测试基于Web的应用程序  
测试web应用程序和测试桌面系统用很多共同点：例如你需要和执行所有标准测试类型一样测试常见的功能点，配置及兼容性。但是由于与应用程序交互的所有分布式系统组件的复杂性成倍的增加的原因，导致web应用程序测试更加的困难。当我们在web环境中看到一个错误时，通常很难指出错误发生的地方，并且由于我们看到的行为或我们接受到的错误信息可能是发生在Web系统中不同部分的错误的结果。错误可能是很难重现的。那么我们如何在web系统中分析错误呢，并且为了重现那些错误又应该做哪些考虑呢？  
当我们对潜在的[**技术**](http://www.51testing.com/javascrīpt:;)有一个了解时，我们可以更好的最大化测试效率－编写更多可重现的bug报告并且在较少的时间里发现更多的错误。说比做更加容易－特别是在web环境里。Web环境在错误倾向技术变量是密度高的。以下是测试Web应用程序的需要考虑的5个基本事项：  
1． 当我们在客户端看到一个错误时，我们所看到的是错误的症状，而不是错误本身。  
2． 错误可能是与环境相关的，并且可能不出现在不同的环境里  
3． 错误可能是存在代码或是配置中的  
4． 错误可能驻留在几个层中的任一个层中  
5． 检查[**操作系统**](http://www.51testing.com/javascrīpt:;)中的两个类别－静态vs动态－需要不同的方法。  
现在让我们来详细的看看这5个需要考虑的事项。  
   
1. 什么是我们真正看到的东西？是一个错误还是一个症状？  
如果不诊断环境，我们不能够确定是什么导致了一个症状出现。如果客户端和服务器端的一个环境特定的变量被移除或被改变的话，我们或许将不能够重现问题。  
例如，我正在测试一个Web的缺陷跟踪应用程序，并且遍历创建一个bug报告的流程。当我选择“新建”按钮时，我接收到一个错误信息：Microsoft OLE DB Provider for ODBC Drivers error '80040e14'。在花了一些时间调查我的浏览器环境后，我发现Javascrīpt在浏览器的参数设置对话框中被禁止了。启用Javascrīpt 就消除了这个错误。（这个问题是否是个bug不在我们今天讨论的范围里）这里是要说如果我在bug报告中增加关于Javascrīpt的信息，我可以节约我们团队花费在分析这个问题的时间。此外，“禁用Javascrīpt”从此应该要添加到我的测试包中；它将被应用到应用程序的各个地方，以使所有潜在的相关问题不会出现。  
   
2. 这个错误是环境依赖的吗？  
为了重现一个环境相关的错误，我们不得不完全地复制活动的准确顺序和应用程序操作所在环境的条件（操作系统，浏览器版本，插件的组件，数据库服务器，web服务器，第三方组件，服务器/客户端资源，网络带宽和通信量等等）。例如，当你试图使用一个28.8 kbps的拨号连接登录到你的Web应用程序中，你会碰到一个由于在认证过程中因超时而导致的登录失败－但是同样的登录步骤如果你用一个1.54 mbps 的T－1连接将会成功的通过认证。在这个案例中，你有一个环境依赖的错误，这个依赖条件是在带宽中。  
环境无依赖的错误，用另一种话说，相对来说是容易重现的－它没有必要复制操作环境。环境无关的错误，需要复制所有都能够揭示错误的步骤。例如，如果公司的名称在所有产品在线页面上错误地拼写为WebTessting.Con, 你就总能看到这个错误－它是和硬件，软件和你操作环境中资源变量无关的。更为常见的是，我们将环境无关的错误称为功能特定的错误。  
   
