服不正确的观念，需要建立正确的自动化测试目标。

- 测试计划产生：主要靠测试工程师与软件生产质量保证专家的合作，自动化工具只能起到一定的辅助作用。
- 一种测试工具不完全适用于所有测试
- 自动测试不一定会减轻工作量
- 测试进度可能不一定会缩短
- 测试工具不一定易于使用
- 自动化测试的普遍应用存在局限
- 测试覆盖率不会达到百分之百

自动化测试的引入

- 前提条件

通常需要同时满足以下条件：

1) 软件需求变动不频繁

从维护成本考虑：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。

2) 项目周期足够长

自动化测试本身就是一个测试软件的开发过程，需较长时间完成。如项目周期较短，就没足够的时间去支持这个过程。

3) 自动化测试脚本可重复使用

成本应不大于所创造的经济价值

自动化测试的引入

- 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。

  多数情况下，手工测试和自动化测试应该相结合，以最有效的方法来完成测试任务。

自动化测试理论

- 软件测试自动化实现的基础

  是通过设计的特殊程序模拟测试人员对计算机的操作过程、操作行为；

  或者类似于编译系统那样对计算机程序进行检查。

- 自动化测试过程中主要涉及以下内容：
  - 对代码进行静态和动态分析。
  - 测试驱动、桩和驱动数据的自动生成：主要是依据所采用的测试方法，如等价类、边界值等自动产生多组测试数据。
  - 自动测试输入：工具录制测试者所做的所有操作，并将这些操作写成工具可以识别的脚本。被录制的脚本中含有测试输入（包括文本和鼠标移动、点击菜单和按钮等动作）

（continue）

自动化测试理论

自动化测试过程中主要涉及以下内容：（续）

- 测试脚本技术：用于自动测试过程存放测试步骤、测试数据等相关内容。
- 测试结果的自动比较：将预期输出与程序运行过程中的实际输出进行比较。
- 自动测试执行：工具读取脚本并执行脚本命令，可以重复测试者的操作。在执行脚本过程中可以完成测试结果的自动比较。
- 自动测试管理：完成测试计划、测试大纲、测试缺陷管理等工作。

下面对代码分析，测试脚本技术，测试结果的自动比较等内容做更多介绍：

自动化测试的原理和方法

- 1）代码分析

  代码分析类似于高级语言编译系统，一般针对不同的高级语言去构造分析工具，在工具中定义类、对象、函数、变量等定义规则、语法规则；

  在分析时对代码进行语法扫描，找出不符合编码规范的地方；

  根据某种质量模型评价代码质量，生成系统的调用关系图等。

自动化测试的原理和方法

2）捕获和回放

代码分析是一种白盒测试的自动化方法，捕获和回放则是一种黑盒测试的自动化方法。

捕获是将用户每一步操作都记录下来。这种记录的方式有两种：程序用户界面的像素坐标或程序显示对象（窗口、按钮、滚动条等）的位置，以及相对应的操作、状态变化或是属性变化。所有的记录转换为一种脚本语言所描述的过程，以模拟用户的操作。

回放时，将脚本语言所描述的过程转换为屏幕上的操作，然后将被测系统的输出记录下来同预先给定的标准结果比较。

捕获和回放可以大大减轻黑盒测试的工作量，在迭代开发的过程中，能够很好地进行回归测试。

自动化测试的原理和方法

- 关于自动化测试中的“录制—回放”技术

  所谓的“录制-回放”技术，就是先由手工完成一遍需要测试的流程，同时由计算机记录下这个流程期间客户端和服务器端之间的通信信息，这些信息通常是一些协议和数据，并形成特定的脚本程序 (Script) 。

  然后在系统的统一管理下同时生成多个虚拟用户，并运行该脚本，监控硬件和软件平台的性能，提供分析报告或相关资料。这样，通过几台机器就可以模拟出成百上千的用户对应用系统进行负载能力的测试。

自动化测试的原理和方法

3）脚本技术

脚本是一组测试工具执行的指令集，也是计算机程序的一种形式。

脚本可通过录制测试的操作产生，然后再做修改，这样可减少脚本编程的工作量。当然，也可以直接用脚本语言编写脚本。脚本中包含的是测试数据和指令，一般包括如下信息：

- 同步（何时进行下一个输入）。
- 比较信息（比较什么，比较标准）。
- 埔获何种屏幕数据及存储在何处。
- 从哪个数据源或从何处读取数据。
- 控制信息等。

自动化测试的原理和方法

脚本技术可以分为以下几类：

- 线性脚本

  是录制手工执行的测试用例得到的脚本。

  优点：

  1）不需要深入的工作或计划，只需坐在计算机前录制手工任务；2）可以快速开始自动化； 3）对实际执行操作可以审计跟踪； 4）用户不必是编程人员； 5）提供良好的（软件或工具）演示。

