

---

# 目 录

---

WAP 性能测试 .....	1
如何做好 .net 自动化测试 .....	5
Cookie 测试及案例分析 .....	12
QTP 字典对象的强化 .....	24
全客户端操作 TestDirector 项目数据移植 .....	37
了解软件性能测试 .....	41
如何测试大型 ERP 系统 .....	48
性能测试方法论 .....	51

## WAP 性能测试

作者：陈雾

随着手机应用的逐步普及,对于 WAP 的测试也逐渐进入了性能测试的阶段,但传统的性能测试工具却往往在这个时候歇了火,这里探讨一下如何使用传统工具来实现对 WAP 应用的性能测试及常见问题。

首先我们先来了解一下什么是 WAP?

WAP (Wireless Application Protocol) 是一种向移动终端提供互联网内容和先进增值服务的全球统一的开放式协议标准,是简化了的无线 Internet 协议。WAP 将 Internet 和移动电话技术结合起来,使随时随地访问丰富的互联网络资源成为现实。说得更简单一些, WAP 是手机上网的一种通讯协定,其意义相当于 TCP/IP。由于手机的画面有限,所呈现的网页也必须做精简化处理,所需要的网页编写语言就是 WML,相当于 HTML。而 WML (Wireless Markup Language - 无线标记语言) 这种描述语言同我们常听说的 HTML 语言同出一家,都属于 XML 语言这一大家族。HTML 语言写出的内容,我们可以在我们的 PC 机上用 IE 或是 Netscape 等浏览器进行阅读,而 WML 语言写出的文件则是专门用来在手机等的一些无线终端显示屏上显示,供人们阅读的,并且同样也可以向使用者提供人机交互界面,接受使用者输入的查询等信息,然后向使用者返回他所想要获得的最终信息。

作为性能测试首先要做的就是录制脚本,这里我们尝试录制 3g.baidu.com 这个 WAP 网站。使用 LoadRunner 的 VUGen 选择 Web (HTTP/HTML) 协议,录制回放后我们会发现 WAP 的网站和 HTML 的网站在这里访问并无任何的区别,这是因为 WML 的实现原理本来就和 HTML 没什么太大区别,只是数据包的格式略有区别而已。所以对于很多 WAP 的应用我们可以直接采用过去对 HTTP 性能测试的方法和流程,但是如果我们把网站换成 sina.cn,这个时候问题出现了。当我们使用 IE 浏览器去访问 SINA 的 WAP 网站时,会发现服务器会自动把我们跳转到一个固定的 HTML 页面上,并没有和手机一样打开 WAP 应用,而如果访问 wap.kaixin001.com 更会直接弹出无法访问的页面错误,对于这种网站性能测试的脚本开发视乎突然无计可施了。

那么为什么使用 IE 访问 3g.baidu.com 就可以,但是访问 sina.cn 就不行呢?这是由于 WML 和 HTML 还是有一定区别的,为了在手机上能够得到最好的显示效果,当客户端连接 WAP 应用时需要对客户端的类型进行判断,如果是手机访问则跳转到 WAP 应用,否则跳转到 HTTP 应用。现在一般最常见的 WAP 客

户端判断方法是根据浏览器发送的 User-Agent 数据段来判断。

现在我们换一个浏览器试试？选择 Opera 浏览器再访问一次上面 IE 无法正常访问的两个 WAP 应用，这个时候我们会发现，在 Opera 浏览器中能够正常的读取 WML 内容了。

为了找到两个浏览器访问 WAP 的区别，接着我们使用 VUGen 的 Windows Sockets 对 IE 和 Opera 这两个浏览器数据包进行一下录制。录制结果如下

Opera 的发送数据包：

```
send  buf0 523
      "GET /?from=sinacn HTTP/1.1\r\n"
      "User-Agent: Opera/9.80 (Windows NT 6.0; U; zh-cn) Presto/2.2.15 Version/10"
      ".00\r\n"
      "Host: 3g.sina.com.cn\r\n"
      "Accept: text/html, application/xml;q=0.9, application/xhtml+xml, image/png"
      ", image/jpeg, image/gif, image/x-xbitmap, */*;q=0.1\r\n"
      "Accept-Language: zh-CN,zh;q=0.9,en;q=0.8\r\n"
      "Accept-Charset: iso-8859-1, utf-8, utf-16, */*;q=0.1\r\n"
      "Accept-Encoding: deflate, gzip, x-gzip, identity, */*;q=0\r\n"
      "Cookie:mode=10081149-c; stat=0909281221019411528549; la=10081149-3\r\n"
      "Cookie2: $Version=1\r\n"
      "Connection: Keep-Alive\r\n"
      "\r\n"
```

IE 的发送数据包:

```
send  buf0 577
"GET / HTTP/1.1\r\n"
"Accept: image/gif, image/x-bitmap, image/jpeg, image/pjpeg, application/x"
"-ms-application, application/vnd.ms-xpsdocument, application/xaml+xml, app"
"lication/x-ms-xbap, application/x-shockwave-flash, application/x-silverlig"
"ht, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/m"
"sword, */*\r\n"
"Accept-Language: zh-cn\r\n"
"UA-CPU: x86\r\n"
"Accept-Encoding: gzip, deflate\r\n"
"User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; SLCC1; .NET"
" CLR 2.0.50727; .NET CLR 3.5.30729; InfoPath.2; .NET CLR 3.0.30729)\r\n"
"Host: sina.cn\r\n"
"Connection: Keep-Alive\r\n"
"\r\n"
```

对比这两个数据包后，我们发现主要的不同只是在 User-Agent 上。接着我们在 VUGen 中尝试用 WEB（HTTP/HTML）协议调用 IE 重新录制一下 sina.cn 的首页请求，默认回放后是错误的 HTML 页面。接着修改 Run-time Setting 中的 Browser Emulation，将浏览器属性中的 User-Agent 修改为手工定义的浏览器类型，类型修改为 Opera/9.80 (Windows NT 6.0; U; zh-cn) Presto/2.2.15 Version/10.00，然后打开日志回放中的服务器返回。当我们再次回放脚本后会发现这个时候服务器的返回内容已经不是 HTML 而是 WML 了。所以对于有客户端校验的 WAP 应用，只需要了解判断数据包的策略。

除了采用 User-Agent 来判断客户端这种方式以外，常见的还有一种通过判断 HTTP\_ACCEPT 中是否支持 WML，这种方式相对可靠一点但也存在一些问题，如果一般的不支持 HTML 的浏览器还好说，只要判断浏览器支持 WML 并且不支持 HTML 就可以，但如果浏览器同时支持 WML 和 HTML 那就难办了。遇到这种问题解决的方式和前面的类似，伪造一个 HTTP\_ACCEPT 信息即可。在 VUGen 中我们可以通过 web\_add\_header() 函数手动构造一个 ACCEPT 数据段，让服务器相信我们是能够解析 WML 格式即可。

对于使用其它开源性能测试工具对 WAP 进行性能测试的思路也是类似的。如果使用 Jmeter，可以先使用 Badboy 录制生成脚本再导出成 Jmeter 脚本。使用 Jmeter 打开该脚本后，修改脚本中 HTTP Header Manager 下的 User-Agent 属性即可。而在 OpenSTA 中只需要修改脚本开头的 User-Agent 常量定义即可。

## 如何做好.net 自动化测试

作者：卢晨之

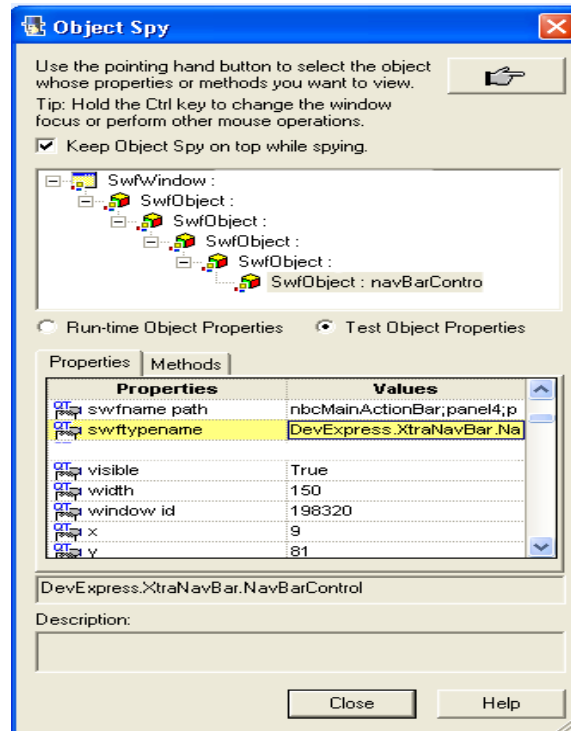
**摘要：**主要介绍了如何做好.net 在 CS 的自动化测试。

**关键词：**.net、自动化测试

QTP 是做 Web 自动化一个很好的工具与利器,但同时 QTP 对.net 在 C#等程序的控件识别能力却不是很理想,所以这也让人望而生畏。可能公司买了 QTP 与.net 的插件,让测试组开始实施这个项目的自动化。但发现测试用例中有 40% 因为控件识别的问题不能实现,使得这部分用例手工完成而浪费大量时间,更糟糕的是你还要面对你上司的责问。其实,你能做得更好。希望读了这篇文章能给大家带来启发。

### ● 控件

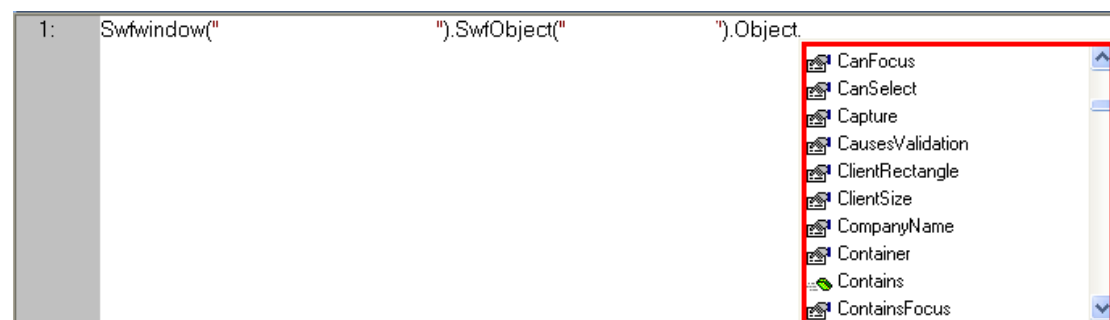
由于微软的控件的外观、功能不是很丰富,所以很多公司采用了第三方控件,这也是 QTP 的.net 插件对它们的识别存在一定的局限性的主要原因。或者说, .net 插件的诞生并非一个完整品,因为这些主流第三分控件也只被封装了部分功能与对象。而面对这些第三方控件,我们需要做的第一件事情就是先获取这个控件的信息。我们使用的是 QTP 的 SPY 功能去获取这个控件的信息,如下图:



我们可以获取到的这个控件包括“CompanyName”，“swftypename”等信息。而我们这里主要获取这个控件的“swftypename”它的值是“DevExpress.XtraNavBar.NavBarControl”。通过这个值，我们知道了这个控件是DevExpress。同样，我们在 QTP 的帮助文档中也可以看到，.net 插件也有部分对象是支持这个控件，同时还支持 Microsoft、Infragistics、ComponentOne 这几个控件。

### ● 使用控件的内置属性与方法

使用这些控件的方法和属性很简单，只要我们添加这个对象进去对象库，而每次透过对象.Object 就可以查看到罗列出的对象方法与属性。如下图：



而实际上也有大部分不能添加到对象库中的对象，我们需要另做处理，后面将再做介绍。

### ● 获取控件的内置属性与方法

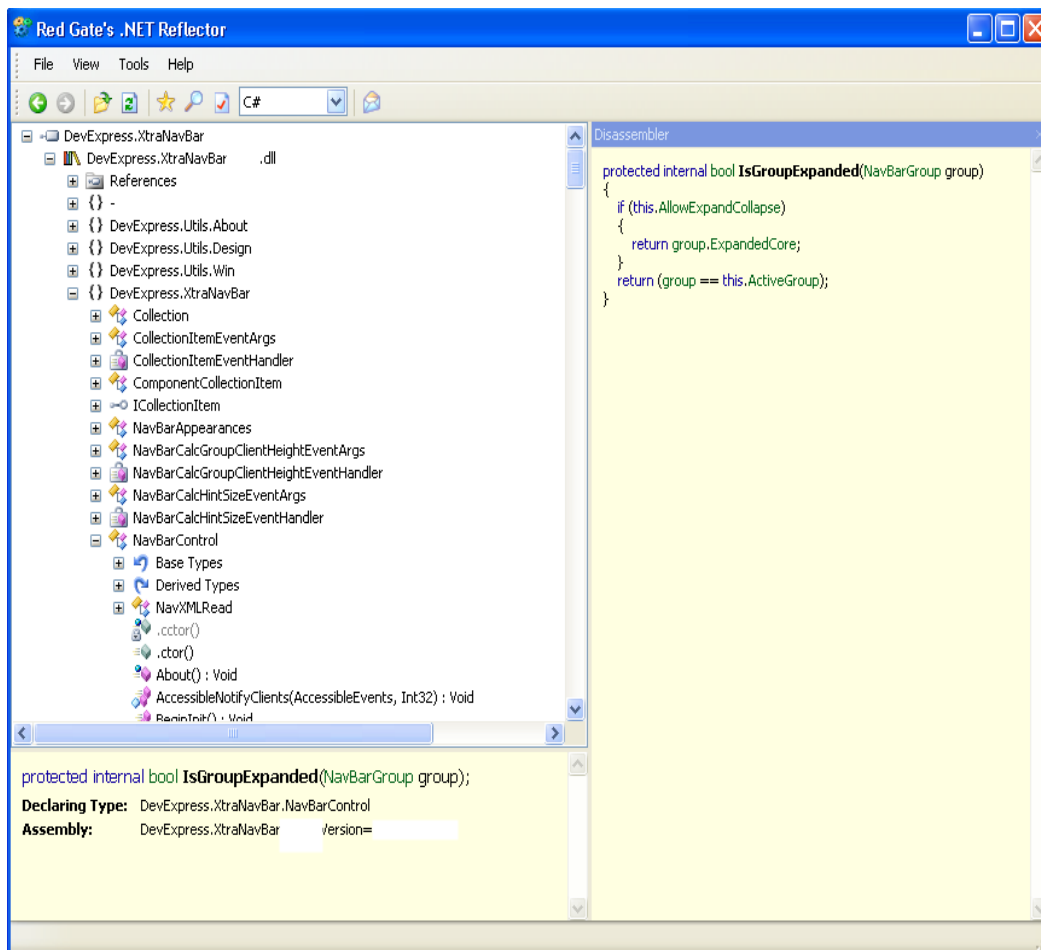
我们知道这个控件的是哪家公司或者是什么类型的目的，是因为我们透过它

可以去查找

跟多关于这个控件的 API，然后调用它去实现自动化。在这里给读者提供 4 个方法：

1. 我们可以在相关的控件主页上，下载到该控件的 API 文档或者在线帮助文档，提供下 DevExpress 的主页：<http://www.devexpress.com>；在这里我们可以使用它的搜索功能，这个对我们的测试有很大的帮助<http://search.devexpress.com>，只要输入我们想要获取 API 的控件名字就可以搜索到它的信息。

2. 我们在本地的客户端文件中，可以找到相关的插件 DLL，而我们也可以使用工具去打开它们，查看它支持的，它继承的方法与属性。例如：Reflector 这款工具；



如上图，我们查询到了 NavBarGroup 这个对象支持的方法和属性，如：IsGroupExpanded 输入的参数是这个 NavBarGroup 的对象，返回值就是 True 或者 False。实际用法：

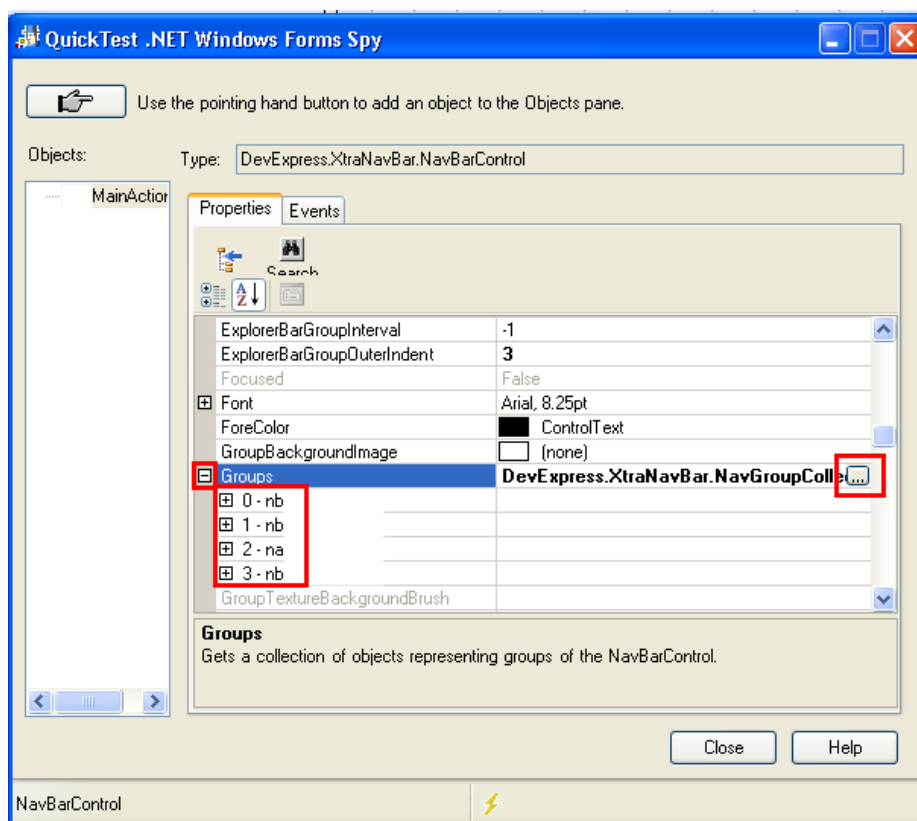
Msgbox Swfwindow().SwfObject().Object.Groups.item(0). IsGroupExpanded