3. 是一个代码错误或是一个配置问题  
错误（或是假定错误的症状）可能会在代码修复中或系统重新配置（客户，服务器或网络）解决（假设错误是真实的）。不要太快的下结果它是一个bug。  
Microsoft OLE DB Provider for ODBC Drivers error '80004005' 对比真正的软件错误，这是一个说明识别可能的配置问题挑战。它显示了由于Web应用程序“登录失败”而引起的一个错误信息。只是简单的查看这个错误信息，是不可能判断这个错误是由于软件bug引起的还是服务器端配置问题，或是兼容性问题，浏览器配置问题或以上所有的。  
在进一步分析这个失败以后，我发现几个可能的产生这个错误信息的条件:  
IIS (Web server) virtual directory has not been set up properly当虚拟目录没有被正确的配置时，将找不到请求的文件，脚本或数据。这是一个典型的服务器配置的问题。然而，如果安装程序未能根据说明书一样配置web服务器，那么这是一个软件的错误。如果一个系统管理员未能根据说明书正确地配置web服务器，这个就变成了用户错误。  
Application directory has not been configured properly to execute scrīpts一个典型的应用服务器目录包含了需要执行的脚本，它们会被代表客户端的Web服务器调用。为了安全的原因，一个Web服务器可以被配置以允许或不允许脚本在一些目录里执行。如果你的应用服务器目录被设计来包含将要被执行的脚本－但是Web服务器被配置为在那个目录里禁用脚本执行－应用程序将不能工作。这是软件错误还是一个配置问题呢？  
Default Web page has not been set up properly这个问题和上面的问题相似  
[**SQL**](http://www.51testing.com/javascrīpt:;) Server is not running为了执行查询，存储过程和访问数据，应用服务器需要连接后台在SQL服务器上的数据库。如果SQL服务器进程没有运行，显然应用程序将不能工作。  
DLL/COM objects are missing or were unsuccessfully registered可能安装程序在安装过程中未能复制应用服务器要使用的所有DLL。如果遗漏了其中一个应用程序所需的DLL，应用程序将不可以工作。  
也可能安装程序正确的复制了所有需要的模块，但是失败的注册一个或多个DLL。例如OLE－Based的对象，例如COM或DCOM，它们的class ID(CLSID) 在它们可以被使用之前必须注册到注册表库中。如果一个应用程序试图访问一个没有被成功注册的COM对象，应用程序将不能工作。  
这个问题通常由安装过程中的错误引起来。另一方面，如果组件必须被手工注册地话，就变成一个配置问题。  
Browser-side Javascrīpt setting has been disabled这是一个浏览器端的配置问题，由于应用程序要求浏览器启用Javascrīpt。这是一个软件错误，配置问题或是一个技术支持的问题呢？  
   
4．哪个层真正的引起了那个问题？  
在Web 系统中的错误通常是很难一直重现因为许多由C/S架构的分布式特性而引入的许多变量。（例如，服务器，客户端和网络组件）。在一个web环境中至少由3个常见的怀疑部分：客户端，服务器和网络。客户端和服务器都会携带诶之和兼容性问题，那些和PC环境相似，所有的组件都在一个盒子里。在C/S系统里，问题成倍的增长，然而，由于可能有很多的客户端和服务器链接在一个网络中。典型的C/S配置和兼容性问题涉及到硬件和操作系统的混合（例如，基于UNIX的 vs基于windows的盒子）以及在服务器端的软件组合（Web服务器包，数据库服务器包，防火墙，COM对象，CORBA对象等等）。问题也可能涉及客户端的软件组合（TCP/IP堆栈，拨号软件，帮助组件，浏览器带宽和浏览器版本）。另外，浏览器设置，例如一些常见的设置，连接设置，安全设置（包括 ActiveX空间，插件，Java，脚本，下载，用户认证等等），内容设置，程序设置，和其他高级设置（包括浏览器选项，多媒体选项，JVM选项，打印选项和HTTP选项）引入很多可以被测试并分析的变量。  
网络提供了另一套变量。网络用几个方式影响着Web应用程序，包括由于带宽和响应时间引起的分时相关的问题（竞态条件，性能，超时等等），由于硬件设备例如网关和路由器导致的潜在的配置和兼容性问题，以及和安全实现相关的端问题。  
   
5．静态和动态操作环境是不同的  
一般来说，有两类操作环境－每个都有自己独一无二的测试牵连：  
静态环境（例如配置和兼容性错误）不兼容性问题可能存在其中，不管可变的条件，例如处理速度和可用的内存  
动态环境（例如资源及时间相关的错误）其他方面可兼容的组件可能出现错误在其中，由于内存相关的错误和反应时间条件（我们将在这一节中更详尽的探讨动态环境）  
   