  缺点：

  1）一切依赖于每次捕获的内容； 2）测试输入和比较是“捆绑”在脚本中的； 3）无法共享或重用脚本； 4）容易受软件变化的影响；5）修改代价大，维护成本高。

自动化测试的原理和方法

脚本技术可以分为以下几类：

- 结构化脚本

类似于结构化程序设计，具有各种逻辑结构（顺序、分支、循环），而且具有函数调用功能。

优点：由千引进其他指令改变控制结构，可以提高健壮性、重用性，增加功能和灵活性，改善维护性。

缺点：测试数据依然“包含“在脚本中。

自动化测试的原理和方法

脚本技术可以分为以下几类：

- 共享脚本

是指某个脚本可被多个测试用例使用，即脚本语言允许一个脚本调用另一个脚本。

优点：

1）以较少的开销实现类似的测试；2）维护开销低于线性脚本；3）删除明显的重复； 4）可以在共享脚本中增加更智能的功能。

缺点：

1）需要跟踪更多的脚本，文档、文字以及存储，如果管理得不好，很难找到适当的脚本； 2）每个测试仍需要一个特定的测试脚本，维护成本仍然比较高； 3）共享脚本通常只是针对被测软件的某一部分。

自动化测试的原理和方法

脚本技术可以分为以下几类：

- 数据驱动脚本

将测试输入存储在独立的数据文件中，而不是存储在脚本中，脚本中只存放控制信息。

用变量取代在录制的脚本代码中固定输入内容，如：名字、地址、数据等，然后通过变量从外部数据文件读取数据的测试。

优点：

1）可以很快增加类似的测试（脚本相同，数据不同）；

2）测试者增加新测试不必具有工具脚本语言的技术或编程知识；

3）对千第二个测试及后续测试无额外的脚本维护开销。

缺点：

1）初始建立的开销较大；　2）需要专业（编程）支持。

自动化测试的原理和方法

脚本技术可以分为以下几类：

- 关键字驱动脚本

是数据驱动脚本的逻辑扩展，将数据文件变为测试用例描述，用一系列关键字指定要执行的任务。

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用例

测试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

优点：1）独立于测试脚本语言开发测试事例；2）所需脚本数量是随软件的规模而不是测试的数量而变化的；3）可以用与工具（及平台）无关的方法实现测试；4）实现测试的方法可以剪裁适合测试者而不是测试工具

自动化测试的原理和方法

例:以下语句指示 QTP 选中 Itinerary 网页上的所有复选框:

```
Set MyDescrīption = Descrīption.Create()
MyDescrīption("html tag").Value = "INPUT"
MyDescrīption("type").Value = "checkbox"
Set Checkboxes =
Browser("Itinerary").Page("Itinerary").ChildObjects(MyDescrīp
    tion)
NoOfChildObjs = Checkboxes.Count
For Counter=0 to NoOfChildObjs-1
    Checkboxes(Counter).Set "ON"
Next
```