3. 使用 QTP 的 SPY 功能，这个就比较直接与快捷，可以在 RunTime Object



这栏中，看到它的属性与支持的方法。但这个 SPY 的工具具有它自身的局限性，这个也是由于 QTP 控件识别的问题引致的，某些控件如果识别不到，就只能抓取到最近一个能识别到的 Parent 层对象。

4. 通过 QTP 中 .net 插件带的工具，.net Windows Forms Spy。这个工具和上面的 SYP 差不多，但它比 SYP 工具好的地方就是它能够查询到它包含的子控件，或者说，它能省去你代码调试与对象查找的时间。例如一个 NavBarControl 对象中存在的 Link 对象，但 QTP 能够识别到的对象只到了 NavBarControl 这一层。如果初次接触你完全弄不懂这个控件下面的这些东西是什么，如何获取到。而当你利用这个工具时候，你可以看到原来它下面还有 2 层，包括了 Group 层和 Link。



## ● 对象挖掘

正如上文提到的，QTP 对象识别的局限性导致了大部分对象无法添加到对象库中，或者不知道这个对象是什么，我们一般使用的是 .net Windows Forms Spy 这个工具，它能够很迅速的挖掘到我们需要的。但某些情况下，我们也需要借助像 Reflector 这类工具的帮助。

如果这些都无法帮助到你，有另外 2 个方法，一个是通过看这个控件的实现代码片，去了解它的构造，另一个是查看该程序的源代码。这 2 个都能够让你知道它是怎么工作，怎么添加子对象进去的，自然就可以逆转过来获取到它们的子对象与跟深一层的对象方法和属性。对象挖掘是一件很让人兴奋但又费神的工

作。这是一个比较有趣的例子，也是非常费神的一个挖掘工作：

```
Swfwindow("").SwfObject("").Object.Properties.PopupControl.Controls.item(0).VisibleRows.item(1).Properties.Caption
```

## ● 封装

如果你琢磨了这个对象用了 1 个小时，一个项目组中其他人或者也需要花上同样的时间去琢磨它，那么这个效率就一定非常低下了，所以我们需要把这些对象的方法和属性封装起来。我们分 2 层：

第一层：相同类型对象封装。有三个问题是我们必须面对的，对象挖掘的复杂性，重复性与多样性。如同在 DevExpress，一个下拉框就有 4 到 5 种（popupcontaineredit、lookupedit、imagecomboboxedit、comboboxedit 等等），但它们对对象的“选值”操作与“取值”却又有部分差异，所以我们把这些类型的对象封装到一起，写成同一个 function，并通过传递下拉对象与值来实现“选值”之类的操作。

第二层：业务操作层。更快捷，更高效的做完一个用例是我们所期待，但如果它是建立在你要先 Spy 看看这个对象是什么的基础上，并对“症”用 Function 的基础上，可见这个效率同样不高。所以我们还必须再封装多一个业务操作层，例如设置订单信息中的相关信息等等。

## ● 使用这些控件方法与属性的建议：

1. 部分控件在封装时候，需要加多 Editable 或者 visitable 等属性加以判断，因为部分的方法无论是否能够修改都会修改掉这个值。包括 QTP 的内置方法，SwfTable 中的 SetCellDataSetCellData 等等，也就是说这个是存在一定风险，没能很好的检测我们需要的结果。

2. 最好使用鼠标控制结合键盘操作。使用 Swfwindow().type 结合 Swfwindow().swfObjet().click x,y 方法操作。X, Y 的获取一般可以通过对象的 Bounds 属性取得它的坐标。例如：

```
Location=MenuObj.Object.VisibleLinks.item(0).Bounds
x=cdbl(Split(Split(Location,"")(0),"=")(1))+cdbl(Split(Split(Location,"")(2),"=")(1)/2)
y=cdbl(Split(Split(Location,"")(1),"=")(1))+cdbl(replace(Split(Split(Location,"")(3),"=")(1),"",")/2)
SwfwinObj.Activate
MenuObj.click x,y
```

3. 使用 Object.GetType.Name 能够帮你得到当前对象的类型，这个也能让你更快的搜索出相关的 API。

## ● 实例

透过上面与大家分享的理论知识，现在我们先挑个“软柿子”捏捏，这个就是 SwfTable。我们打开 QTP 的帮助文档，看到文档中正好提示支持这个控件，DevExpress XtraGrid。如果使用这个控件，你或许会发现某些 Table 的方法它是不支持的，因为 QTP 没封装得很彻底的缘故。

1. 我们添加这某个 Table 进对象库，而我们 Spy 出来的 type 是 DevExpress.XtraGrid.GridControl 这个控件。

2. 使用 .net Windows Forms Spy，查看这个对象，或者查看源代码与网上搜索相关代码，不难发现这个对象下面有一个 MainView 或者 Views 的对象，它就是存放了大部分对象的一个“容器”。而我们在 API 中看到，这个 MainView 其实包括了 3 种 View 类型: GridView, CardView, TreeView。

3. 而我们通过下面代码获取这个 View 是属于什么类型：

Swfwindow().swfTable().Object.MainView.getType.name

4. 到这里，你如果看到的这个 View 类型是属于 CardView，那么 QTP 的 .net 插件是没有封装这个的，它只封装了 GridView。而 CardView 的最大特点就是 Column 变成了 Row，而 Row 变成了 Column。

5. 所以我们需要再一次把 CardView 这个 View 类型的方法与属性封装出来。

```
Public function GetCellPorp(SwftableObj,Row,Column,PropertyName)
    View=SwftableObj.Object.MainView.getType.name
    Select Case View
        '由于 QTP 支持了这个对象的方法，我们就直接使用它方法，如果还想
        Case "GridView"
            GetCellPorp=SwftableObj.GetCellProperty(Row,Column,PropertyName)
        Case "CardView"
            'CardView 的特点就是 Column 与 rou 相反，但为了更好的使用，我们需要在封装的时候再倒过来，让它适合我们的感知。
            '再往下，我们需要分析出这个 Row 与 Column 的值，是字符串或是数字，如果是字符串需要做循环找出这个 Column 对象或者 row 的 handle。
            'Code 省略

            '接着我们需要在这个封装中添加进我们的 PropertyName 传参类型的分类
            Select Case PropertyName
                Case "Value"
                    'Code 省略
                Case "Colname"
                    'Code 省略
            End Select
        Case "TreeView"
            'Code 省略
    End Select
End Function
```

## ● 结语

熟悉 web 自动化的人，觉得 Web 很简单，也很好做，因为我们够了解它，也能很轻松了解到它的构造，能够得心应手的操作它。但面对一个 CS 程序，完全的看不到代码和不了解的黑盒子，不了解它是如何运作的情况下，让我们觉得 CS 的自动化离我们太远了，希望读完这篇文章能让您有所收获。

学习自动化测试是一个很漫长的过程，从理论到工具再深造理论再脱离工具的一个过程，除了项目进度关系要求使用工具外，我们应该多研究这个 QTP 是如何实现的，脱离它才是我们需要得到本事，试着去了解 Window 或者其它操作系统后台工作原理，离开了工具，你也能造出车来，这也是我们能在工作与项目组中处于不可替代的位置。

如何更好的做好.net 的自动化测试，还有很多需要探讨与研究的。欢迎大家加我 MSN:Luchenzhi@hotmail.com

## Cookie 测试及案例分析

作者：Richard Brauchle 译者：刘英

**摘要：**在网站中被用于交换文件的协议通常是无状态的，但是大多数网站又必须维持这种状态。因此，为了维持这种状态，开发人员采用的一种方法就是使用 cookies。本文将提供基本的技术背景和真实案例来帮助读者了解 cookies 怎样工作以及怎样测试系统的 cookies。并使用 amazon.com 网站作为案例来演示 cookies 测试技术。

### 有状态系统和无状态系统

根据 whatis.com 的定义，无状态系统是“不记录先前的交互和每次交互请求处理完全基于其本身的信息”，相反，有状态系统是记录先前的交互。

为了详细说明无状态系统，我们将会提到 Hypertext Transfer Protocol (HTTP) 协议。

### HTTP 的无状态性

如果在地址栏中输入 <http://www.testwareinc.com>，回车，在浏览器和 Testware 服务器之间将开始会话：

浏览器：“嗨！Testware 服务器，可以请问有’<http://www.testwareinc.com/index.html>?’这个网页吗？”

Testware 服务器：“是的，你请求的这个网页时存在的。”

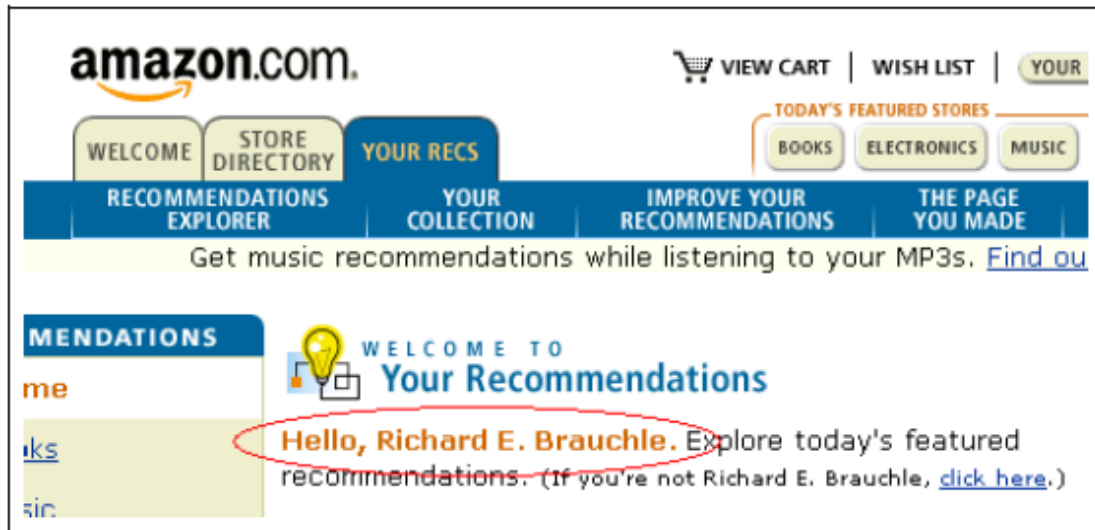
Testware 服务器：“这是这个网页的文本：<文本内容…>。”

一旦浏览器收到使用 http 协议请求的 index.html 的最后一个字节，Testware 服务器必然会“忘记”关于你的请求。如果你现在浏览网站的其他页面，Testware 服务器会把上面的操作作为新的请求响应，而不会记住你早先的请求了。这对于 Testware 不是什么坏事；因为它根本不需要知道你早先请求来做出新请求的响应。但是这种状态对于基于 web 方式的系统就没有关系了吗？

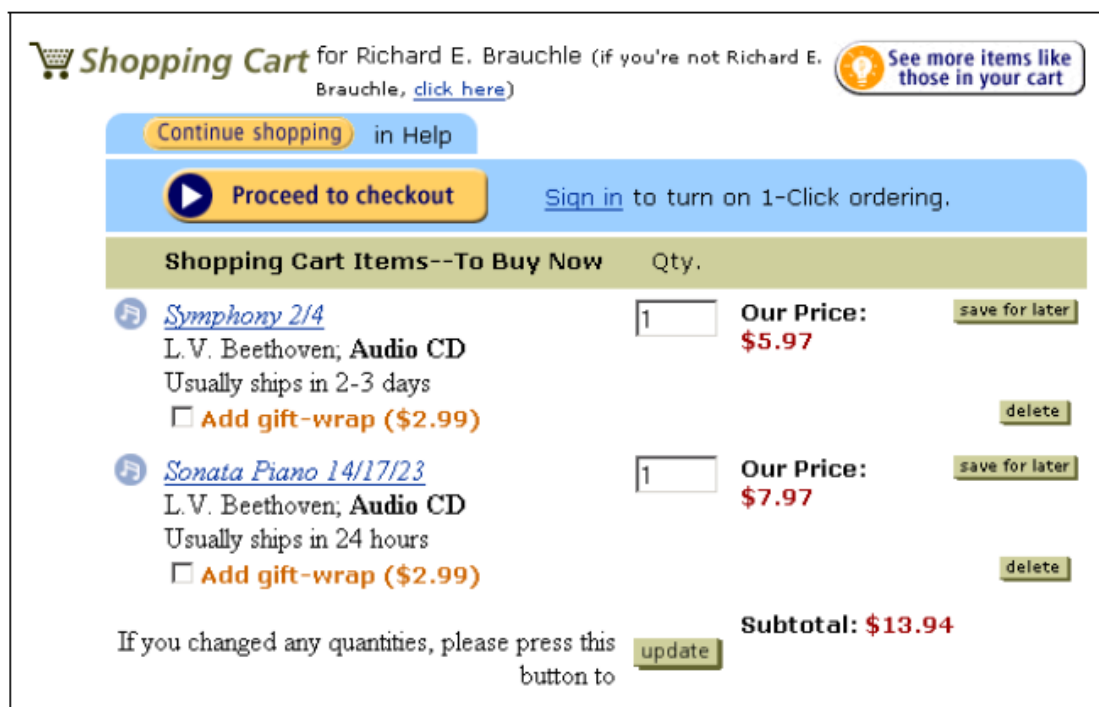
### 网页中的状态或无状态

我们很快会看到，有无状态是网站的问题！amazon.com 是一个深受书籍和音乐爱好者的在线购物提供商。如果没有办法克服 http 的无状态属性和网站上状态的维持，那么下面的操作是不可能发生的：

● “hello, <your name>”：关于 amazon 主页反馈给顾客的信息，如果没有状态，那么这个网站怎么知道你的名字？



● 虚拟购物车：在无状态的情况下，amazon 怎样保持对单个商品和所购商品数量进行追踪了？



那么 Amazon 是怎样通过无状态的 Http 协议来完成以上“图片”中的操作了？部分答案是 cookies。

### 使用 cookies 维持状态

Whatis.com 是这样定义 cookie 的“把网站上的信息放入到你的硬盘，以至于能记住你上次所操作的内容。”为什么？“基于网页的每个请求都是独立于其他请求的，网页服务器不会记住先前送达给每个用户的网页，或者任何关于你先前的操作。”



那么 cookie 是怎样依靠浏览器和操作系统来储存的了？IE 使用单独的文件来存储每个 cookie，通常在操作系统的 Window 目录下（C:\Documents and Settings\[User Name]\Cookies）。NS 储存所有的 cookie 使用独立的文件，并且命名为 cookies.txt，通常在 NS 的安装目录下。firefox 在 firefox 的安装目录下（C:\Documents and Settings\[User Name]\Application Data\Mozilla\Firefox\Profiles\xxxxxxx.default\）

### Per-Session Cookies 和 Cookies 过期

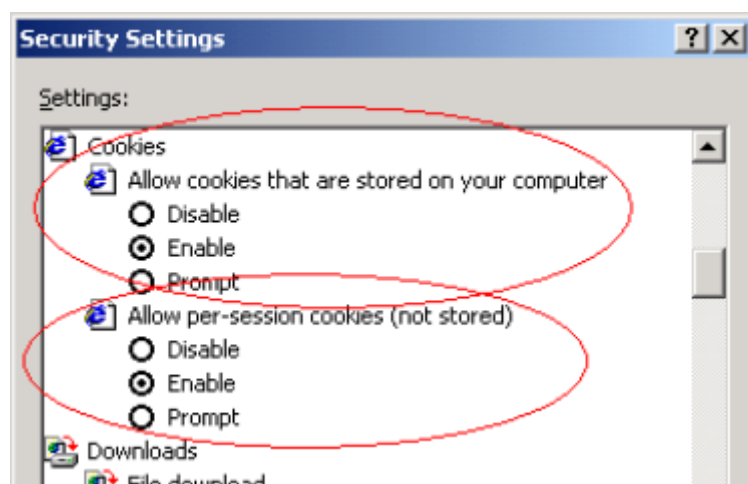
当 web 服务器设置了一个 cookies 在你的系统中，你就能给 cookies 一个过期时间。一旦时间匹配，所有过期的 cookies 都将会被删除。