静态操作环境：配置和兼容性变量  
配置和兼容性问题可能会出现在web系统中的任何一个点上：客户端，服务器端，或网络中。配置问题包括不同的服务器软件和硬件设置，浏览器设置，网络连接，和TCP/IP堆栈设置。浏览器设置/ 前面提到的Javascrīpt例子说明了配置问题的一种类型。图1和图2展示的是另一个配置问题的类型，两种可能的物理服务器配置：one-box 和two-box配置。  
我们用来示范的所测试应用程序有一些制图的功能，可以让用户生成度量报告，例如条形图和直线图。当用户请求一个度量报告时，应用程序服务器执行的伪码如下：  
1． 连接服务器并运行查询，  
2． 编写查询结果到一个名为c:\temp\chart.val的文件中，  
3． 执行Chart的JavaApplet。从c:\temp\chart.val文件中读取数据以生成一个图表  
4． 发送JavaApplet到浏览器  
在测试这个应用程序过程中，我发现图表功能可以在以上的配置上运行，但是却不能在其他配置上工作。在我更进一步的研究之后，我认识到问题可能出现在two- box配置中。在检查代码之后，我认识到问题在步骤2和3中。在步骤2中，查询结果被写到数据库服务器本地驱动器中c:\temp\chart.val。在步骤3里，Chart JavaApplet是运行在应用服务器上而不是和数据库服务器在一个相同的盒中。当它试图在应用服务器本地驱动器中打开c:\temp\ chart.val文件时，文件并不存在。  
在这个用例中，我不建议在遇到问题时就阅读代码，我把调试的工作留给开发人员。我只不过想指出识别哪个服务器配置是有问题的，并且在bug报告中含括这些信息。我也会在测试下的应用程序支持的全部的分别式配置下运行一个粗略的测试用例包。  
配置问题在静态操作环境中也是很终于的。例如，在图3中我们看到在Netscape Navigator和IE浏览器的一个兼容性区别。  
这个例子并不是要说IE比Netscape Navigator更好，它只不过意味着在浏览器之间有不兼容性问题－并且代码应该假设相对路径在所有的浏览器中都可以工作。更重要的是，它建议当你在一个环境中发现一个错误时，如果它是一个环境相关的错误的话，同样的错误可能不会出现在不同的环境中。  
动态的操作环境：事情不会保持一样  
当特定环境的属性值不是每次都在测试过程中保持常量时，它会引起操作环境变为动态。属性可以从资源特定（可用的RAM，磁盘空间等）转变为时间特定的（网络反应时间，用户要提交的交易顺序等）。  
当一个测试用例取决于步骤集和操作环境的准确复制，然而（由于它的动态本质）操作环境不可能被复制，错误变得不可重现或很难重现。  
顺便说一下，这也是内存相关错误通常较难重现的原因。当一个内存覆盖的错误出现在代码中时，例如，它常常会引起一个内存覆盖的问题。然而，从一个黑盒测试的角度看，我们永远没有机会看到这个错误的症状直到执行或读取特定的代码或数据溢出字节。在这个例子中，步骤集代表了黑盒测试的准确集合。内存覆盖错误代表了在代码中的真实的错误。被执行或读取的被覆盖的字节的条件代表了动态的操作环境或需要揭露（重现）错误的条件。  
这是一个动态环境相关错误的Web应用程序例子，我们在其中将调查一个时间相关错误。功能说明书要求：  
• 在系统中的项目名称必须是唯一的  
• 为了可能的复制需要在客户端使用Javascrīpt来执行错误识别和处理  
• 用户将可以通过请求项目设置页面增加或删除项目名称  
• 当一个用户创建一个新的项目名称时，浏览器端的Javascrīpt检查输入的名称和内嵌在HTML页面中选择列表（如图4）。  
看看图5中的时间相关的错误。在项目设置页面之前和之后的屏幕截图中说明了应用程序失败检测重名的“Doomed”。图4解释了这个时间相关的错误，它包括了两个用户增加新的项目名称到同一个数据库中。  
如表1中所示，用户A和B同时创建新的项目，但是并不知道其他人的动作。在步骤3中，用户A增加了一个名为Another的项目。由于这个项目名称已经存在，他浏览器的Javascrīpt会显示一个提示他输入不同项目名称的信息。  
用户B增加了一个项目名称为Doomed。她浏览器的Javascrīpt不会检测Doomed为一个已经存在的项目名并且添加它到数据库中并返回列表。更新过的项目名称列表被发送到用户B。  
用户A随后添加相同名称Doomed到项目列表中。他浏览器的Javascrīpt没有在HTML列表中检测，因此Doomed会再次被添加到数据库中－同样到了返回的列表中。更新的项目名称列表被发送给用户A，并且包括两个Doomed的条目。  
这个结果未能满足产品的说明书。除非这种情况出现在一个设计良好的测试用例，偶然发现这个错误并且试图重现它不是一个简单的工作。在这个例子中，实际的错误是应用程序在检查服务器端重名（除了客户端检查以外）的失误。这些步骤包括用户A的活动。通过用户B的活动创建了动态操作环境－这些活动对于用户A是隐藏的或不知道的。  
   
总结  
为了有效的在Web环境中分析并重现错误，你需要对操作环境有个掌握。你也需要理解环境特定的变量可能会影响你复制错误的能力。在应用程序有着这份文章中的一些技能，我希望你的Web测试经验将会更少的被挫败和更加开心。  
记住没有任何东西将替代你的测试技能－你编写出好的测试用例，问相关what-if的问题，保留仔细的记录，并且有系统的研究难以重现的错误的能力。就是这些技巧不仅在寻找错误中给你提供帮助，而且也会帮助你发现那些和他们相关的隐藏错误。