自动化测试的原理和方法

4）自动比较

- 静态比较和动态比较

  静态比较，在测试过程中不比较，而是将结果存入文件或数据库，最后比较结果；

  动态测试，在测试过程中比较。

- 简单比较和复杂比较

  简单比较，实际结果和预期结果完全相同；

  复杂比较，允许有一定误差。

自动化测试的原理和方法

4）自动比较

- 敏感性测试比较和健壮性测试比较

  敏感性测试比较，比较尽可能多的信息。如测试用例的每一步都比较；

  健壮性测试比较，只比较最需要的信息。如最后结果。

- 比较过滤器

  对预期输出进行预处理，执行过滤任务后，再比较。

5）测试管理

软件自动化测试工具简述

- 自动化测试工具的作用及优势
- 自动化测试工具的特征
- 自动化测试工具的分类
- 选择自动化测试工具
- 使用测试工具和自动化的实质
- 常用测试工具概要

自动化测试工具的作用及优势

- 软件测试自动化通常借助测试工具进行。

  测试工具可以进行部分的测试设计、实现、执行和比较的工作。

  部分的测试工具可以实现测试用例的自动生成，但通常的工作方式为人工设计测试用例，使用工具进行用例的执行和比较。

- 自动化测试工具的作用：

  （1）确定系统最优的硬件配置。虚拟硬件进行配置测试。

  （2）检查系统的可靠性。大负载，长时间。

  （3）检查系统硬件和软件的升级情况。软硬件对系统性能的影响。

  （4）评估新产品。

自动化测试工具的作用及优势

- 自动化测试工具的优势主要体现在以下几个方面：

  （1）记录业务流程并生成脚本程序的能力。

  （2）对各种网络设备（客户机或服务器、其它网络设备）的模仿能力。

  （3）用有限的资源生成高质量虚拟用户的能力。

  （4）对于整个软件和硬件系统中各个部分的监控能力。

  （5）对于测试结果的表现和分析能力。

自动化测试工具的特征

- 支持脚本化语言(Scripting Language)
- 对程序界面中对象的识别能力
- 支持函数的可重用
- 支持外部函数库
- 抽象层—将程序界面中的对象实体映射成逻辑对象
- 分布式测试(Distributed Test)的支持
- 支持数据驱动测试(Data-Driven Test)
- 错误处理
- 调试器(Debugger)
- 源代码管理
- 支持脚本的命令行(Command Line)方式

自动化测试工具的分类

测试工具可以从多个不同的方面去分类。

- 从入侵角度，分入侵式工具和非入侵式工具：

  非入侵式工具：如果工具仅用于监视和检查软件而不对其进行修改，就认为是非入侵式工具。

  入侵式工具：如果工具以任何方式修改了程序代码或者控制了操作环境，就属于入侵式工具。

  由于入侵的程度各有不同，测试员通常设法使用侵入性尽量小的工具，以减少工具影响测试结果的可能性。

- 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

  (Continue)

自动化测试工具的分类

- 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

  - 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

自动化测试工具的分类

- 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:选择测试用例、设置测试环境、运行测试用例、记录测试执行过程、分析潜在的故障，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

- 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 等

自动化测试工具的分类

- 根据使用方式，自动化测试工具可以分为：

查看器和监视器、驱动程序、桩、压力和负载工具、干扰注入器和噪声发生器和分析工具。

- 查看器（viewer）和监视器（monitor）

查看器或监视器：测试工具能看到正常情况下看不到的运行的细节。如：

1）代码覆盖率分析器(查看器的一种)

提供一种方式来查看哪些代码行得以运行、什么函数正在运行、执行测试时所运行的代码分支。

大多数的代码覆盖率分析器是入侵式工具，因为它们需要编译并链接到原程序中才能获得所需信息。

自动化测试工具的分类

2）通信分析器（communications analyzer）（一种查看器）

监听线路，提取经过的数据，在另一台计算机上显示。

利用该系统可查看通信数据的正确性以及观察软件缺陷为什么会产生，从哪产生等。

这类系统对软件是非入侵式的。

在网络中，真正监视器被称为嗅探器（sniffer）。

3）大多数编译器所带的代码调试器也可以看做是查看器。

看到一般用户看不到的数据的工具都可以归类为查看测试工具。

自动化测试工具的分类

- 驱动程序

  驱动程序是控制和操作被测试软件的工具。

- 桩

  桩接收或者响应被测软件发送的数据。

  一般在开发过程中不能得到某些设备，或这些设备很少，桩就可以使测试在没有硬件的条件下进行，使测试更加有效。

  仿真器（emulator）：仿真器是在实际使用中用来代替真正设备的设备。

  仿真器和桩的区别在于：桩还给测试程序提供手段来查看和解释发送给它的数据，桩是仿真器的超集。

自动化测试工具的分类

- 压力和负载工具

  压力（stress）和负载（load）工具用于向被测试软件增加压力和负载。

  一般的压力测试软件可以分别设置内存量、磁盘空间大小、文件数量，以及在该机器上运行软件的其它可用资源，以查看软件运行情况。

  负载工具和压力工具的相似之处在于，它们为软件创造了用其它方式难以创造的环境条件。

  如，运行在web服务器上的商用程序可以通过模拟一定数量的链接和单击次数来增大负载，使其不堪重负。

自动化测试工具的分类

- 干扰注入器和噪声发生器

  干扰注入器（interference injectors）和噪声发生器（noise generators）是类似于压力和负载工具的另一类工具。它们在行为上更具有随机性。