如果 web 服务器没有给 cookie 一个过期时间，cookie 就是 per-session cookie。当你关闭浏览器时 Per-session cookie 将自动删除，他们只存在你刚开始打开网页到关闭网页那会。

### 检测 Cookie

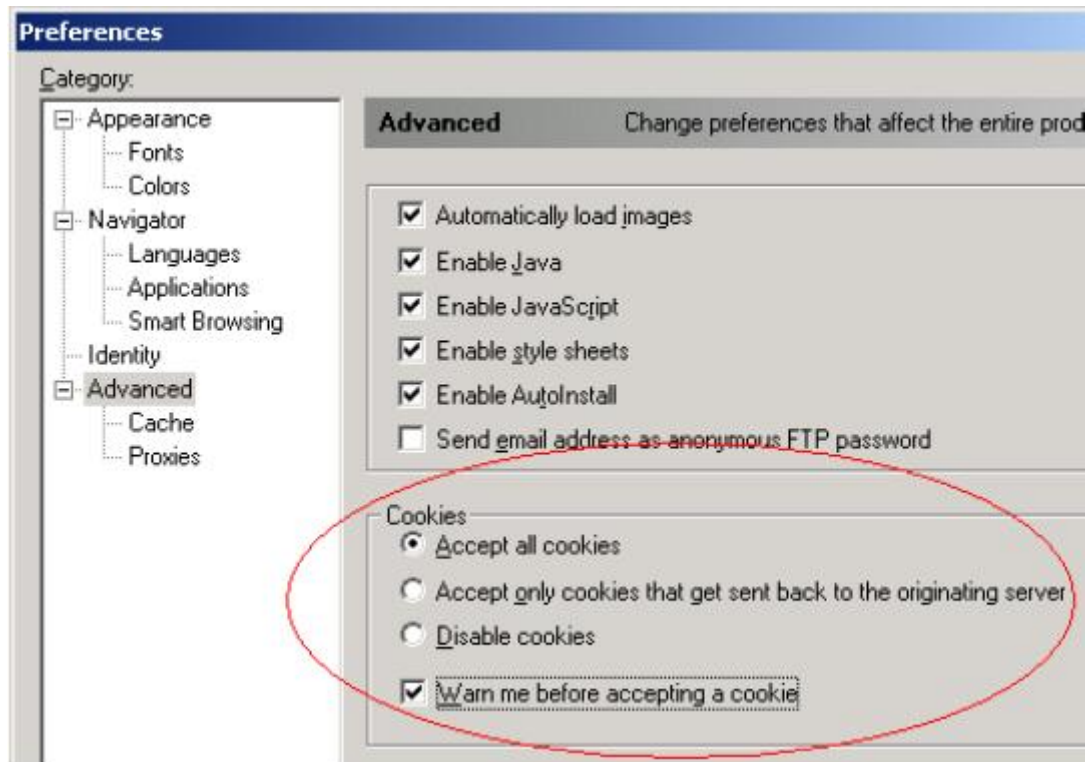
如果 web 系统需要你测试 cookie，那么你将怎样测试了？如果有相关的网站设计文档和功能概要等，那么你首先读这些文档。但大多数情况下，特别是缺乏这些文档的时候，以下方法是非常有用的：

- 找到你 PC 机中 cookie 储存目录
- 删除所有存在的 cookies。
- 设置浏览器选项“提示 cookie”，IE 中，选择“工具”→“ie 选项”→“安全”→“自定义级别”→选择“Allow cookies that are stored on your computer”和“Allow per-session cookie (not stored)。”下的 enable 按钮。（IE 6.0 中是工具——internet 选项——隐私——设置(设置为中或是低)。

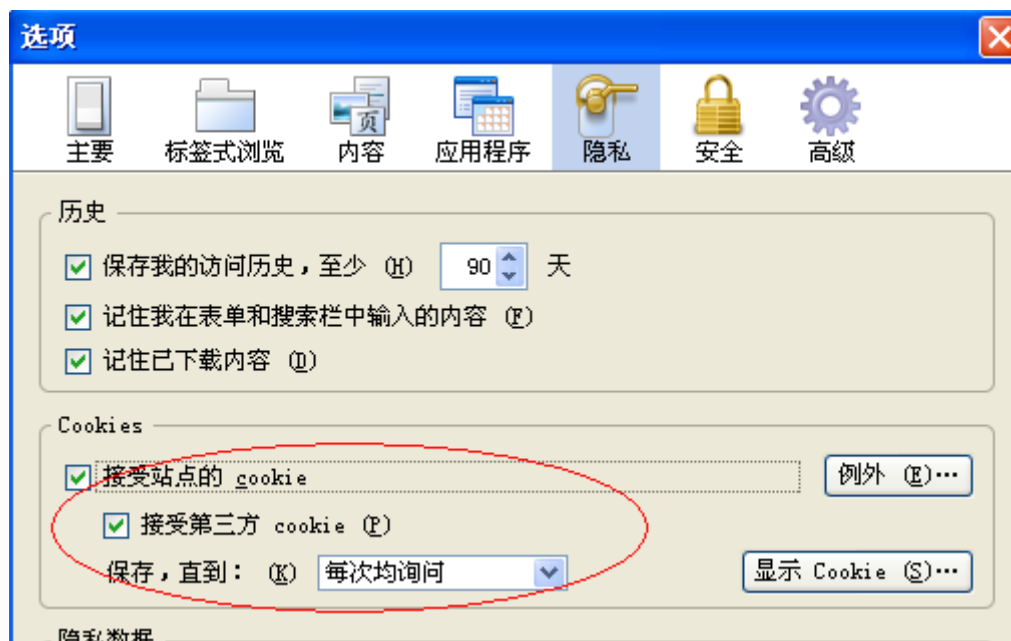


在 Netscape，选择编辑→参数选择→高级→选择“warn me before accepting a

cookie”

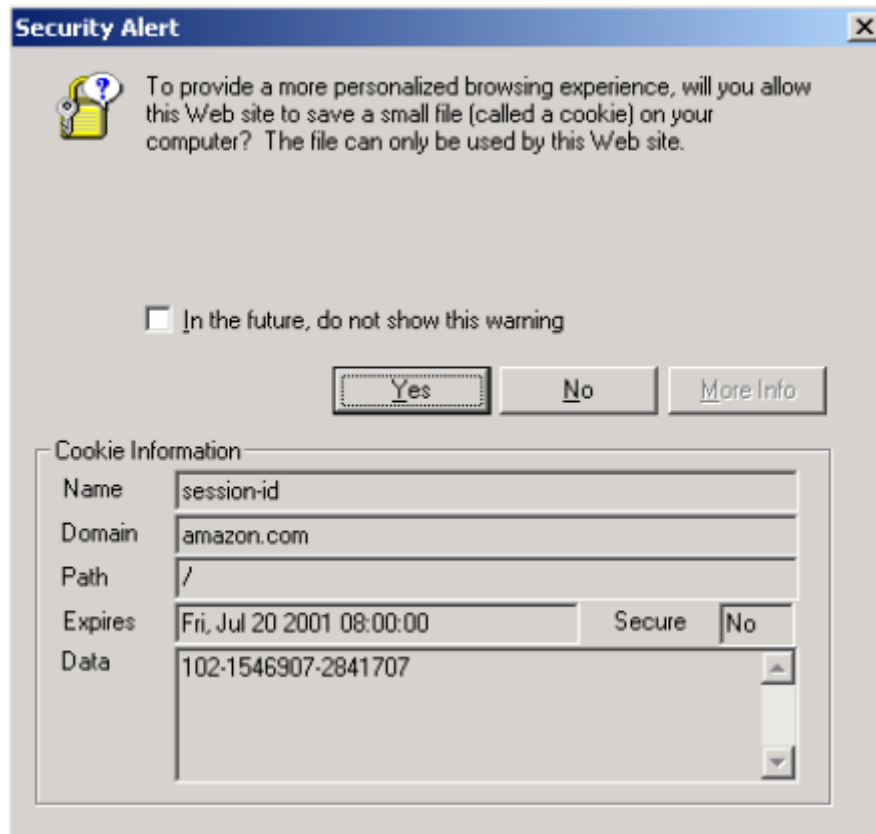


在 Firefox 中选择工具→隐私→cookies→选择“接受站点 cookie”，“接受第三方 cookie”，保存，直到每次访问



- 地址栏通过网站上所有主要特征和功能来查询 cookie 路径
- 怎样知道 cookie 路径？当网站企图记录状态信息到你的 PC 机的 cookie 中，你将得到一个提示信息，IE 提示如下：





NS 使用如下提示:



● 每次这个对话框出现时，都会记录 cookie 的详细信息和什么操作引起 cookie 被创建或修改。那么点击“yes”接受 cookie。打开 cookie 文件，复制/粘贴 cookie 详细信息到“cookie log”中，用于之后的观察和分析。保留这些数据，包括 cookie 名字，数量，创建 cookie 活动日志相关联的活动。注意：一些网页对 cookie 是非常活跃的，设置或者修改你访问的每一个网页的 cookies。创建一个网站不同类型的 cookie log，将花费很长的时间，并且会使你在一定程度很烦躁。最好的选择是首先从开发人员那里得到尽可能多的 cookie 活动。

## Amazon.com 对 cookie 的使用

现在让我们通过 amazon.com 怎样使用 cookie 来更加明确目前为止我们已经讨论的 cookie 的概念。在分析之前，我们也会遇到在 cookie 测试中一个普遍的问题-----搞清楚 cookie 字段的意义！

在开始之前，我删除了储存在我 PC 机中所有的 Nets cookies，并且设置 cookie 选项来提醒我。接下来输入 [www.amazon.com](http://www.amazon.com) 到地址栏

amazon 服务器创建的第一个 cookie 活动信息是：

amazon.com TRUE / FALSE 994320128 session-id 102-7224116-8052958

这个信息表明 cookie 将于 7/5/2001 过期（我也将开启剩下的两个部分来了解更多详情）

amazon 服务器创建的第二个 cookie 活动信息也表明 7/5/2001 过期。

.amazon.com TRUE / FALSE 994320181 session-id-time 994320000

amazon 服务器的第三个 cookie 内容如下，过期时间为 1/1/2036。

.amazon.com TRUE / FALSE 2082787330 ubid-main 077-4356846-2652328

第四个 cookie 是一个 per-session cookie，由于 NS 的提示并没有包含过期时间。因此 per-session cookie 也没有写入到硬盘中。解释这个 cookie 也只能通过 NS 的提示对话框了。



第五个 cookie 过期时间为 1/1/2036，包含以下数据：

.amazon.com TRUE / FALSE 2082787787 x-main hQFiIxFHUFj8mCscT@Yb5Z7xsVsOFQjBf

在接受完第五个 cookie，amazon.com 主页也加载完了，URL 是：

<http://www.amazon.com/exec/obidos/subst/home/home.html/102-7224116-8052958>

可以看到在 URL 末尾是一串数字，这是一个 session ID 储存在第一个 cookie 中的。

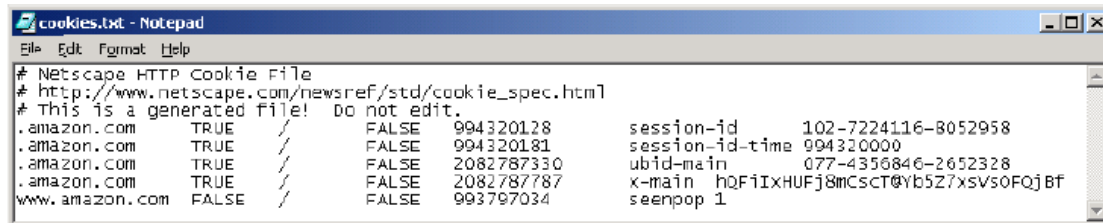
第六个 cookie 过期时间为 6/29/2001，内容如下：

www.amazon.com FALSE / FALSE 993797034 seenpop 1

当正在接收上面的 cookie 时，第二个浏览器对话框将会弹出一个免费购物

信息提示。逻辑猜测这个 cookie 的目的是，追踪你是否看过这个广告信息。

当所有的 cookie 都被设置时，Ns cookie.txt 文本内容如下：



```
cookies.txt - Notepad
File Edit Format Help
# Netscape HTTP Cookie File
# http://www.netscape.com/newsref/std/cookie_spec.html
# This is a generated file! Do not edit.
. amazon.com TRUE / FALSE 994320128 session-id 102-7224116-8052958
. amazon.com TRUE / FALSE 994320181 session-id-time 994320000
. amazon.com TRUE / FALSE 2082787330 ubid-main 077-4356846-2652328
. amazon.com TRUE / FALSE 2082787787 x-main hQFiIXHUFj8mCScT@Yb5Z7xsvs0FQjBf
www.amazon.com FALSE / FALSE 993797034 seenpop 1
```

为什么这里只有五个 cookies 在文本中？因为 per-session cookies 只保存内存中，并没有写入 cookie.txt 文件中。

### Cookie 内容和结构分析

在我们分析所有 amazon.com 设置的 cookie 时，让我们快速查看下 cookie 的结构和每个字段的意思。

第一个 cookie 是：

. amazon.com TRUE / FALSE 994320128 session-id 102-7224116-8052958

使用 [www.cookiecentral.com](http://www.cookiecentral.com) 的信息，将 cookie 分解成不同的字段：

● . amazon.com 用于验证域名的 cookie，只是机器设置的 cookie，让 amazon.com 域名能够读到这个 cookie。（然而，这也是个 bug，web 浏览器对 cookie 执行后将允许未授权的网页通过先前的 cookie 访问）

● TRUE 是一个标记表明是否所有在域名内的机器能够访问这个 cookie

● / 是 cookie 路径验证

● FALSE 是一个安全标记表明是否有一个安全连接需要访问这个 cookies

● 994320128 是 UNIX 的 cookie 过期时间，UNIX 时间是从 1970 年 1 月 1 日 00:00:00 格林尼治标准时间的秒数)

● session-id 是被 cookie 储存的变量名

● 102-7224116-8052958 是变量值

### Amazon.com 网站 cookie 分析

访问 amazon.com 主页产生了六个 cookies----一个 per-session cookie 和五个持久 cookies，由于网站设计文档和开发人员都不能告诉我们关于 cookie 的意思，所以我们将放入表格，以便于破解 cookie 数据意义。我们将只考虑前面五个 cookies，以此我们可以判断第六个 cookie 的目的

cookie #	domain	accessible by all machines	path	secure connection needed	expiration	name	value
1	.amazon.com	TRUE	/	FALSE	994320128	session-id	102-7224116-8052958
2	.amazon.com	TRUE	/	FALSE	994320181	session-id-time	994320000
3	.amazon.com	TRUE	/	FALSE	2082787330	ubid-main	077-4356846-2652328
4	.amazon.com	TRUE	/	FALSE	(per-session cookie)	obidos_path	(see above)
5	.amazon.com	TRUE	/	FALSE	2082787787	x-main	hQFiIxHUFj8mCscT@Yb5Z7xsVsOFQjBf

第一个 cookie 是一个 session ID 分配给我的订单 session。这里主要内容是变量名“session-id”。另一个关键是这个变量值：102-7224116-8052958,能够在主页的 URL 末尾看到，在第五个 cookie 设置后可以在主页的 URL 的末尾看到，[www.amazon.com/.../home.html/102-7224116-8052958](http://www.amazon.com/.../home.html/102-7224116-8052958)

cookie 1 有效期为 7/5/2001，在 cookie 设置之前我已经通过 NS 的警告对话框看到了，所以 cookie 中对于 UNIX 过期时间 994320125 等于 7/5/2001 (cookies 过期时间=1970+994320125/(365\*24\*60\*60))

第二个 cookie 的目的并不明显，基于名字和值分别为 session-id-time 和 99432000，我猜测可能是我的本机 Unix 访问 amazon.com 网站的 session 最大结束时间。因为从 NS 的提示框知道 cookie 将于 7/5/2001 过期，所以我推测 994320181 的过期值等于 7/5/2001。为什么？这里只有两个 UNIX 值 994320181 - 994320000 = 181 = ~ 3 分钟左右。

第 3 和 5 的 cookie 的目的是很难推测。Cookie3 和 5 的名字是 ubid-main 和 x-main,并不能让我们立即理解。两个 cookie 的过期时间都是 2036，所有 amazon 对其必定长久使用。

第四个 cookie，只是一个 per-session/没有保存 cookie，包含一个子字符串“continue-shopping-url”的长值。我猜测是如果我从购物车页面点击“Continue Shopping”按钮，这个 cookie 的值就告诉 amazon 服务器在哪里送达给我。在这之前，我花了很长时间弄清楚某些 cookie 的使用是什么。我也和 amazon 的开发人员进行沟通或者通过一些设计文档和功能需求来更进一步得到详细的解答。

