使用 ACT 对 Web 程序进行性能容量测试

刘艳会

通过对 WAS 使用文档的阅读，我们已经知道了压力测试的必要性，并且我们也已经可以使用 WAS 来对 Web 应用程序进行压力测试。下面我们将对 Microsoft 提供的另外一个类似的测试工具 ACT 进行介绍。我们将从以下几个方面来介绍：

- ACT 概要介绍
- 使用录制的方式生成测试脚本
- 理解测试报表
- 录制脚本会遇到的问题以及解决方法
- 使用 VBScript 编写 ACT 测试脚本
- ACT 与 WAS 的比较

1 ACT 概要介绍

ACT（全名为 Microsoft Application Center Test）是 Microsoft 的一个单独的工具，不过这里我们介绍的是 ACT 是 VS.NET 版本，不具备一些功能（可以参见帮助中的“ 功能 ”一节）。

ACT 是专门为对 Web 服务器进行压力测试和分析 Web 应用程序（包括 Active Server Pages (ASP) 及其所用的组件）的性能和可伸缩性问题而设计的。它通过与服务器建立多个连接并快速发送 HTTP 请求来模拟成员众多的一组用户，可以对 Web 应用程序进行持续时间长、高负载的应力测试。

开发人员可通过使用 Application Center Test，方便地测试 XML Web 服务和应用程序的性能和功能是否正确。使用浏览器的记录功能快速创建性能测试脚本，这些脚本可在 Visual Studio .NET 环境中修改和运行。Application Center Test 提供完全自动化的模型，以使开发人员可方便地创建测试套件（当新项目版本可用时，这些套件可自动运行），从而提高开发效率和准确率。