  如，挂在通信线路上的干扰注入器可以测试软件能否处理由噪声引起的错误情况。

  决定在哪里和如何使用干扰注入器和噪声发生器时，考虑何种外部因素会影响测试软件，然后设法改变和操纵这些影响因素看软件如何应付。

- 分析工具

  用于分析测试。软件的复杂性和方向性总是在变，要视具体情况来决定最有效的工具是什么，以及如何运用它们。

自动化测试工具的分类

- 按测试工具的收费方式，又可分为以下几类。

  商业测试工具。

  开源测试工具。

  免费测试工具。

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

使用测试工具和自动化的实质

如正确规划和执行，自动化测试可提高测试效率并能发现其它方式不能发现的缺陷。

如自动化测试步入歧途，会导致无数的自动化测试努力被放弃，项目成本大大增加。考虑以下因素：

1）软件变更；

2）人眼和自觉是不可替代的；

3）验证难以实现；

4）容易过分依赖自动化；

5）不要花费太多时间使用达不到测试软件目的的测试工具和自动化；

6）编写宏、开发工具和编制猴子都属于开发工作；

7）某些工具是入侵式的。

常用测试工具

- 目前，软件测试方面的工具很多

  商业测试软件主要有HP （以前的MercuryInteractive（MI）），IBM Rational等。

  免费/开源的，层出不穷。

Mercury公司测试工具

- HP Mercury（美科利）质量中心：提供一个全面的、基于Web的集成系统，可在广泛的应用环境下自动执行软件质量管理和测试。其主要产品如下：
  - **Winrunner**：是一种企业级的用于检验应用程序是否如期运行的功能性测试工具。通过自动捕获，检测，和重复用户交互的操作，WinRunner 能够辨认缺陷并且确保那些跨越多个应用程序和数据库的业务流程在初次发布就能避免出现故障，并且保持长期可靠运行。
  - **Loadrunner**：是一种预测系统行为和性能的负载测试工具。通过以模拟上千万用户实施并发负载及实时性能监测的方式来确认和查找问题，LoadRunner 能够对整个企业架构进行测试。通过使用LoadRunner ，企业能最大限度地缩短测试时间，优化性能和加速应用系统的发布周期。

Mercury公司测试工具

- **TestDirector：**是基于Web的测试管理解决方案，它可以在公司内部进行全球范围的测试协调。TestDirector能够在一独立的应用系统中提供需求管理功能，并且可以把测试需求管理于测试计划、测试日程控制、测试执行和错误跟踪等功能融合为一体，因此极大地加速了测试的进程。 TestDirector提供完整且无限制的测试管理框架，实现对应用测试全部阶段的管理与控制。
- **QuickTest Professional：**:是一个功能测试自动化工具，主要应用在回归测试中。QuickTest针对的是GUI应用程序，包括传统的Windows应用程序，以及现在越来越流行的Web应用。它可以覆盖绝大多数的软件开发技术，简单高效，并具备测试用例可重用的特点。其中包括：创建测试、插入检查点、检验数据、增强测试、运行测试、分析结果和维护测试等方面。

Rational公司测试工具

- Rational（http://www-900.ibm.com/cn/software/rational/）公司产品如下：
  - **Rational Functional Tester**：对 Java、Web 和基于 VS.NET WinForm 的应用程序进行高级自动化功能测试。
  - **Rational Functional Tester Extension for Terminal-based Applications**：扩展了Rational Functional Tester，以支持基于终端的应用程序的测试。
  - **Rational Manual Tester**：使用新测试设计技术来改进人工测试设计和执行工作。
  - **Rational Performance Tester**：检查可变多用户负载下可接受的应用程序响应时间和可伸缩性。
  - **Rational Purify for Linux and UNIX**：为 Linux 和 UNIX提供了内存泄漏和内存损坏检测。
  - **Rational Purify for Windows**：为 Windows 提供了内存泄漏和内存损坏检测。

Rational公司测试工具

- **Rational PurifyPlus** 企业版：为 Windows、Linux 和 UNIX 提供了运行时分析。
- **Rational PurifyPlus for Linux and UNIX**：为 基于 Linux 和 Unix 的 Java 和 C/C++ 开发提供了分析工具集。
- **Rational PurifyPlus for Windows**：为基于 Windows 的Java、C/C++、Visual Basic 和 托管 .NET 开发提供了运行时分析。
- **Rational Robot**：客户机/服务器应用程序的通用测试自动化工具。可以对使用各种集成开发环境 (IDE) 和语言建立的软件应用程序，创建、修改并执行自动化的功能测试、分布式功能测试、回归测试和集成测试。
- **Rational TestManager**：提供开放、可扩展的测试管理。
- **Rational Test RealTime**：支持嵌入式和实时的跨平台软件的组件测试和运行时分析。