## Cookie 测试

现在我们知道什么是 cookie，在 web 系统中他们是怎样来提供状态，以及 cookie 的内容分析，让我们来演示怎样通过使用 cookie 来测试网页

### 禁止 cookie

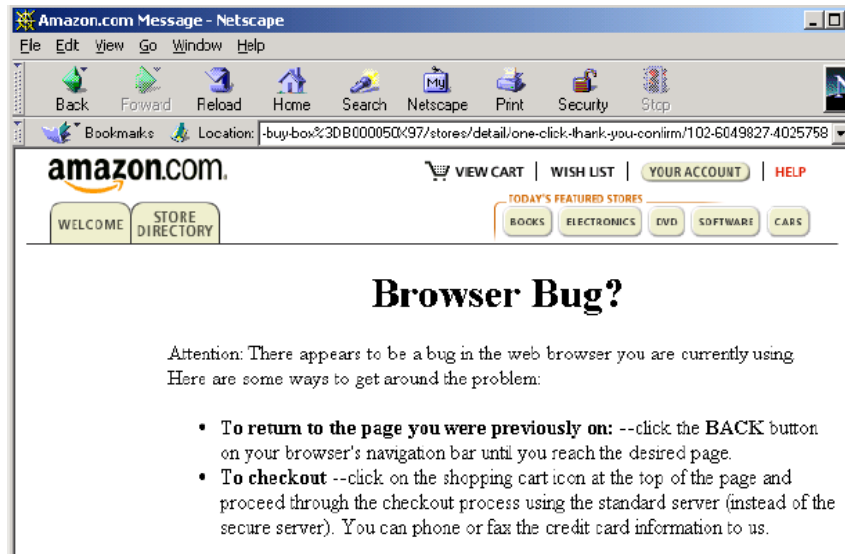
可能这是 cookie 测试中最容易的一部分了。如果所有的 cookie 都禁止了，那么网页会发生什么？首先关闭所有的浏览器窗口，删除 PC 机中的所有 cookies。当 cookie 运行时，浏览器是打开的，cookie 文件就会存在。所以你必须关闭浏览器删除 cookies，关闭浏览器也就移除了内存中 per-session cookie。

禁止所有 cookies，并且使用网页的主要特性和功能。大多数时间，你都将发现他们不能工作，因为 cookie 已经禁止了。这并不是 bug，但是是无法更改的事实：在网页上禁止 cookie 就使要求使用 cookie 的网页功能也禁止。

随着 cookie 的禁止，你的测试工作也会变少。网站的用户是否很明显必须有 cookie 才能使用网站呢？服务器试图设置一个 cookie 也会失败吗？如果是，那么送达网页一个用户状态，cookie 必须能使网站上工作吗？或者，企图在运行多次相同的操作后用户会非常沮丧为什么网站不能工作了？

Amazon.com 这个已经成功通过测试。我能够使用网站所有的功能----搜索，购物车，检验功能---即使 cookie 完全禁止。我敢肯定状态维持是服务器那边所关心的，其维持是基于主页 URL 末尾的 session-ID。让我们测试这个假设。主页 URL 是 [www.amazon.com/.../home.html/104-0274809-0482344](http://www.amazon.com/.../home.html/104-0274809-0482344) 如果改变最右边的数字从 4 到 5，重新请求 URL，Amazon 会丢弃我编辑的 URL，通过创建一个带有新的 session-ID 的 URL，从中断的情况下恢复 URL 为 [www.amazon.com/.../home.html/107-0357560-1728507](http://www.amazon.com/.../home.html/107-0357560-1728507)。所以，出现了假设的正确性。

让我们再进一步，我在 amazon 的主页选择 Yamaha CD-ROM kit，并且增加到我的购物车。购物车页面的 URL 是 [www.amazon.com/.../one-click-thank-you-confirm/107-0357560-1728507](http://www.amazon.com/.../one-click-thank-you-confirm/107-0357560-1728507)。改变最右边的数字从 7 到 8，然后回车编辑后的 URL，丢失了我的购物车，并且返回了一个错误页面，这进一步支持了先前的假设服务器端状态是通过 URL 中一个 session-id 来维持的。



即使用户禁止了 cookie，那么服务器端的状态维持仍然允许在 amazon.com 购物，这是一个巧妙的设计。如果 cookie 可用，我们可以看到先前 amazon 设置的 session ID cookie 会认识你的 session ID。为什么？如果你离开一个非空购物车的页面，然后返回，session ID 的 cookie 可以被用于重新获取你先前的订购 session 和购物车状态。

#### 有选择性的拒绝部分 cookies

如果一些 cookie 被接受，而一些 cookie 被拒绝，那么网站会发生什么了？首先删除你 PC 机上的所有 cookie，包括你测试的网站和浏览器 cookie 选项提示。然后进行网站的主要功能操作。当网站每次设置 cookie 时，你将会被提示。那么接受一些，拒绝一些。（首先分析网站对于 cookie 的使用，起草一个关于接受/拒绝 cookie 的测试计划）在部分 cookie 被拒绝后，网站是怎样停顿的？基于上面所述，web 服务器会检测某些被拒绝的 cookie，会做出适当的提示吗？或者，网站会出现故障，事故，损坏数据等其他不正当的方式吗？

让我们来制定关于部分 cookie 被拒绝的测试策略来测试 amazon.com 主页。每个测试用例都将要求六个 cookie 既有接受也有被拒绝的情况，所以  $2^6=64$  种测试用例。测试用例的一个在子集列举如下表。



Test case #	Cookie 1 (persistent)	Cookie2 (persistent)	Cookie 3 (persistent)	Cookie 4 (per-session)	Cookie 5 (persistent)	Cookie 6 (persistent)
1	Reject	Reject	Reject	Reject	Reject	Reject
2	Reject	Reject	Reject	Reject	Reject	Accept
3	Reject	Reject	Reject	Reject	Accept	Reject
4	Reject	Reject	Reject	Reject	Accept	Accept
5	Reject	Reject	Reject	Accept	Accept	Accept
.....						
64	Accept	Accept	Accept	Accept	Accept	Accept

例如，如果我们执行第五个测试用例，当访问 [amazon.com](http://amazon.com) 时，我拒绝了前面三个 cookies，但是接受了第四，第五，第六个 cookies。

第一个测试用例等于是之前执行的禁止 cookie 测试，但是我仍然保留了这个测试用例在表中。如果把接受和拒绝用二进制的形式表示为 0 和 1，这个表实现了包括 0---63 在内的十进制到二进制的转换。

基于在禁止 cookie 测试对于 [amazon.com](http://amazon.com) 的执行，我猜想网页将通过更多甚至所有的部分被拒绝 cookie 测试用例。我执行第 2 和第五个测试用例，关闭浏览器，并删除了所有的 cookies。执行完后，我都能够使用网站的主要功能，并且，没有任何问题。看上去网站的设计者已经考虑的 cookie 的问题，使得顾客能够在 [amazon.com](http://amazon.com) 上购物不受任何影响。

注意的是以上的测试用例只处理了 [amazon.com](http://amazon.com) 第一次被创建时的接受或拒绝 cookie 的测试用例。我们应该也测试接受和拒绝 cookies 的修改，以上的 cookie 只是个初始化的设置。如果当服务器企图频繁修改 cookie，如果你不接受这个改变，维持老的 cookie 的值，那么会发生什么了？

### 破坏 cookies

现在我们就有机会在测试时来真正滥用网站了！你首先需要来做的是“检测 cookie”来决定怎样和在哪里使用网站的 cookies，并且了解 cookie 数据的意思。现在，执行网站的主要功能，通过这种方式，来创建和修改 cookies，试着这样做：

● 改变持久 cookie 数据。（由于 per-session cookie 只被存储在内存，他们不能被编辑。不过可以通过工具来破坏这些 per-session cookies，但是我开始并没有意识到。）例如：在第一个 cookie 被写时，改变 session-id 的变量名，比如改成 ses-id 或者 sexqion-id。记住，你必须关闭浏览器来编辑这些 cookies。在编辑之前，如果我访问了 [amazon](http://amazon.com) 网站，关闭浏览器，重启浏览器然后返回 [amazon.com](http://amazon.com)，

将返回我之前的 session 维持的基于 cookie 中的 session ID。但是，如果破坏 session-ID 的变量名，Amazon 会检测破坏的变量名，和复原被丢弃的六个 cookie，并重新创建新的变量。在编辑 cookie 之后，重新启动浏览器，加载/继续使用该网站。那么破坏的 cookie 会引起网站故障吗？会引起数据丢失或数据库故障吗？

第二个例子：改变 session-id 的数据值，在最右边的数字中加 1，102-7224116-8052958 变成 102-7224116-8052959。你现在是不是看到了其他人的购物 session 了？是否有任何丢失或数据库故障了？

● 删除部分 cookies。在该网站上执行更多的操作，然后删除这些 cookie。继续使用该网站。会发生什么了？它容易修复吗？会有数据丢失或数据库故障吗？

### Cookie 加密

最后一个 cookie 测试是很简单的。当在你测试网站上调用 cookie 使用时，请特别注意下 cookie 数据的意思。一些敏感信息比如用户名和密码都不应该储存在文本中，以至于让其他人读到；在送到你的 PC 机时，这些数据应该被加密。我已经测试了许多网站，这个很明显的规则都别忽略了。这个用例能够确定某些敏感数据类型---例如信用卡号，即使加密了，也不应该储存在 cookie 中。

基于上面对 amazon.com 的 cookie 分析，我想说 amazon 已经很容易就通过了 cookie 加密测试。没有敏感的用户名或信用卡信息存储在文本中。我发现这也是黑客通过 cookie 数据进入账户的途径之一。在用户 A 的机器上，我打开了 amazon 主页，并且订购一本 John Adams 的书到我的购物车，我从 cookie.txt 文件中复制了五个持久 cookie，粘贴到了用户 B 的机器上。然后在用户 B 的机器上打开 amazon 主页，然后用户 B 的机器上允许我操作用户 A 的刚才订购的 John Adam 书的购物车。

这个小实验进一步表明，只需要编辑用户 B 与用户 A 的匹配的 session-id，就能不允许用户 B 来访问用户 A 的购物车。这也让我相信了一个或多个被用于他人 cookie 作为一个独特的密匙来减少这类黑客取得成功的概率。

### 结论

通过使用 cookie 可以维持 web 系统中的状态信息（其他维持状态的方法还包括隐藏表单字段和在 HTML 链接中嵌入状态数据，我希望 web 测试工程师也能开拓这些方法）。作为测试人员，我们的工作是通过与开发人员沟通，读系统文档或执行网站使用上述技术找到 cookie，并以此来设计测试用例。



## QTP 字典对象的强化

作者：Founding 测试工作室

在储存和检索信息的时候，`Scripting.Dictionary` 对象被我们广泛的应用。我们已经做了一系列的例子来阐述它的基本用法。比如，用作全局变量的保存字典，或者用来保存各种方法的参数。

然而，尽管 `Scripting.Dictionary` 功能十分强大，它还是存在一些不可避免的缺陷。我们无法根据一个 `Index` 来获取他的对应值。(特别是在使用循环的时候，这点尤为明显。)又比如，当我们试图添加一个已经存在的项，它就会抛出一个异常。它的弱点还体现在合并，输入和输出方面。

今天，本文将一步一步的介绍，怎样来建立一个新的字典对象。从而使它能够更健壮和更灵活的被我们所使用。在阅读之前，你需要回顾一下 [VBScript class](#) 的相关知识。

在此强调一下，我们要做的只是对原来的字典对象进行重构。我们需要一个内置的普通 `Scripting.Dictionary` 对象，在此基础上给它添加一些更灵活的功能。这种重构的技术在编码方面是很普通的，它使得我们可以对原来的对象进行自主的扩展。就好像，我们是站在巨人的肩膀上工作，站的高看的远。

我们所做的重构，一般来说不会破坏原来对象的功能。就是说，`Scripting.Dictionary` 原来所包含的所有代码，都会被完整的重现在我们构建的新对象之中。我们保留原先的所有属性和方法，并且拓展我们自己的方法。

现在就让我们开始吧。

### 原始接口：

首先，我们要定义我们的初始类。建立一个 `private` 类型的原始字典对象。下面是需要添加的构造函数和析构函数。

```
Sub Class_Initialize 'Will fire automatically when create a new instance
    Set oDic = CreateObject("Scripting.Dictionary")
End Sub

Sub Class_Terminate 'Will fire automatically when an instance is destroyed
    Set oDic = Nothing
End Sub
```

接下来，我们要构造一些简单的 `public` 接口，让我们可以像传统的字典对象一样使用新对象。举例来说，我们需要为新对象添加一个 `.Count property` 接口，而且原始的字典对象已经存在了这个接口，那么我们就可以直接套用内置字典对

象的.Count property 接口。

```
'All these properties are read-only
This is why they only contain a Get block

Public Property Get Count 'Returns the number of items in our dictionary
    Count = oDic.Count 'The answer is the number of items in the inner dictionary
End Property

Public Property Get Keys 'Returns the dictionary keys (array)
    Keys = oDic.Keys
End Property

Public Property Get Items 'Returns the dictionary items (array)
    Items = oDic.Items
End Property

Public Property Get Exists(Key) 'Returns a True/False if the Key exists
    Exists = oDic.Exists(Key)
End Property
```

正如你所看到的，添加这些接口不需要额外的编码和操作处理，相对来说直接而简单。

那么，接下来的工作，看上去就不是那么一成不变的了。

### 修改后的接口：

那么我们就从.Remove 方法开始。在原始的字典对象中，当我们试图删除一个不存在的项的时候，会抛出一个异常。那么我们就需要做个简单的补丁，我们在删除之前做一个判断，只删除那些已经存在的项。

```
Public Sub Remove(Key)
    If oDic.Exists(Key) Then oDic.Remove(Key)
End Sub
```

让我们继续来看.Add 这个方法，它的作用是在字典对象中添加一条记录。原始的字典对象存在一个很烦人的特点：当我们添加的项与原来已经存在的项，二者关键字存在冲突的时候，它会抛出一个错误。我们希望这个方法变成简单的覆盖当前关键字下的原始记录。那么在我们的新方法中，我们就这样写：

```
Public Sub Add(Key, Value)
    If oDic.Exists(Key) Then
        'We cannot simply add the item, we have to replace it
        oDic.Item(Key) = Value
    Else
        'Here we can add the key as usual
        oDic.Add Key, Value
    End If
End Sub
```

但是稍等一下，读者有没有发现存在什么问题呢？事情当然没有想象的那么简单。数据字典的项里面保存内容，并不仅仅是一个简单的变量，有时候是某个对象。（当然也可以是字典对象本身）但是在 VBScript 中，给一个对象赋值，需要用到 SET 关键字。如果我们试图添加的 VALUE 是某个对象，那么语法就要求使用 Set oDic.Item(Key) = Value，而不是简单的 oDic.Item(Key) = Value。显然，我们之前的写法是很容易出错的。那么我们做一个简单的改变。利用我们之前改写的.Remove 方法。既然是覆盖原先内容，那么我们先删除原先的记录，再添加一条新的。

```
Public Sub Add(Key, Value)
    Call Me.Remove(Key)
    oDic.Add Key, Value
End Sub
```

让我们多花点时间来看一下在这里我们是怎么做的，在我们的字典对象里，我们要删除一个已经存在的 key 的项，但是我们刚刚创建的.Remove 方法只是起到的删除的功能，所以我们要继续修改我们的脚本程序，我们需要在使用刚才的代码的同时调用我们的 new 和完善我们的 Remove 方法。在这里我们需要特别注意的，我们需要创建一个新的 key 和值。

接下来，重点介绍一个方法：.Item。稍许复杂的是，我们需要区分，传进来的参数是真正的“key”还是 key 的索引值。同样的，我们也需要区分它返回的 ITEM 是一个普通值还是一个对象。那么我们首先来解决关于 key 的问题：

```
Public Function Item(Key)
    Dim arrKeys 'Will hold the inner dictionary keys
    Dim sRealKey

    arrKeys = oDic.Keys

    If IsNumeric(Key) Then
        sRealKey = arrKeys(Key) 'We have to translate the number to the corresponding
key
    Else
        sRealKey = Key 'We can use the key as it is
    End If

    Item = oDic.Item(sRealKey)
End Function
```

有没有注意到？我们忽视了一个可能存在的异常情况。如果我们的有个项的 **KEY** 值是“1”，那我们的代码就可能错误的认为，这是个索引值。或许我们可以强化一下代码，来规避这个错误。其实不如用种更简单的方法，我们直接规定，不允许建立这样的 **KEY** 值。

接下来我们要解决的第二个问题是，我们已经获取了 **KEY**，但存在返回值类型的不同。同样的问题在之前写 **Add** 方法的时候就出现过。当返回一个对象的 **ITEM** 值时，在语法上需要

用 **SET** 关键字。这次，我们不能像 **Add** 方法那么简单的回避，我们不得不正视这个问题。修改如下：

```
Public Function Item(Key) 'Returns an item, either by a key or an index
    Dim arrKeys 'Will hold the inner dictionary keys
    Dim sRealKey 'The actual key which holds the needed value

    arrKeys = oDic.Keys

    If IsNumeric(Key) Then
        sRealKey = arrKeys(Key) 'We have to translate the number to the corresponding
key
    Else
        sRealKey = Key 'We can use the key as it is
    End If

    'If the relevant item is an object, we'll have to use the Set keyword to return it
    If IsObject(oDic.Item(sRealKey)) Then
        Set Item = oDic.Item(sRealKey)
    Else
        Item = oDic.Item(sRealKey)
    End If

End Function
```