2 开始使用 ACT

使用 ACT 创建测试脚本有两种方法：

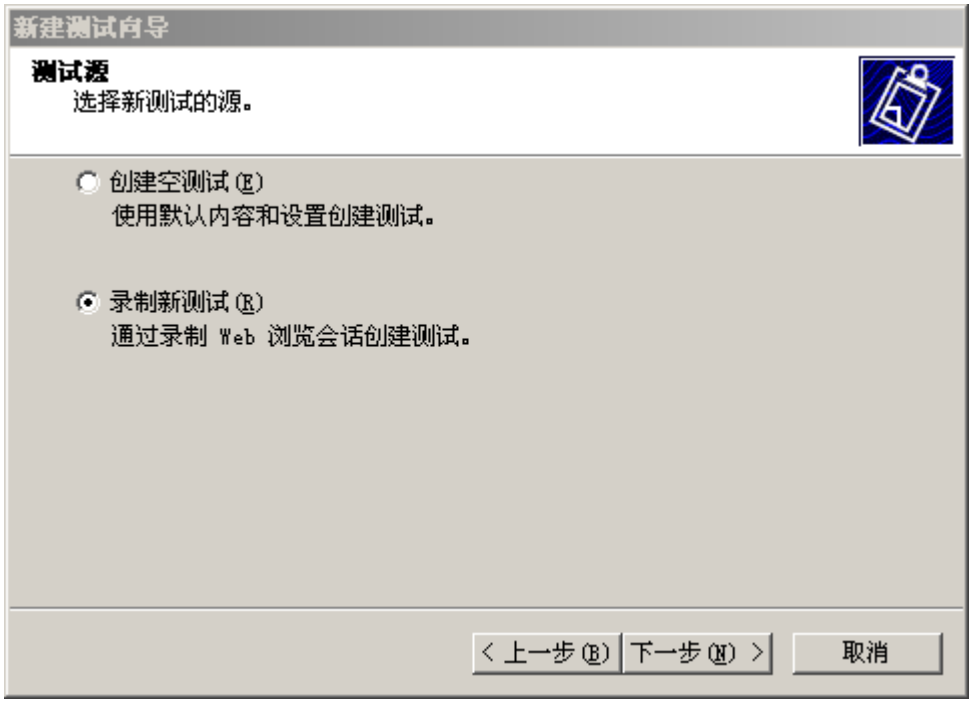
- 通过记录浏览器的活动
- 手工制作

在这一章中我们只介绍通过第一种方式。第五章我们将介绍第二种方式。

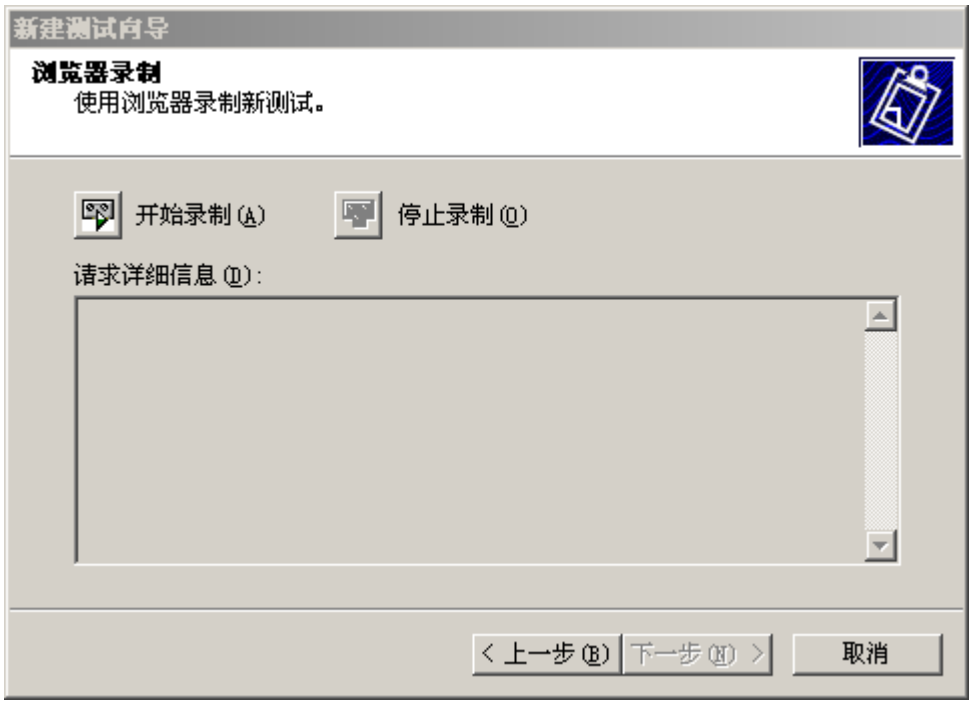
2.1 建立一个新的测试

点击菜单“ 操作 ”——“ 新建测试 ”