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。

一些开源测试工具

- Selenium

一个用于Web应用程序的功能测试工具。Selenium测试直接运行在浏览器中，就像真正的用户在操作一样。支持的浏览器包括IE(7、8、9)、Mozilla Firefox、Mozilla Suite、Google Chrome等。

这个工具的主要功能包括：

测试与浏览器的兼容性——测试你的应用程序看是否能够很好得工作在不同浏览器和操作系统之上。

测试系统功能——创建回归测试，检验软件功能和用户需求。支持自动录制动作和自动生成 .Net、Java、Perl等不同语言的测试脚本。

一些开源测试工具

- 一些免费的功能测试工具
  - Appium

    一个支持移动App的开源、跨平台的自动化测试工具，用于测试原生和轻量移动应用，支持 iOS, Android 和 FirefoxOS 平台。
  - Autoit

    是一个使用类似BASIC脚本语言的免费软件,它设计用于Windows GUI中进行自动化操作。它利用模拟键盘按键，鼠标移动和窗口/控件的组合来实现自动化任务。
  - **Linux Test Project**（http://ltp.sourceforge.net/ ）：

    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。

一些开源测试工具

- 功能测试工具
  - **MaxQ**（http://maxq.tigris.org/）：

    MaxQ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昂贵的商用测试工具。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。
  - **WebInject**（http://www.webinject.org/）：

    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。

一些开源测试工具

- 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。

一些开源测试工具

- 性能测试工具
  - DBMonster （http://dbmonster.kernelpanic.pl/ ）：

    一个生成随机数据，用来测试SQL数据库的压力测试工具。
  - **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。

一些开源测试工具

- **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。

- **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。

一些开源测试工具

- 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

一些开源测试工具

- 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）

自动化测试解决方案举例

（来源：百度百科）

- 公司背景介绍
- 公司应用系统的情况
- 公司软件测试现状
- 可供选择的方案
- 方案评价

自动化测试解决方案举例

- 公司背景介绍

A公司是一家大型保险公司，拥有近20个城市的分公司，并在其中5个城市建立了IT支持中心。平均每年的上线应用数量在20个左右（新业务系统和原有业务系统的主要版本发布）。

目前A公司的专职测试团队人数不足30人，而且测试团队的测试人员技能参差不齐，目前测试只是作为项目上线前的一道工序而已。在测试团队内部也几乎没有自动化的手段，主要依靠手工测试。

由于已上线应用系统的问题，开发团队必须分出一部分资源去维护和修复上线应用，而同时测试团队的测试成果和效率却无法和这些应用质量挂钩，也更无从谈起对软件质量的控制。所以，A公司决定在软件质量和测试方面进行投入，他们考虑以下几方面：

自动化测试解决方案举例

- 公司背景介绍
  - 引进软件测试流程管理的自动化，提高软件测试过程的管理水平，使软件测试和软件开发一样可被评估、被衡量。
  - 实现性能测试自动化，所有应用上线之前必须有应用性能风险评估报告和相关部门的确认
  - 逐步实现功能测试的自动化，在目前人员配置的情况下，把部分手工测试变成自动化测试，提高测试可信度，降低人为错误。
  - 通过软件测试自动化，管理软件测试中的案例、缺陷、报告等资产，进一步提升软件测试的效率并建立测试基础库。
  - 在规划中，将来的2～3年内使所有的应用系统上线都必须有数字化的测试数据作为依据。

自动化测试解决方案举例

- 公司应用系统的情况

  由于保险公司的业务种类繁多，同时在经过了几十年的经营后，公司内的应用系统从早期的终端方式到现代的J2EE和.NET等应有尽有，鱼龙混杂。

  IT部门已建立3年规划，即在未来的3年时间内将所有终端和C/S方式的应用转换成B/S架构，但当前仍然需要对这些旧应用系统进行维护，以保证业务的顺利进行。

  对于开发部门来说，目前新应用开发基本上以B/S架构为主，主要是WebHTTP应用和部分.NET Form应用。

自动化测试解决方案举例

- 公司软件测试现状

企业机构在做测试自动化选型时一定要考虑清楚企业内部哪些部分可以实施自动化、哪些部分暂不实施自动化、哪些部分仅在某几个项目做自动化试点。切忌匆忙上马或盲目否定，缺乏实事求是的理性思考。