### 新的接口：

最后一个原始接口.Item，已经被我们改写完成了。接下来我们就来创建一些新的接口，从而使得新的字典对象功能更加强大。

我们第一个要添加的新接口就是.Key。它将根据索引值，返回 KEY 的真实值。虽然简单，但是很实用。

```
Public Function Key(iIndex)
    Dim arrKeys

    If iIndex > Me.Count -1 Then Exit Function 'There is no such key

    arrKeys = Me.Keys
    Key = arrKeys(iIndex)
End Function
```

接下来，我们添加另一个有意思的新接口.Clone。看名字就可以猜到，它的作用是复制一个当前的字典对象。使得字典对象可以更容易的被其他方法所引用。

```
Public Function Clone
    Dim oResult
    Dim i

    Set oResult = New NewDictionary

    For i = 0 to Me.Count - 1
        oResult.Add Me.Key(i), Me.Item(i)
    Next

    Set Clone = oResult
End Function
```

现在，要介绍一个革命性的方法：合并。我们需要建立另一个新的字典对象，然后把它的数​​据合并到我们的数据对象里。为了避免重复，冲突的问题，我们设定本地的字典对象的内容拥有更高的优先级，我们只添加那些新的项。当我们已经实现了新的.Item 和.Key 方法以后，要实现合并的功能就变得非常简单了。

```
Public Sub Merge(oOutsideDictionary)
    Dim i

    For i = 0 to oOutsideDictionary.Count - 1
        If Not Me.Exists(oOutsideDictionary.Key(i)) Then _
            Me.Add oOutsideDictionary.Key(i), oOutsideDictionary.Item(i)
    Next
End Sub
```

最后，我们要添加.Import 和.Export 方法，来实现字典内容的导入和导出功能。通常我们使用字典对象来记录一些测试中的重要信息，我们希望这些信息能够被保存以便继续使用。当 QTP 脚本运行完毕，所有对象都将被释放。因此，我们需要将有用的信息保存到外部文件中。例如，TXT 文件。我们定义保存的格式为：Key>Value|Key>Value|...。当然，这样只能保存简单的数据类型。对象类型，是无法保存的。

```
Sub Export(sFileName)
    Dim i
    Dim oFSO
    Dim oFile
    Dim sData

    On Error Resume Next 'Protects from object items

    For i = 0 to Me.Count - 1
        sData = sData & "|" & Me.Key(i) & ">" & Me.Item(i)
    Next

    sData = Mid(sData,2) 'Get rid of the first, unneeded '|'

    Set oFSO = CreateObject("Scripting.FileSystemObject")
    Set oFile = oFSO.CreateTextFile(sFileName, True)

    Call oFile.Write(sData)

    oFile.Close

    Set oFile = Nothing
    Set oFSO = Nothing

    On Error Goto 0
End Sub
```

```
Sub Import(sFileName)
    Dim i
    Dim oFSO
    Dim oFile
    Dim sData
    Dim arrData, arrSingleField

    On Error Resume Next 'Protects from wrong file names

    Set oFSO = CreateObject("Scripting.FileSystemObject")
    Set oFile = oFSO.OpenTextFile(sFileName, 1, True)

    sData = oFile.ReadAll

    oFile.Close

    Set oFile = Nothing
    Set oFSO = Nothing

    arrData = Split(sData, "|")

    For i = 0 to UBound(arrData)
        arrSingleField = Split(arrData(i), ">")
        Me.Add arrSingleField(0), arrSingleField(1)
    Next

    On Error Goto 0
End Sub
```

我们还可以添加更多的方法和属性，以便于更灵活的操作字典对象。至此，我们的字典对象已经耳目一新了。

最后，我们将所有代码整合如下。



Class NewDictionary

Private oDic 'Will hold the hidden inner dictionary

\*\*\*\*\* Class LifeCycle \*\*\*\*\*

Sub Class\_Initialize 'Will fire automatically when create a new instance

Set oDic = CreateObject("Scripting.Dictionary")

End Sub

Sub Class\_Terminate 'Will fire automatically when an instance is destroyed

Set oDic = Nothing

End Sub

\*\*\*\*\* Basic properties \*\*\*\*\*

'All these properties are read-only

'This is why they only contain a Get block

Public Property Get Count 'Returns the number of items in our dictionary

Count = oDic.Count 'The answer is the number of items in the inner dictionary

End Property

Public Property Get Keys 'Returns the dictionary keys (array)

Keys = oDic.Keys

End Property

Public Property Get Items 'Returns the dictionary items (array)

Items = oDic.Items

End Property

Public Property Get Exists(Key) 'Returns a True/False if the Key exists

Exists = oDic.Exists(Key)

End Property

\*\*\*\*\* Improved Methods \*\*\*\*\*

**Public Sub** Remove(Key) 'Removes the key from the dictionary (it it existed)

**If** oDic.Exists(Key) **Then** oDic.Remove(Key)

**End Sub**

**Public Sub** Add(Key, Value) 'Adds a new value to the dictionary. Overwrites existing values

**Call** Me.Remove(Key)

oDic.Add Key, Value

**End Sub**

**Public Function** Item(Key) 'Returns an item, either by a key or an index

**Dim** arrKeys 'Will hold the inner dictionary keys

**Dim** sRealKey 'The actual key which holds the needed value

arrKeys = oDic.Keys

**If** IsNumeric(Key) **Then**

sRealKey = arrKeys(Key) 'We have to translate the number to the corresponding key

**Else**

sRealKey = Key 'We can use the key as it is

**End If**

'If the relevant item is an object, we'll have to use the Set keyword to return it

**If** IsObject(oDic.Item(sRealKey)) **Then**

**Set** Item = oDic.Item(sRealKey)

**Else**

Item = oDic.Item(sRealKey)

**End If**

**End Function**

```
***** New Methods *****
```

```
Public Function Key(iIndex) 'Returns the key at a given index
```

```
    Dim arrKeys
```

```
    If iIndex > Me.Count -1 Then Exit Function 'There is no such key
```

```
    arrKeys = Me.Keys
```

```
    Key = arrKeys(iIndex)
```

```
End Function
```

```
Public Function Clone 'Returns a copy of the dictionary
```

```
    Dim i
```

```
    Dim oResult
```

```
    Set oResult = New NewDictionary
```

```
    For i = 0 to Me.Keys - 1
```

```
        oResult.Add Me.Key(i), Me.Item(i)
```

```
    Next
```

```
    Set Clone = oResult
```

```
End Function
```

```
Public Sub Merge(oOutsideDictionary) 'Merges the dictionary with another one
```

```
    Dim i
```

```
    For i = 0 to oOutsideDictionary.Count - 1
```

```
        'Add the value, but don't overwrite
```

```
        If Not Me.Exists(oOutsideDictionary.Key(i)) Then _
```

```
            Me.Add oOutsideDictionary.Key(i), oOutsideDictionary.Item(i)
```

```
    Next
```

```
End Sub
```

```
Sub Export(sFileName) 'Exports the dictionary to a text file
```

```
    Dim i
```

```
    Dim oFSO
```

```
    Dim oFile
```

```
    Dim sData
```

```
On Error Resume Next 'Protects from object items
```

```
'First, create a string holding the dictionary data
```

```
For i = 0 to Me.Count - 1
```

```
    sData = sData & "|" & Me.Key(i) & ">" & Me.Item(i)
```

```
Next
```

```
sData = Mid(sData,2) 'Get rid of the first, unneeded '|'
```

```
'Now, write the string to an external file
```

```
Set oFSO = CreateObject("Scripting.FileSystemObject")
```

```
Set oFile = oFSO.CreateTextFile(sFileName, True)
```

```
Call oFile.Write(sData)
```

```
oFile.Close
```

```
Set oFile = Nothing
```

```
Set oFSO = Nothing
```

```
On Error Goto 0
```

```
End Sub
```

```
Sub Import(sFileName) 'Builds the dictionary from an external file
    Dim i
    Dim oFSO
    Dim oFile
    Dim sData
    Dim arrData, arrSingleField

    On Error Resume Next 'Protects from wrong file names

    Set oFSO = CreateObject("Scripting.FileSystemObject")
    Set oFile = oFSO.OpenTextFile(sFileName, 1, True)

    sData = oFile.ReadAll

    oFile.Close

    Set oFile = Nothing
    Set oFSO = Nothing

    'Split the data string to its separate key>item pairs
    arrData = Split(sData, "|")

    For i = 0 to uBound(arrData)
        'Split each pair
        arrSingleField = Split(arrData(i), ">")

        'Add the value to the dictionary
        Me.Add arrSingleField(0), arrSingleField(1)
    Next

    On Error Goto 0

End Sub

End Class
```

## 全客户端操作 TestDirector 项目数据移植

作者：杭聪

**摘要：**TestDirector（简称 TD）是业界广泛被使用的 B/S 架构测试管理工具，其中的缺陷管理模块是软件开发团队使用频率最高，用于管理软件项目的缺陷。当测试服务器发生变更时，往往需要将 TestDirector 中的测试项目数据从一台服务器移植到另一台服务器。传统的做法往往涉及很复杂的 TD 相关数据文件和后台数据库操作，本文将介绍和推广一种单纯在 TD 客户端就能搞定的简单、便捷的移植方法。

**关键词：**缺陷管理、TestDirector、移植

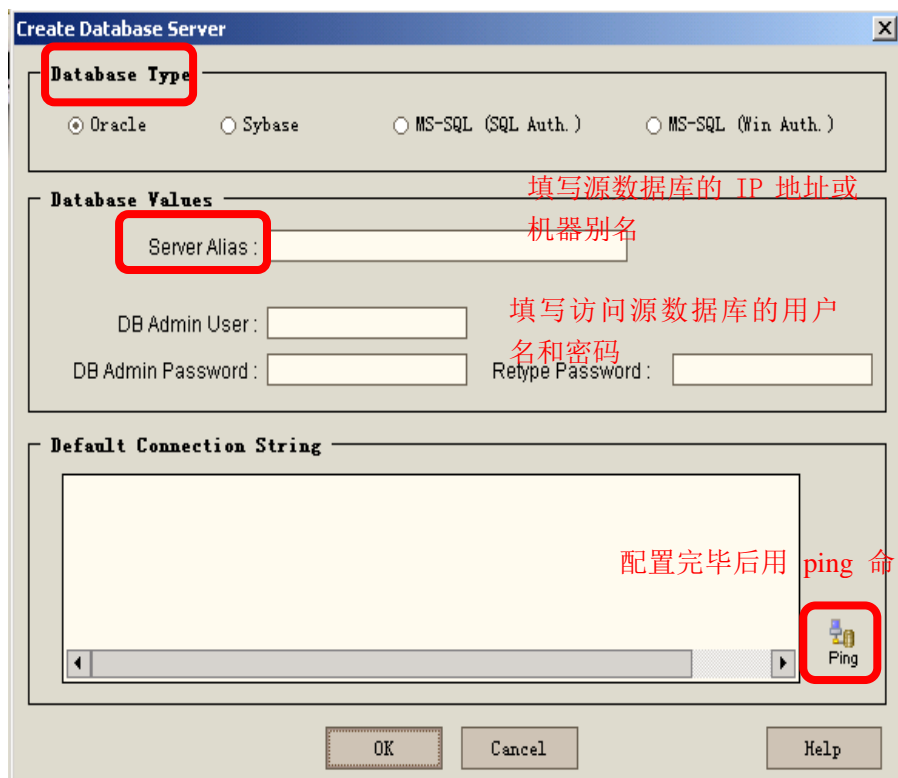
TestDirector 可以在同类型或不同类型的数据库之间完成项目数据移植。移植操作完全可在客户端界面进行,无须对后台数据库进行任何操作。但目前很多测试同行对 TD 数据移植存在些误解,认为唯一的移植方案是在后台数据库进行表间复制等繁琐的操作,比如流传比较广的《Test Director 8.0 项目数据库维护和移植》,就用很长的篇幅描述了繁琐复杂的数据移植操作。

下面以同类数据库间的 TD 项目数据移植为例,异类数据库间的数据移植的步骤也类同。在众多 DBMS 中,Sql server 常被选作主流的 TD 数据库,本文也以 Sql Server 为例进行操作说明。假设主机 A 上已安装 TD 服务端（数据库可能在 A 机也可能在另外的数据库服务器上），要把该 TD 库的项目数据移植到主机 B 上（主机 B 上已安装好 TD 服务端），操作步骤如下：

### 1.登录 B 机上的 Site Administrator，选择 DB Servers 标签

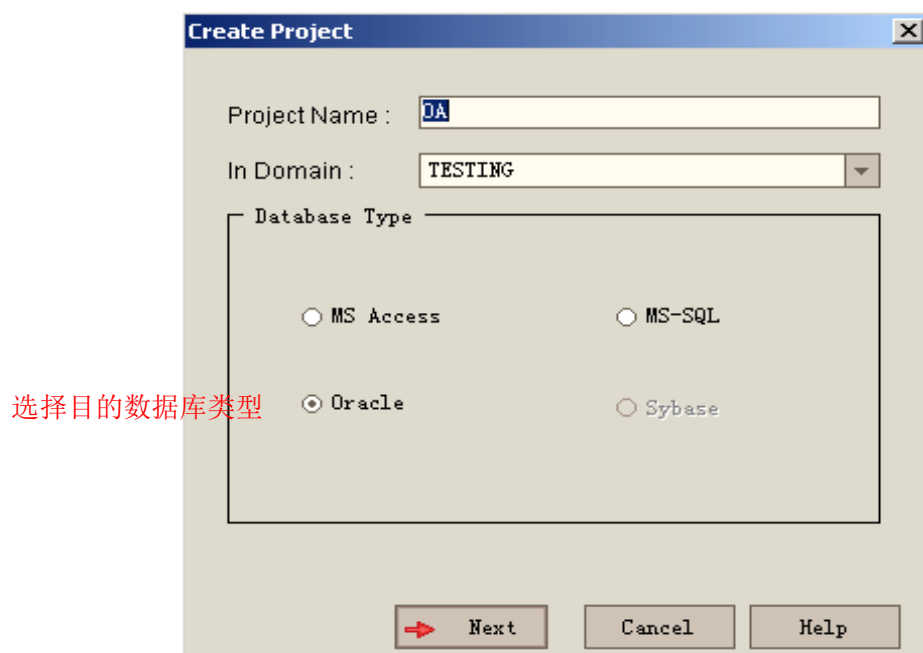
新建一个 DB server，并填写源数据库的信息（即被移植的项目数据所在的数据库），如图所示：

### 选择后台数据库的类

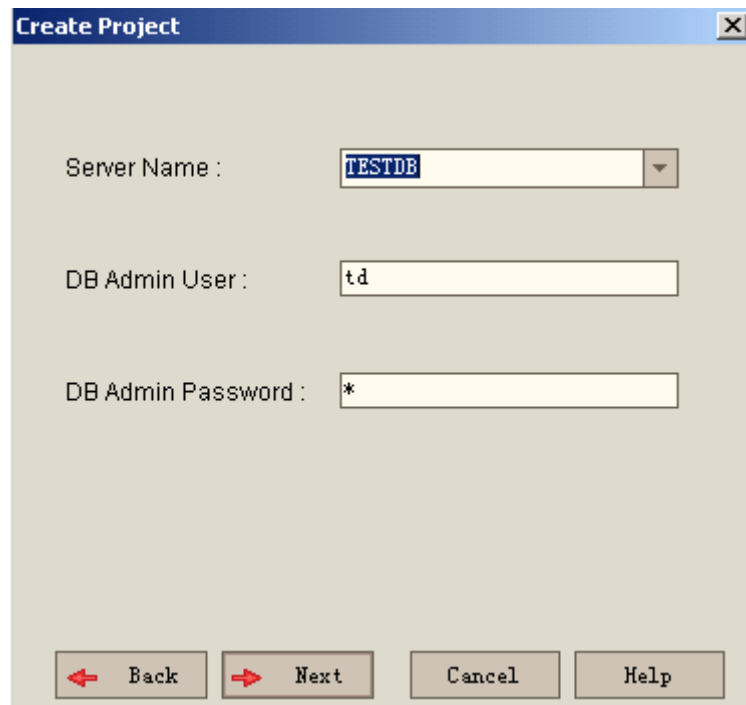


2.选择 Project 标签->Restore Project，打开源主机的 TD 共享目录 TD\_dir 下即将要移植的测试项目文件夹，选择 DBid.ini，恢复源项目至目的主机上，但所恢复的测试项目所在还是指向源主机的数据库。（注：如果不执行第 1 步，直接执行第 2 步，TD 就会提示找不到 DBid.ini 文件中的 DB Server）

3.选择 Project 标签，新建 domain 和 project，输入项目名称和所属的域名，以及数据库类型，如图所示：



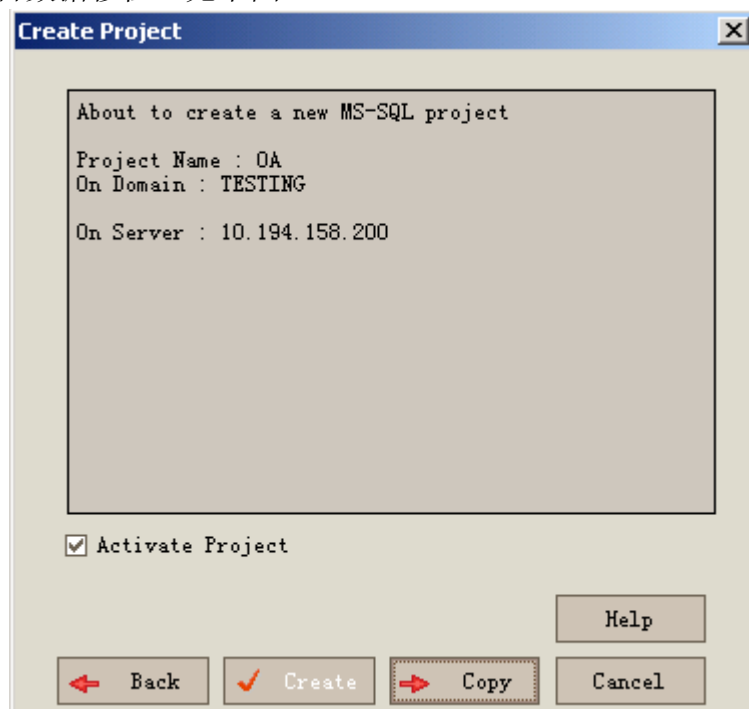
选择源数据库服务器名，并填写用户名 / 密码



The 'Create Project' dialog box contains the following fields and buttons:

- Server Name:** A dropdown menu with 'TESTDB' selected.
- DB Admin User:** A text input field containing 'td'.
- DB Admin Password:** A text input field containing a single asterisk (\*).
- Buttons:** 'Back' (with a left arrow), 'Next' (with a right arrow), 'Cancel', and 'Help'.