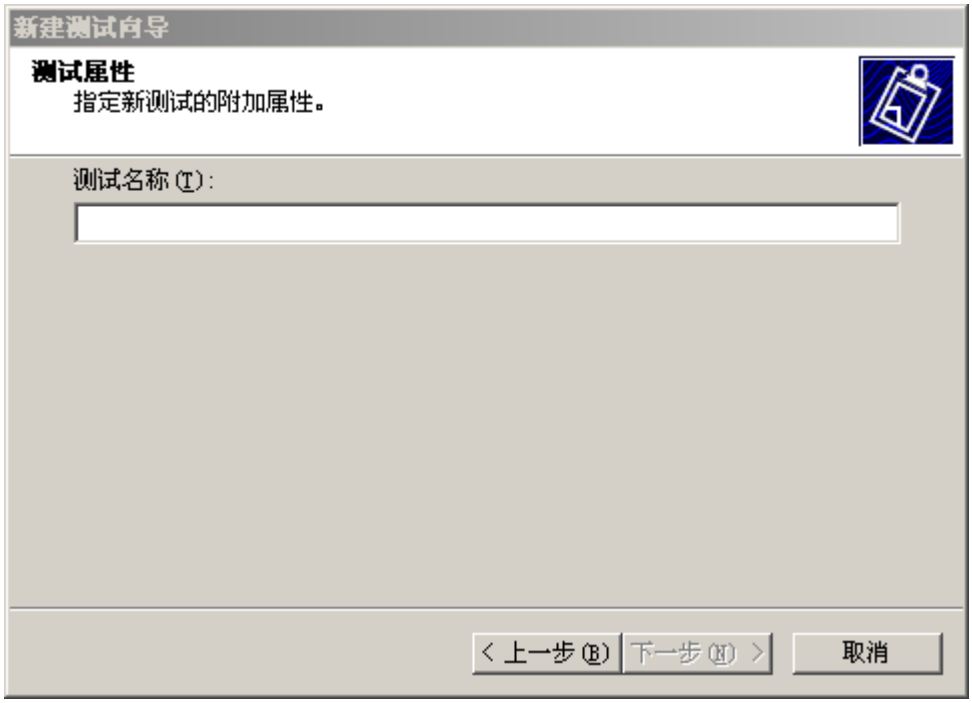
出现新建向导界面，直接下一步，然后选择“ 录制新测试 ”，见下图



然后选择脚本语言（只能选择 VBScript），进入下图时，点“ 开始录制 ”，ACT 同时出现 IE 窗口，在 IE 地址栏中输入要测试的站点的 URL，比如（http://192.168.6.199），然后开始操作。操作的过程可以看作是执行测试用例的过程。