自动化测试解决方案举例

- 公司软件测试现状

  测试部门目前仅负责系统测试和对用户验证测试进行管理，对于之前的单元测试和集成测试主要由开发团队中划分出的一部分临时测试人员完成。由于缺乏监测手段，测试部门也无法收集和确定集成测试和单元测试的完成情况，在整个软件测试过程中，业务需求是由开发部门进行管理，但测试需求目前尚没有提出要求，测试案例主要通过在公司公用的文件服务器中的目录管理方式管理，对测试中缺陷流程等管理主要依靠邮件的流转进行处理。目前90％以上的测试是通过Excel和Word等测试案例文档来完成，测试人员对软件测试自动化的认识仅停留在“记录＋回放”的认识上。

自动化测试解决方案举例

- 可供选择的方案

  方案A:

  A公司可以采用HP-美科利（HP-Mercury）公司产品为主的软件测试自动化方案。

  - 依照原先的邮件流转过程配置TestDirector缺陷管理流程，为每个保险业务的开发小组和测试团队分配相应的用户许可证，取消原有邮件方式。
  - 部署QuickTestProfessional，以便完成应用程序相关功能测试。
  - 部署LoadRunner。从测试团队中分化出专职的性能测试自动化工程师和小组，和业务部门协调，建立A公司应用系统上线性能指标，通过LoadRunner给出测试指标。

自动化测试解决方案举例

- 可供选择的方案

  方案A:

  - 建议A公司成立专门的质量控制部门，对TestDirector中的数据定期进行分析，建立相关质量模型，以便于企业量化管理和过程改进。

自动化测试解决方案举例

- 可供选择的方案

  方案B:

  A公司也可以采用IBMRational产品为主的软件测试自动化方案。

  - 采用Testmanager来进行整个测试流程的管理，为相关开发和测试小组成员分配相应权限，改变以前通过邮件以及Word、Excel文档管理测试的工作方式。
  - 部署Robot，用它来完成功能相关的测试工作以及新版本发布时的冒烟测试。此外，Robot也能较好地完成性能相关测试。统一的操作方式降低了工具的学习周期和培训带来的大笔开销。
  - 部署Purifyplus，使测试工作前移到开发阶段。由于Purifyplus能较好地支持白盒测试，编程人员在编码阶段引入的错误能尽早被检测到，这大幅降低了后期测试的开销。

自动化测试解决方案举例

- 可供选择的方案

  方案B:

  A公司也可以采用IBMRational产品为主的软件测试自动化方案。

  - 建议A公司成立专门的质量控制部门，对Testmanager中的数据定期进行分析，建立相关质量模型，以便于企业量化管理和过程改进。

自动化测试解决方案举例

- 可供选择的方案

  方案C:

  A公司也可以采用开源软件为主的软件测试自动化方案。

  - 采用Bugzilla来进行Bug跟踪管理，采用BugzillaTestRunner进行测试用例管理，采用CVS进行测试资源的配置管理。
  - 采用MaxQ和WebInject对B/S结构的应用系统进行功能测试。
  - 采用DBMonster、Open—STA、LoadSim进行性能相关测试。
  - 可采用Xunit架构的开源工具对不同语言的程序单元进行单元测试。

自动化测试解决方案举例

- 可供选择的方案

  方案C:

  - 建议A公司成立专门的开源软件维护小组，以解决可能会碰到的工具维护工作。
  - 建议A公司成立专门的质量控制部门，对Bugzilla、TestRunner、CVS中的数据定期进行分析，建立相关质量模型，以便于企业量化管理和过程改进。

自动化测试解决方案举例

- 方案评价

由于不同客户在组织架构、员工素质以及流程管理水平等方面的不同，很难用一两句话来说明不同解决方案的适用性。上面3种可行的方案，具体选择哪一个，需仔细权衡。

一般性建议：

对于不想受制于某个测试自动化厂家的企业，开源是一个理想的选择。它不需要支付成本，工具的源代码可以随意修改，因而具有较好的灵活性。但开源工具的弊端也是明显的：缺乏使用培训和技术支持，没有正确保证，工具的用户界面一般也较为粗糙。

对于比较看重培训和售后支持的企业，建议选择IBMRational或HP(Mercury)或其他厂家的产品。这样虽然需要支付一部分费用，但省去了工具维护所需要的大量工作。