在随后出现的对话框里选择“copy”，这个是移植的关键，通过项目复制功能完成项目数据移植（见下图）。



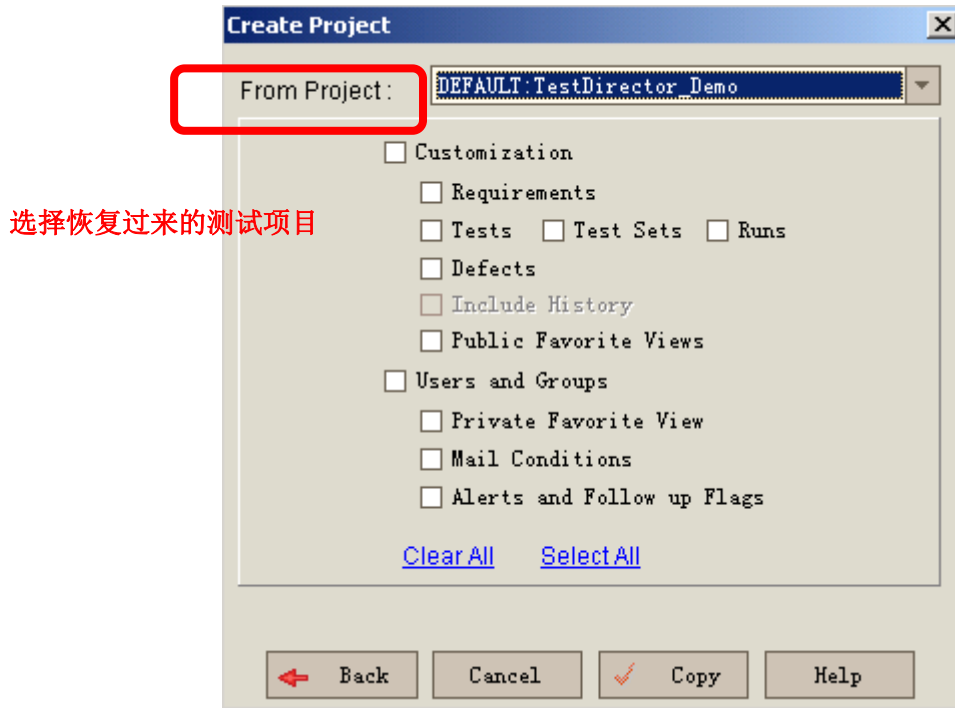
The 'Create Project' dialog box shows the 'Copy' step with the following details:

- Text Area:**

```
About to create a new MS-SQL project
Project Name : OA
On Domain : TESTING
On Server : 10.194.158.200
```
- Checkbox:** ☒ Activate Project
- Buttons:** 'Back' (with a left arrow), 'Create' (with a checkmark), 'Copy' (with a right arrow and a dashed border), 'Cancel', and 'Help'.

在“copy”对话框里选择复制源（从源主机恢复的测试项目）并配置复制项，继而执行 copy 操作。





经过以上步骤后，A 机的项目数据便移植到 B 机的 TD 数据库里，移植完后可将临时建立的连接到 A 机的 DB server 与恢复项目删除。用这种方法移植的项目数据，不用担心项目数据的遗漏或是 TD 发生意想不到的错误，比如进入到缺陷管理页面，所有缺陷记录的 ID 号缺失，新增缺陷时 TD 不能自动生成 defect ID 从而导致无法保存。

## 了解软件性能测试

作者：Dale Perry 译者：成韩丽

这是该系列四部分中的第一部分，第二部分将会在 [StickyMinds.com](http://StickyMinds.com) 中发表。

大多数和软件相关的性能测试项目都会失败，当我谈到失败的时候，并不是说测试异常结束了，我所谈论的是没有给测试的人员提供一点有用资料的情况。

失败的产生有很多的原因。在本系列的文章中，我将会描述基本的问题和如何避免它们，所以如果你正在执行一个软件性能测试的任务，你将会理解一系列的问题，并理解哪些是你能够完成的和哪些是你不能完成的。这些将不能解决所有的问题，也不是提供给软件性能测试专家测试，他们中的大多数人可能已经通过反复试验了解到这些想法。

该系列中的很多想法都是基于 Ross Collard of Collard & Associates 的一个课程开发的，我在那里教 SQE 培训。在这里，我无法因为这些概念和想法而得到好评。

下面的将是我们通过这一系列的文章将会研究的大体范围：

### 第一部分

- 测试人员在过程中的角色
- 确定性能和业务目标
- 如何让性能测试和开发过程相适应

### 第二部分

- 系统架构和基础构件
- 选择准备运行的测试类别

### 第三部分

- 了解工作量（业务简介）
- 量化和定义度量单位

### 第四部分

- 了解工具
- 执行测试和报告结果

这些问题在很多不用的领域得到不同序列的解决，我将会按照自己认为最适合的方法进行。这个方法已适用于所有类型的平台：大型机、客户机/服务器、嵌入式、Web 网络等。其中第一部分说明了测试者所扮演的角色，了解性能目标以及何种性能测试是适用于开发进程的。

### 测试者所扮演的角色

在软件性能测试中，性能测试者的角色更多的时候更像是一个顾问——你指导、指挥并协助技术人员识别和纠正问题。性能测试者测试：他们没有按照基调、不调试或者不纠正识别的问题。一个好的性能测试是一个团队努力的结果，所有主要角色和利益相关者都必须是性能测试计划进程中的一部分，其利益相关者包括可能去调整、纠正、调试或者为调整系统、应用程序组件以及涉及决策性能目标和目的的人员。

一个好的性能测试团队包括以下所有关键利益相关者：

- 测试/质量保证小组
- 用户/客户
- 管理者
- 市场人员
- 开发人员
- 网络管理员
- 系统管理员
- 安全管理员

在早期的进程中未涉及的关键人员当结果和结论出现后几乎都是保障问题，在软件性能测试计划进程的早期，将涉及的关键人员介入会有助于减轻以后的问题，特别是结果不理想时。

### 确定性能和业务目标

下一步，我们需要确定性能测试的目标和目的。人们容易在早期出现的一种错误思想就是误以为在软件性能测试中，所有他们需要的仅仅是一个负载测试工具（我们将会在本系列的第四部分讨论工具）。虽然工具是必不可少的，但是一个工具只能够对一个问题提供一种答案，而你所需要了解得是这个问题。如果不了解你需要回答的性能问题，那么你将会很难决定性能测试是否是成功的或者是否是失败的。

我所了解的第一个问题就是“我如何解释工具产生的结果？”我的回答就是“你期望工具做什么？”大多数人都无法回答这个问题。这是指示该性能测试可能没有什么价值的第一个指标。

在很多情况下，测试人员从管理者处得到的是大概的方向。但是，管理者往往倾向于关注结果，比如投资回报率和客户满意度，而这并不是性能目标。如果不了解项目性能测试的目标和目的，工具给予你的大量信息你将无法解释或者不能使用它们去满足管理者的期望。

收集和确定性能的目标和目的是在计划性能测试时最不愉快的一个方面——它时常倾向于行政上的，但是，在你告诉别人不能做事情之前——而不是之后

——你已经浪费了他的时间和金钱。如果你无法实现已经要求的性能目标和目的，那么同意运行测试是没有意义的。

理解性能目标的第一步就是把它们从业务目标中分离出来，图 1 展示了一个例子。



图 1 将性能目标从业务目标中分离出来

要成为一个软件性能目标，必须要有一些系统或者应用程序的要素，这些要素是可以在测试过程中测量的。性能表现不佳时，通常是资源在重压下或者达到最大容量（瓶颈）时。如果你不能设计一个测试用例，那么同意完成任务将不是一个好的主意。我发现以下的一系列问题有助于确定性能目标和业务目标之间的差异：

- 什么样的测试是我将要进行的？
- 什么样的数据是我能够收集的？
- ✓ 为了测量“性能”，你必须得到在系统或应用程序上运行的是什么样的信息。
- 我如何“证明”目标已经完成？
- ✓ 我将使用什么样的测试方法证明一个明确的目标或目的已经实现？

“性能”是指在一个定义好的环境下一些程序如何运转。当你查看如用户工作效率的有关问题时，没有一个要素使你能够测量，去证明客户将会更加的有效率。系统或者应用程序可以完美运行，用户可以在任何他希望的地方工作，你不能强迫人员通过“调整”他们实现更加高效的工作，承诺完成这种类型的目标或

目的几乎注定是失败的。

### 性能测试如何适应于开发过程

很多人看待软件性能测试是你将产品展现给用户的在运输前的最后一道工序，这样的看法是不正确的和危险的。图 2 说明了性能测试的传统方法，这是基于一种观点，即一个完整地系统或应用程序为了性能测试必须存在。

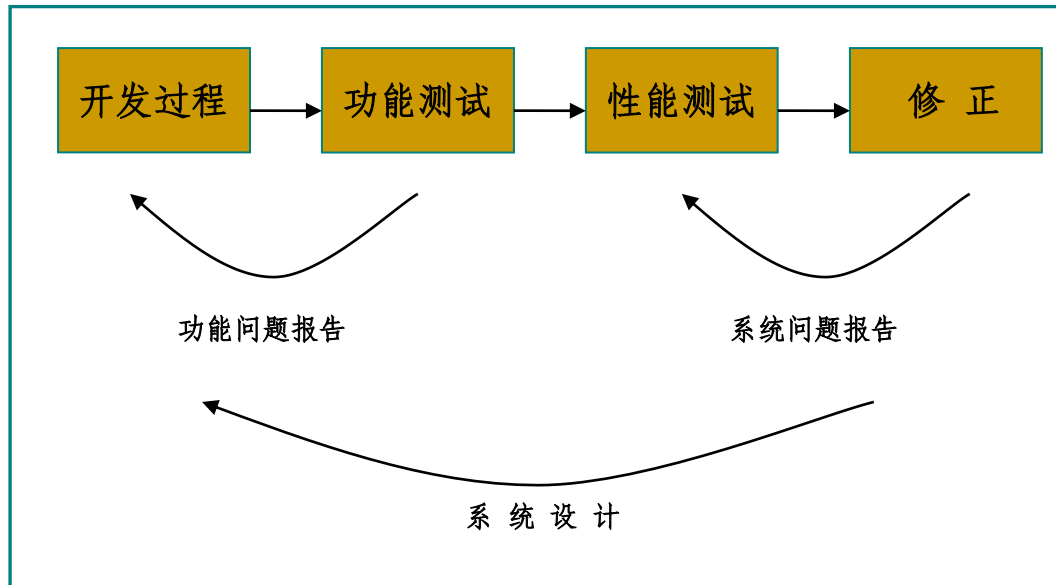


图 2 传统性能测试方法

这种方法主要的问题是，后期发现问题的过程中可能需要修改应用程序或系统，这将需要重新测试被影响的功能——其次是回归测试——需要在性能测试再运行前执行。必须指出的是，一次更改多于一个的功能是非常难协调的，多个技术小组将会参与以及需要去协调，它也可能使一次多种变化可以掩盖解决问题的方法，同时调整数据库和应用程序的代码可能会使得他们互相抵消。因此，很多团队使用一次更改一个因素（OFAT）的方法，。

不幸的是，OFAT 方法不能取得好的效果，在以下几种情况下：

- OFAT 假定的因素是互相独立的。
- 有太多复杂的、难以理解的和隐藏的互相影响的因素。
- OFAT 时间太长。
- 其他的方法（典型的一次调整很多因素的）更难应用的很好。

例如，一旦该数据库是“固定的”，并且重新运行测试，我们可能会发现新的问题，网络或应用程序的，这些问题必须是固定的，反过来这些固定的问题在数据库中又创造了更多的问题。这是有可能终止在一个连续循环中，从一个因素到下一个因素然后重复。

显然，性能测试需要一个稳定的环境和软件，尽管如此，这并不意味着所有的功能都要实现。通过结合预防性思维（静态测试）和增量或迭代式开发，有可

能在有一个或两个稳定功能时就启动性能测试。

增量和迭代式开发应用程序为在一个项目中尽早执行性能测试提供了一个很大的机会。开发过程中尽早的识别性能问题，就能有更多的机会去更改设计和框架，在这些改变的代价变得很高以前，图 3 展示了一个事例：

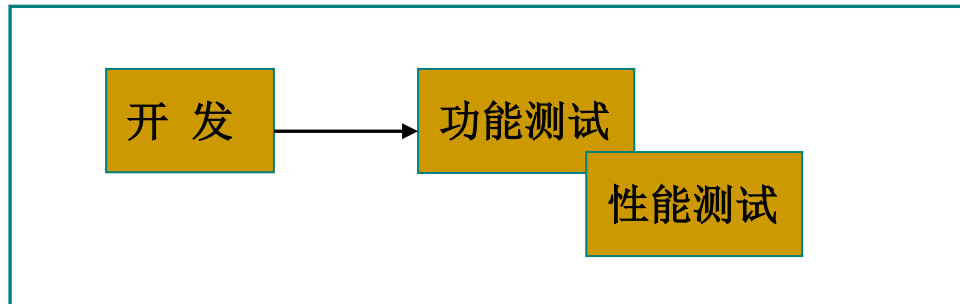


图 3 增量和迭代开发允许早期进行性能测试

在性能测试中利用一个简单的回归技术胜于设计功能，我们可以将潜在的对性能影响的功能，在架构和版本的早期就包括进去。这样可能会影响项目计划，正如功能影响性能可能也会给开发带来很多的问题一样。

一旦第一个基础的功能被测试，性能测试小组就可以开始测试，在此期间，功能小组继续开发下一个增加的功能。这种方法有双重优势：基础架构在早期得到测试，对于完整的功能有了一个限制，并且改变系统架构或者设计的代价还远远不那么昂贵。这种方法的缺点是功能测试小组和性能测试小组会很难协调。性能测试在基础结构和架构中不能发挥很大的作用，这些结构不属于可控测试。

下面的这些是潜在的网络、应用程序和数据库问题，这些问题可以在开发过程的早期使用静态方法确定：

#### 网络问题：

- IP 连接和 HTTP 的使用
- 过度的使用安全功能
- 负载均衡的类型和特点
- 应用程序服务器和数据库服务器可能连接超时
- 包的大小（在应用程序中的大小相对于在网络中的大小）
- 连接池和连接共享
- 服务器在网络中的位置
- 增加延迟和抖动

#### 应用程序问题：

- 过度的记忆容量分配和删除容量分配
- 不恰当的任务初始化和内存管理工作
- 不恰当的碎片集，尤其是在出现缺陷或者未执行时



- 损耗过长时间去保持运转（定时器）
- 持续不断的应用，如 BlackBerry 设置
- 应用配置参数冲突
- 资源消耗高的功能（计算机辅助设计/计算机辅助制造等等）

#### 数据库问题：

- 索引设计
- 使用了动态索引
- 在锁连接中存在潜在的死锁
- 低效利用缓存
- 过度使用缓存
- 频繁使用占用资源的功能比如参照完整性
- 使用预处理程序和触发器事件
- 表格碎裂（过度使用第三常规表格）
- 不恰当的时间表和索引重组

如果你能在他们设计成为系统之前就确定性能问题，这些将远低于昂贵的纠正。当然，这可能会使得项目延缓，如需求或设计需要重写和修订。需要的时间之前，正确的设计缺陷将会对项目产生更大的影响，这就像一个重大的设计项目结束时，被告知该项目时间表存在风险。

不论选择的是何种方法，都应该而且必须在开始阶段为性能测试规划和分析活动，该阶段是项目的较早期。等待的时间过长意味着性能测试小组的灾难。测试者是性能专家（通常是系统管理员、数据库管理员、网络工程师和开发者）的眼睛和耳朵，这些专家负责诊断和解决问题。

由于复杂性和资源的要求，需要建立大规模的性能测试，许多公司可能会决定外包一部分或者全部的测试。通常情况下，外包测试比较容易，他们不需要熟悉业务（例如，性能测试），而其他的则需要测试者了解公司的业务规则（例如，功能测试）。

对于专注于性能测试的公司会掌握很多关键技能表，尽管如此，如果你没有适当的计划还是不要选择外包，外包商将要做和你相同的工作——并且他们需要从你这里得到资料。你将会付钱给那些问你问题的人，这些问题你可能也要问你自己。如果你准备寻找昂贵的咨询师来帮助你找到性能问题的原因，确保在他们到达前一套初步的性能测试脚本已经完成，这将会节省大量的昂贵的“绕弯子”时间。

如果性能测试将要在远程执行，以及应用程序或者系统仍处于开发或者测试，而且仍没有通过企业的防火墙，制定一个可行的早期的测试，以确保远程的

测试者能够通过防火墙和访问系统的测试。

大多数的外包商可以按照以下的类别分为一类或者几类：

- 现场顾问（补充的工作人员）——在客户端的站点进行顾问工作。
- 远程顾问——顾问在自己的站点进行工作，系统或者应用程序托管在客户端站点。
  - 测试实验室——应用程序和系统在异地安装，并且由顾问进行测试。
  - 应用服务供应商或管理服务供应商——内部人员利用硬件和软件托管在第三方的位置在自己的站点测试应用程序或系统。
- 了解性能测试中的各个角色以及将性能测试融入早期的开发过程对于一个成功的测试是非常必要的。如果你没有理解涉及的关键角色并且过早的开始了计划的过程，很大程度上将不会成功。一旦这些关键问题可控，我们可以开始分析问题，并设计一系列必要的性能测试。在下一系列中，我们将会关注架构和基础设施以及确定需要的性能测试类型。



## 如何测试大型 ERP 系统

作者：大傻

**摘要：**大型的 ERP 系统特点是 ERP 系统是一个在全公司或企业范围内部应用的、高度集成的系统，而且业务复杂，操作频繁，数据在各业务系统之间高度共享。而且类似这种系统会根据公司业务发展和其它需要，会在原先的系统不断的增加，修改相应功能，造成系统模块混乱，不稳定。

针对 ERP 系统的特点，我们如何测试呢，要使用怎么的测试过程和测试方法，需要怎样的测试人员，如何进行版本控制，都是我们测试时要想的问题。

**关键词：**大型，业务，数据，测试

### 1 解决方案

#### 1.1 人员

##### 1. 内部培养

- 组织业务培训可以考虑以下三种：1.部门内部进行相应互培，部门里肯定有人在哪个领域有自己特长，内部这种交流可能效果最好，甚至可以结合考核，一年必须内培几次，评分要达到多少。2.邀请公司资深专家授课，可以让人力资源或负责公司内部培训的部门组织开展相应的讲座。3.外聘一些资深人员来授课，或者将一些值得培养的人员送出去，他们回来再来培训整个部门。

- 有意识去培养一些员工有哪方面的技能，如将人员根据业务分类，哪几个人主要负责哪些业务，甚至是哪几个模块。让这些人参与项目调研，评审。甚至在合适的时候“下放”到工程一段时间，做一些实施的工作。

- 让老员工带新员工，部门内部建立帮带制度，并作为考核、晋升的重要指标和条件之一。

##### 2. 其它方法

测试执行过程中可以请一些行业人员或客户来做兼职测试，看看他们关心哪些。比如医疗软件我们就找医生、护士，财务我们就找些会计、出纳。

#### 1.2 方法

##### 1.2.1 升级项目

##### 1. 规范测试流程、提高测试效率

- a) 测试人员尽早参与：由于业务流程复杂，测试人员越尽早熟悉业务需求，越有助于更好的了解业务，了解功能流程，方便全面的设计测试用例；

- b) 严格规范测试流程：由于测试工作本身处于被动状态，存在很多随意性和不确定性因素，所以有效提高测试效率，首先必须制定严格的测试规范、并严

格按流程执行，以便更好的把握测试质量；每个公司都有不同的测试流程，制定最适合自己的，并不断改进、完善的测试规范；

c) 实时控制测试过程：新开发的系统，常因为需求不明确，开发过程中改变需求等，所以实时控制测试流程，及时跟踪变更，实时更新用例；

d) 多总结、多沟通、交流：在项目开发过程中，复杂系统的测试，由于项目进度、测试资源等因素的制约，真正留给测试的时间往往不足，测试人员的每周例会，每日总结很必要，明确每天的测试内容，落实测试工作；多于开发人员沟通，了解开发人员的设计思路，能帮助查漏补缺；

## 2. 提高测试技术、保证测试质量

a) 测试用例的设计：对 ERP 系统，设计高质量的测试用例，需要十分精通业务流程，系统测试用例应尽早设计，尽早完善；

b) 测试执行：由于业务流程复杂，相关联模块密切，测试执行建议结合测试人员的经验和测试用例执行，测试执行时首先保证单个功能正确、数据准确，保证基本流程正确，在基本流程正确的基础上，测试相关业务间的紧密关系，功能的影响、数据的影响；

c) 数据库交互：熟悉数据库结构，通过数据库查询验证数据准确性；

d) 测试方法：交互测试的重要性，交互测试帮助测试人员取长补短，查漏补缺；考虑安全性测试、实用性测试、数据库测试；

e) 自动化测试：不盲目引入自动化测试，自动化测试一般用在系统开发完成，回归测试中；

### 1.2.2 升级项目

系统升级测试基本上是在前期版本的基础上，对新增或修改的一部分功能进行测试，或对以前存在的 BUG 进行修复后测试，并能保证整个系统的所有功能正常。系统基本功能与整个流程是通的，把握系统测试的重点，测试相对容易。

1. 新增功能测试：对新增的模块和功能统一划分，理出与其相关的所有流程，在保证新增功能正确后，验证所有相关流程；

2. 兼容性测试：与原有系统数据的兼容，建议使用空数据库测试基本功能，在基本功能正常下，再导入现有数据测试，验证数据兼容；

3. 整体回归测试：使用最接近用户使用的实际数据，对所有功能回归；

4. 对于长期升级的项目一定建立测试用例复用库，每次升级更新一次，注意复用库结构，可以按业务和模块，如某某业务，某模块，某公用。

### 1.2.3 测试注意事项

1. 测试时不要一个用户从头走到底，模拟实际业务情况，定义不同用户为不同的角色操作对应业务。

2. 所有业务流程属性测试时都要覆盖到，并对有些有关系属性要组合测试。
3. 如果业务流程是有过程，必须有考虑逆操作，如果业务之间有关联，必须根据业务组合设定测试用例。
4. 测试数据的准备必须有跨月、跨年的数据，并且业务流程每个步骤都需要数据，系统基础数据中每种类型数据也需要涉及。
5. 收集第一个客户使用该系统半年内出现的 BUG，并进行分析原因，我们测试中为什么没有发现。

## 性能测试方法论

### Performance Testing Methodology

译者：潘斌

#### 简介

性能测试主关注点之一就是测试能带来多少价值，性能测试是否值得去做？为了确保这一点，交付制品标准必须是可以估量的。

在开始前，我回顾了过去几年参与过的项目，仔细分析了那些成果，以及在大部分成功项目中所做的。我惊喜地发现（虽然之前没有意识到），在这些项目中我都遵循了相同的方法。

#### 增值

在一个性能测试项目中，我们如何确定（估算）测试产生的价值？是否带来了应有的价值？测试是否成功？我们所做的是否把问题暴露了出来？

一个成功的性能测试应该包含以下所有或大部分的内容：

- 成功的测试

什么才是一个成功性能测试？这个可以从多个方面来看。所有的需求都满足并且没有错误被报告出来，这次成功可能是成功的。如果测试没有通过，存在性能或负载方面的问题，它还是一次成功的性能测试，因为问题被发现了。

- 及早发现问题

及早发现缺陷可以降低成本。这一点性能测试和功能测试一样。性能测试应及早开始，在修复这些问题变得复杂且昂贵之前发现它们。

- 改进系统性能

改进系统性能是把测试放在第一位的主要原因之一，这样带来的价值也是巨大的。

- 发现非性能问题

这里面包括一些功能测试不易发现的缺陷。一个很好的例子就是每天事务数计数器。我最近测试一个程序，需求是每天 9999 个事务。这是一个放在数据库里的计数器，没有显示在程序上。在某个点上，负载测试用户开始报失败了。资源和性能都很好，但是这一天只通过了 999 个事务。系统只满足了这些，而不是 9999。之前没有提过缺陷，因为功能测试从未达到过每天 999 的事务量。

- 可见的交付品

讨论自己发现的问题这很不错，不过如果能把它们展示出来（能看到并且有效），这样就更好了。可见的交付品更易于交流你的成果，还能展示投入了多少

时间。

本文所说的方法包括以下交付品：

- 评估报告
- 测试策略/计划
- 测试脚本
- 测试场景
- 测试结果
- 测试结果总结报表
- 测试报告及简报

## 方法

所有的项目都应该采用相同的方法，这点很重要。它确保所产生的交付品都是一致的，并且质量也是可以度量的。通过对交付品进行评估，我们可以对不同的项目进行比较。这样容易识别出成功的项目，而对不成功或者未带来应有价值的项目，我们也可以轻松地找出其中正在犯或者已经犯过的错误。

**恰当的方法是成功性能测试的保证。**

通过点状图的形式查看系统提升，或者把改进区域以百分比的形式显示，这两种方式都可以确定所带来的价值。图 1 展示了两个平均响应时间图。左侧的图显示的是第一轮测试的响应时间，而右侧的图显示了这个项目最后一轮测试的响应时间，性能的改进（增值）是很明显的。这些图表在测试以后应该可以看到，这样就可以把问题展示给人们。

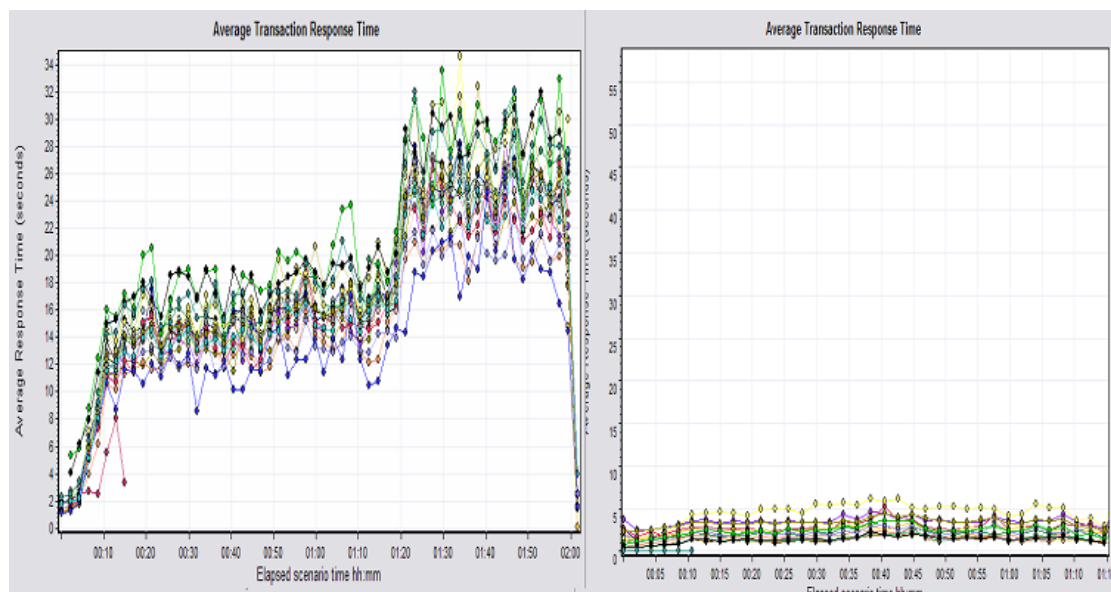


图 1 首轮和最后一轮测试结果的比较

性能测试由六个阶段组成，每个阶段都有制品完成。表 1 显示了每个阶段所

应交付的制品。

表 1

阶段	制品
1-项目评估	评估报告
2-计划	测试计划
3-制作脚本	测试脚本
4-测试执行	测试场景，测试结果
5-结果分析	测试结果总结报告
6-报告	测试报告及简报

### 第一阶段 项目评估

我所完成的所有成功项目都始于一个规范的评估过程。这个阶段决定了这项工作是否可以做？如何做？评估是一个收集信息的过程。分析需求、研究系统架构，并确定在特定情况下，需求是否能满足。为了做好测试，还要其他人明确你的需求，包括时间需求，什么时候得到有效结果。

**期望管理应持续在整个项目过程中**

人们通常假设结果能很快很容易地交付。如果你在用一些自动化工具，通常对性能测试而言，这句话更是适用。评估过程是期望管理的开始。

#### 需求

性能测试的需求通常是非常具体的。开始评估时需求并不清晰，发现需求应该是评估过程的一部分。对每一个细节收集尽可能多的信息。这一点对项目成败而言及其关键。要生产出好的制品，所有的信息都要用到。如果有些信息没有，可以采用适当的方式进行沟通。表 2 我们可以看到评估阶段涵盖的一些关键区域。

表 2

项目评估	
哪些是必须达到的目标？ (要解决的业务问题)	<ul style="list-style-type: none"> <li>● 用户数</li> <li>● 可接受的响应时间</li> <li>● 要测试的业务过程</li> <li>● 基线版本</li> <li>● 数据量</li> </ul>
架构/平台	<ul style="list-style-type: none"> <li>● 你熟悉这个架构吗？</li> <li>● 你有针对这个架构的经验吗？</li> <li>● 系统部件（软件及硬件）</li> </ul>
测试环境	<ul style="list-style-type: none"> <li>● 是否适合性能测试？</li> <li>● 硬件</li> <li>● 软件</li> </ul>
将使用哪些工具？	<ul style="list-style-type: none"> <li>● 你熟悉这个工具吗？</li> <li>● 这些工具是否适合这个架构？</li> <li>● 安装和使用这些工具的软硬件配置要求</li> </ul>
监控	<ul style="list-style-type: none"> <li>● 哪些是必须监控的？-需求</li> <li>● 哪些能够被监控？</li> <li>● 建立监控的需求</li> </ul>
可用时间	<ul style="list-style-type: none"> <li>● 可用时间 VS 动作时间</li> <li>● 给你充足的时间</li> <li>● 期望管理</li> <li>● 充足的时间=有价值的结果</li> </ul>
执行测试的需求	<ul style="list-style-type: none"> <li>● 跟关键人物的沟通</li> <li>● 硬件需求</li> <li>● 软件需求</li> <li>● 数据需求</li> </ul>
客户期望	<ul style="list-style-type: none"> <li>● 不要对任何事点头</li> <li>● 指出你的局限</li> <li>● 突出风险</li> <li>● 突出测试范围外的内容，并说明理由</li> </ul>
评估报告	<ul style="list-style-type: none"> <li>● 能做吗？</li> <li>● 怎么做？</li> <li>● 谁来做？</li> <li>● 要投入多少时间和精力？</li> <li>● 需要排除哪些东西？</li> <li>● 要交付的制品是什么？</li> </ul>

结合所有的调查结果，可以在评估结尾草拟一个报告，它包括一些决策，如项目执行措施以及测试预估时间等。图 2 展示了一个典型评估报告的目录。



目录	
1	目标..... 3
2	评估过程..... 3
3	评估结果..... 4
4	风险..... 4
5	性能测试建议..... 5
5.1	性能测试方法..... 5
5.2	时间和投入..... 5
5.3	前提与假设..... 6
5.4	交付品..... 6
6	结论..... 7

图 2

## 第二阶段 计划

在项目评估中收集的信息可以用来制定性能测试计划，并用来启动性能测试计划。性能测试计划必须包括所有的细节、核对清单以及测试执行参考。测试计划是测试过程的脊梁，它还是一个工作文档，要在项目过程中不断更新。完整的测试计划确保没有细节被遗漏。

*完整的测试计划确保  
没有细节被遗漏*

请注意本文并没有覆盖测试计划的所有内容，但有一些是应该包括在测试计划中的关键元素。它们是：

- 愿景
- 目标
- 范围
- 系统图表
- 排除内容
- 监控
- 责任人
- 环境
- 需要的测试硬件
- 测试的软件要求
- 测试的数据要求
- 测试的工具要求
- 安全系统
- 测试场景
- 结果分析



## ■ 报告及反馈

补充完整测试计划，这一点非常重要，涉及测试要求的这些章节都要完整。这里面包括测试开始前所需的资源以及其他后勤支持。完整计划保证测试执行所需的各种东西都将到位。图 3 显示了一个典型性能测试计划的目录。

目录	
1	愿景.....3
2	目标.....3
3	范围.....3
3.1	待测试组件.....4
3.2	排除内容.....4
3.3	监控.....4
4	测试要求.....5
4.1	责任人.....5
4.2	环境.....5
4.3	需要的测试数据.....5
4.4	需要的测试软件.....6
4.5	需要的测试硬件.....6
4.6	安全系统.....6
5	测试过程指导.....7
5.1	核对清单.....7
6	结果分析.....8
附件 A	.....9

图 3

## 第三阶段 编写脚本

刚开始测试时，如果发现问题，我们一般都会归咎于测试工具和脚本。你必须 100% 确信你的脚本不会导致系统上的任何问题或错误。掌握工具原理以及它是怎样和系统交互的，这些很重要。你需要完全信任你的工具和脚本。你需要确信这一点，同时还要争取关键人物的信任。

**你需要完全信任你的  
工具和脚本**

对性能测试人员而言，最大的挑战之一就是赢得测试以外人员的信任。包括开发人员、系统架构师、数据库管理人员和网络管理人员。任何和系统或程序表现有关的人员都会受到性能测试结果的影响。如果你能让这些人站在你这边并在早期获得他们的信任，那么项目的前景就比较乐观了。

编写脚本是测试的开始。你必须对这个程序有 100% 的认识，这一点相当重要。好的测试人员在测试开始前对系统要有感觉。当你在研究系统和编写脚本的时候，记录下响应时间以及很慢或很忙的进程。这些都可能是系统潜在

**好的测试人员对系统要  
有感觉**

的瓶颈。开始执行脚本的时候，密切关注你之前记录的哪些地方，有可能你在正式完成性能或负载测试之前就能定位出一些问题。

监控工作在编写脚本阶段也要开始了。第一次运行脚本时，事务的响应时间就应该被记录下来。跟踪这些响应时间是很重要的，因为系统表现的变化通过仅运行一个脚本就可以被发现，这样可以节约做完整性能测试的准备和配置时间。每次系统有所改变，都要运行单独的脚本至少一次。图 4 显示了在一个脚本回放日志中响应时间的例子。在图 5，一次改动以后，可以看到系统的这个响应时间有了负面的影响。发现这个问题就没有改动以后进一步测试的必要，因为性能明显变糟，或者可能是实施过程出现问题。

每时每刻对单个脚本的每个细节保持关注可以减少很多时间和麻烦。有这样的情况，为多个用户每人准备了一套数据，然后准备大干一场的时候发现有些地方不对，整个数据或测试场景不得不重新再来一次。没有什么比这种情况更糟了。关注细节并且充分理解系统的表现有助于避免这类的情况。

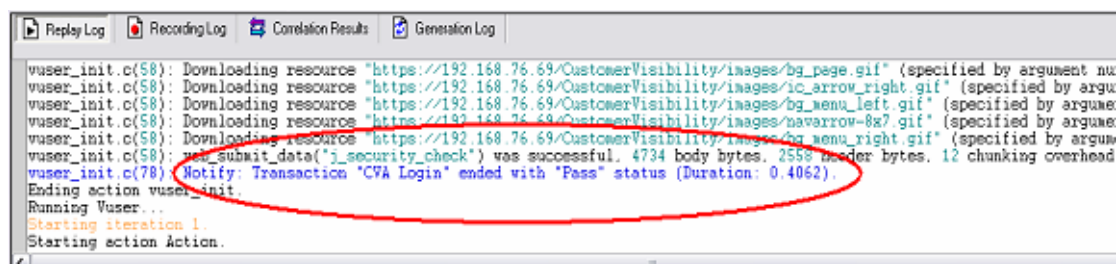


图 4 改动前的响应时间

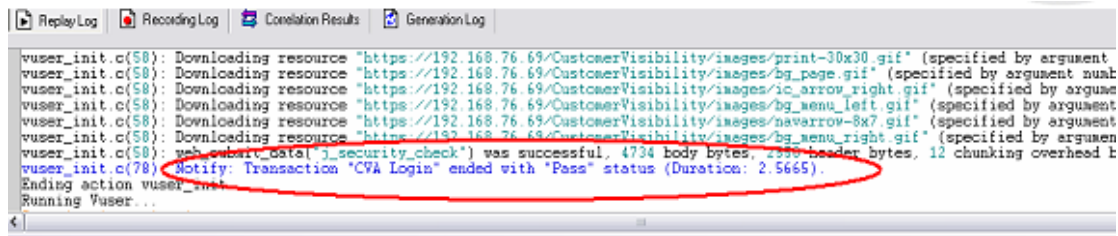


图 5 改动后（负面影响）的响应时间

最好的做法是在编写脚本时记住以下几点：

- 确定这个程序是可以脚本化测试的；
- 确保对程序有 100% 的了解
- 理解业务过程
- 知道需要哪些测试数据
- 了解系统运行的环境
- 对系统表现有所感觉
- 确保脚本运用到整个环境
- 为每个步骤添加响应时间的度量

- 从第一次脚本运行开始记录响应时间
- 验证脚本在数据库中的执行
- 恰当地管理测试数据
- 在每次改动或实施以后运行单独的脚本至少一次

#### 第四阶段 测试执行

谈到性能测试有不同的观点和方法。大部分人把负载测试和压力测试称之为性能测试。这无所谓对错，但我是把性能测试看做一个整体，它包括表 3 提到的那些测试。本文提到的性能测试方法就是从这个术语发展而来的。

表 3

性能测试	
类型	描述
基线测试	建立性能极限标准
负载测试	从整个系统的角度模拟产品负载
压力测试	给系统增加负载直到极限
渗入测试	负载情况下长时间的测试
数据量测试	海量的数据量/吞吐量。数据库增长

#### 基线测试

基线测试经常被提及，但也总是被忽略。它所带来的远不止是建立一个性能测试的标准，它也是性能测试方法中最重要的步骤之一。花上一些时间和精力来检查细节，85%以上的性能问题可以再基线测试中被发现并解决。不幸的是通常基线测试没有足够时间。所以在项目开始就要包括基线测试，把它排入计划，这一点很重要。

基线测试是通过每个脚本单独完成。每个脚本一般运行 1，2，5，10 以及 20 个用户。最大用户数会根据项目而有所不同，它依赖于事务类型以及脚本录制的这些业务过程。在一些情况下 5 或 10 个用户可能是某些特定脚本的上限。

在基线测试过程中应该完成全部的监控指标。所有的结果必须被保存下来并分析。这样做的优点是所有的度量值在一个进程或事务中，这样问题可以被识别出来，不会有因为分离进程而导致的问题或错误。如图 6 显示了一个脚本的响应时间。图 7 显示的是 20 个用户的测试结果，它导致了非常高的 CPU 占用率，它仅在一个脚本中发生。

**85%的性能问题可以在基线测试中被发现并解决。**

**这样做的优点是所有的度量值都在一个进程中**

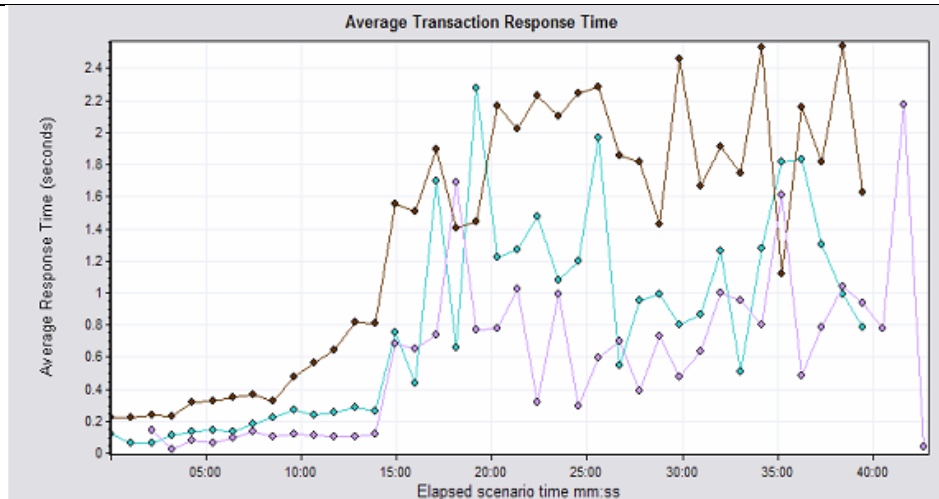


图6 一个脚本的响应时间图，它有三个度量值

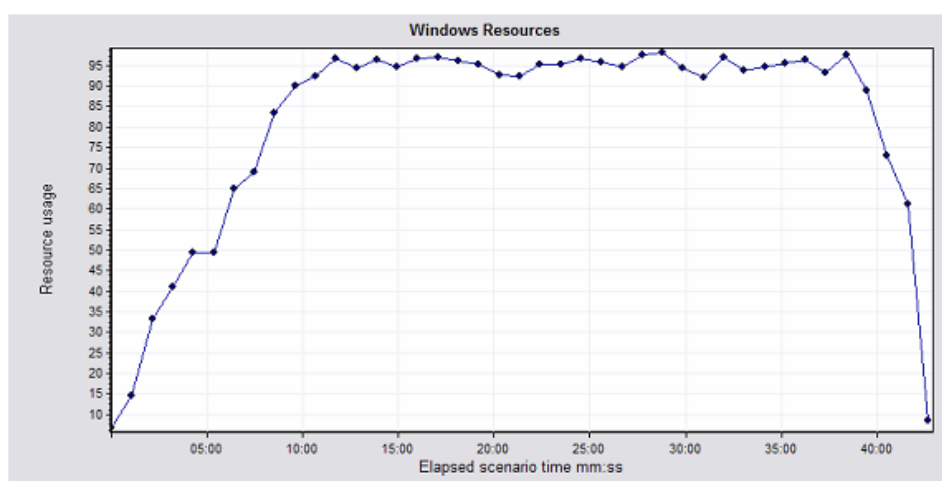


图7 20个用户访问单个进程，CPU 占用率非常高

这个问题通过基线测试发现，它在一个脚本中被隔离出来。没有浪费时间来配置和准备多个脚本完整的负载测试。如果从如图8所示的一个典型负载测试响应时间图中，定位出造成高CPU的原因，那样要花更多的时间。我们的目标是在基线测试中及早消除问题，在第一次负载测试完成时能有一次几乎是“干净的运行”。这样可以减少每个人，在尝试有很多用户同时又监控多个事务的情况下，定位问题或瓶颈原因所带来的挫折感。

目标是在第一次负载测试完成时能有一次“干净的运行”

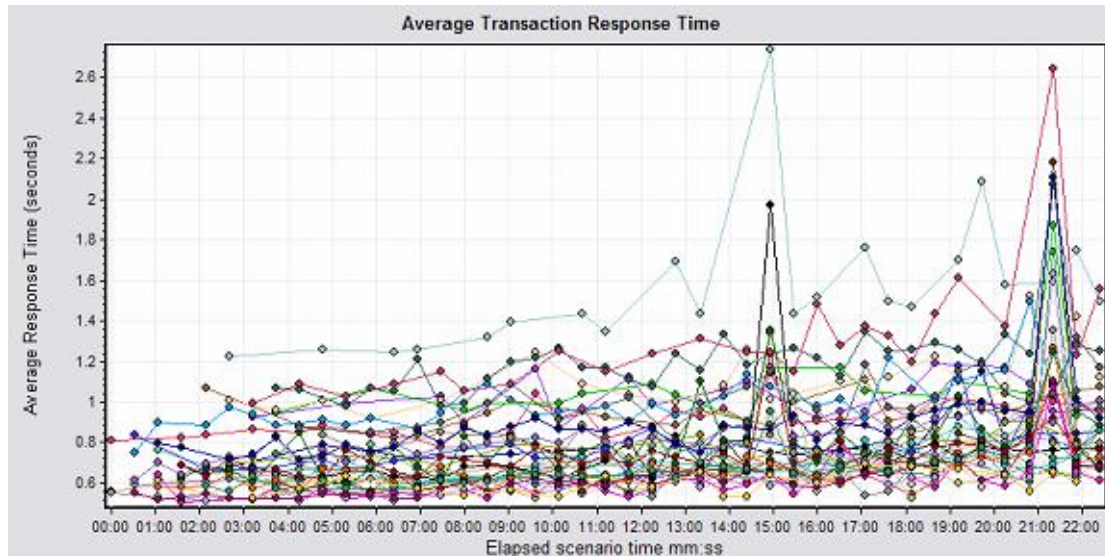


图 8 在很多进程中分隔问题是不可能的

在负载测试开始之前,按照这个方法完成所有基线测试,这能确保每次测试运行都能产生有意义的结果。这是测试过程中很重要的一个方面。因为早期有价值的结果,能使人看到你的成绩,就像在一开始你所创造的价值一样。

负载测试

*早期有价值的结果,能使人  
看到你的成绩*

当开始我的性能测试生涯时,我受到训练,如何进行计划、编写脚本以及执行负载测试。虽然在编写脚本阶段以及数据准备阶段会单独执行脚本,但目标是为负载测试做准备。这样作主要的缺点是要经过长时间的准备,才能得到第一轮测试结果,而这个结果经常是没有意义甚至是不可读的。即使通过这样的方法发现问题了,要确定问题原因也是非常困难和耗费时间的。加上性能测试结果中这个非常“繁忙”的图标,为什么要采用更有效更有意义方法的原因就很清楚了。

上述因素导致一种能快速交付且结果更有意义新方法的发展。负载测试现在只是性能测试项目中的一部分,而且大部分的问题在第一次负载测试之前已经被发现并且解决了。

*大部分问题在第一次负载测  
试前已经被发现和解决*

压力测试

运行压力测试的目的是发现系统的临界点。它应该在所有来自于性能测试的问题都被解决以后开始。压力测试的结果可以用作规划容量。它可以让管理员指导系统的崩溃即恢复情况。在项目结束前至少要包括一次压力测试。

浸入测试

浸入测试是运行较长时间的负载测试。内存泄漏可能是在浸入测试中最主要要去发现的问题,但在测试中经常可以发现连接超时的问题,数据库的增长也是可以被监测到的。在计划做浸入测试时,应该要询问所有有关人员,在各自领域



中，哪些是需要被监测的。

在计划和执行侵入测试时，时间和其他逻辑问题诸如测试数据，是要克服的主要问题。测试应尽可能长地运行，或者在某些监测点中发现某些趋势。如果出现任何严重的缺陷，这个就很有可能决定了侵入测试的持续时间。

#### 数据量测试

数据量测试指的是规模，或者更具体指的是数据库或文件的大小。这不一定是需求，但有可能很重要，这取决于所测试的程序类型。数据库

**数据库的规模可能会对性能造成极大地影响**

的规模可能会对性能造成极大地影响。如果相对于实际应用的场景，性能测试是在一个小规模的数据库中测试，那么它应该作为一个风险被提出来。

在数据量测试中，大用户量的负载测试可能不需要，而需要仔细规划如何执行测试。

### 第五阶段 结果分析

测试结果分析可能是性能测试中最具挑战性的一个方面。它从测试场景设计开始，测试这个场景，当你在测试结尾看着这个测试结果，它会给你正确的“图案”。并不总是会得到有意义的结果，但我相信你在测试中想要得到的结果，必须是设计性能场景时的目标。

**你在测试中想得到的结果，必须是设计性能测试场景时的目标**

测试结果对性能测试人员而言，是最重要的交付品。这毕竟是最有效展示测试成果的方式。在本文开始，我提到测试项目的第一次和最后一次结果。这是工作的目标。把第一次和最后一次测试结果进行比较，这作为性能的结果，可以看到性能改进了多少。

为了确保结果分析的成功，需要遵循这两个规则：

#### 1、保存所有东西

保存你每次测试的结果并给它取名。使用适当的明确的命名，这样便于在以后阶段中参考。

#### 2、保持对所有事的跟踪

为什么测试失败，为什么性能很糟糕或是跟之前的不一样？这些都要记录下来。为什么性能良好或是比以前更好？这些也要记录下来。有哪些改变？在每次测试运行以后，要把改动，以及这些改动对系统性能的影响都记录在测试总结报告中。包这些都保存下来，它们会在以后用到或者在最后的测试报告中用到。

性能测试是一个需要运行多次，反复迭代的过程。简短的总结是在运行中表示测试结果最有效的方式，在测试运行中经常没有足够的时间编写一份完整的测试报告。总结文档是很好的有形交付品，它可以使你的努力更容易被投资人看到。

测试总结包括以下内容：

- 概述
- 场景总结
- 用户数
- 最大用户数
- 间隔
- 总吞吐量（字节）
- 总点击数
- 每秒点击数
- 图表
- 响应时间图表
- 系统资源图表
- 比较图
- 下次测试建议

## 第六阶段 报告

性能测试方法中最后一个阶段，是在报告中反馈整个项目的成果以及进展。一份完整的性能测试报告和一份简报一起交付，在简报中向相关人员介绍报告的内容。

*一份单独的报告不太有效*

这样做的目的是讲解报告的内容，以及回答任何人对测试和成果可能会提出的问题。通过经验，我发现一份单独的报告不太有效，大部分人不会去读它。用报告加简报的形式，这样非技术人员也能读懂。

这个期末报告不是指某次测试的结果，它涵盖了整个测试进程的所有成果。这里面的图表主要用来进行比较，突出整个项目中性能提升的重点。不需要包括结果明细，但是要引用一些有关的结果总结，这里面涉及到一些具体的问题。

## 总结

本文所说的方法在各种架构的各种项目中得到验证。它并不是唯一有效方法，但它能确保你得到一个不错的结果。看到增值部分并能确保成功，这些是开发并实施这一方法的主要原因。