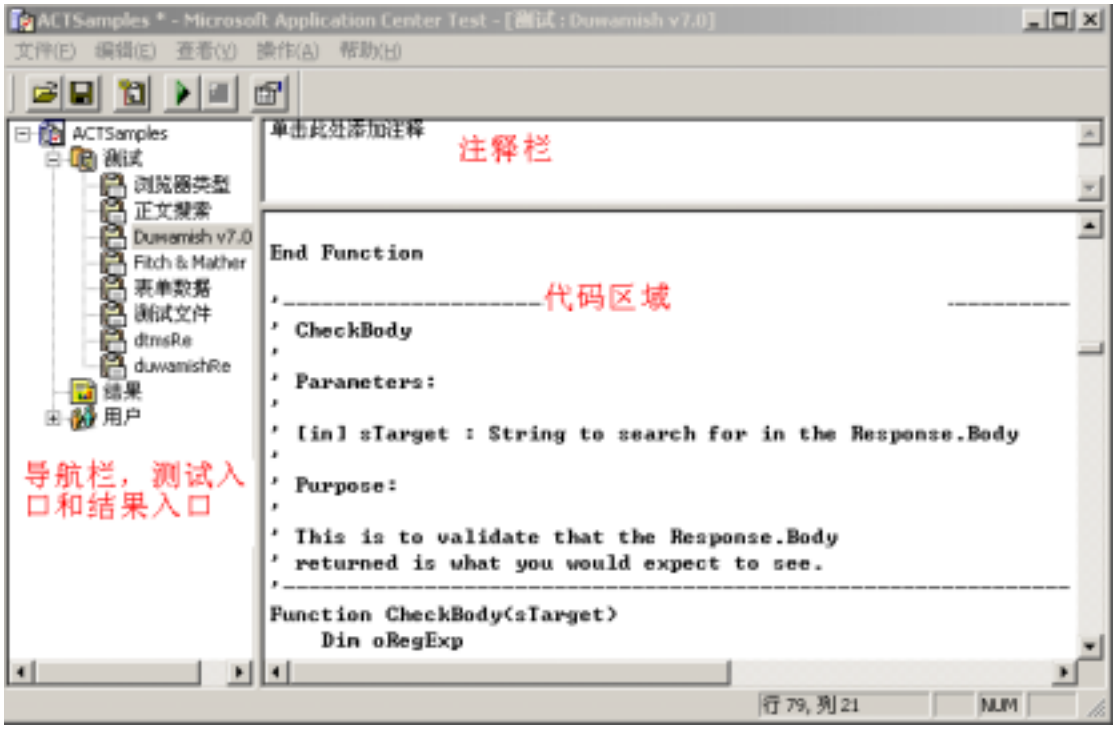
等测试用例执行完以后，点上图的“ 停止录制 ”按钮，输入测试的名称



下一步，即可录制完成测试。

2.2 编辑测试脚本

选中需要编辑的测试名称，其中的脚本是 ACT 是自动生成的。



在代码区域直接编辑，就像记事本一样，非常的方便。

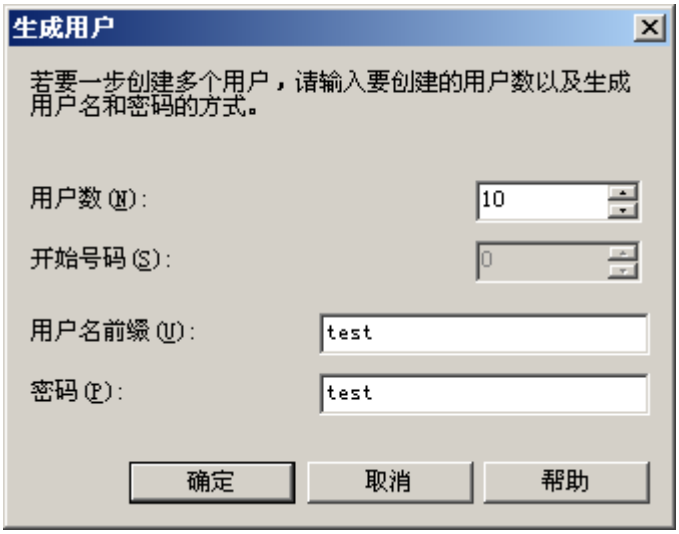
2.3 设置测试属性

2.3.1 为系统建立多用户

在需要身份验证的 Web 应用程序中，为了模拟多用户，我们需要事先建立 Web 系统中的用户。比如我们需要添加用户组 ABC 来测试 ABC 系统，可以这样：

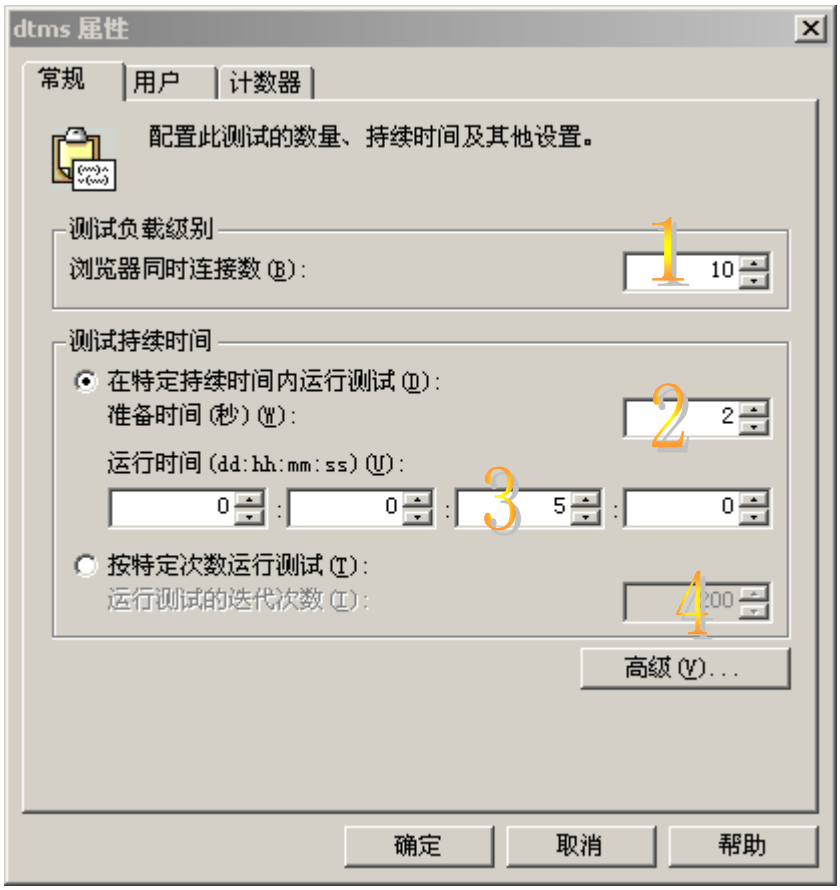
1. 在导航栏中选择用户，点右键，选择“ 添加 ”，出现“ 新建用户组 ”，选择“ 新建用户组 ”，点右键，选择 “ 重命名 ”，改成 “ A 系统用户组 ”

2. 在 A 系统用户组中生成用户：选择主菜单“ 操作 ”——“ 生成用户 ”，出现“ 生成用户 ”对话框，输入需要生成的用户数、用户前缀以及用户密码，确定，ACT 会生成 test[0-9] 的用户，所有用户密码都为 test，当然这些必须和 A 系统的用户一致。



2.3.2 设置属性

选择测试名称，右键，点“ 属性 ”，见下图



打开属性窗口的“常规”选项卡

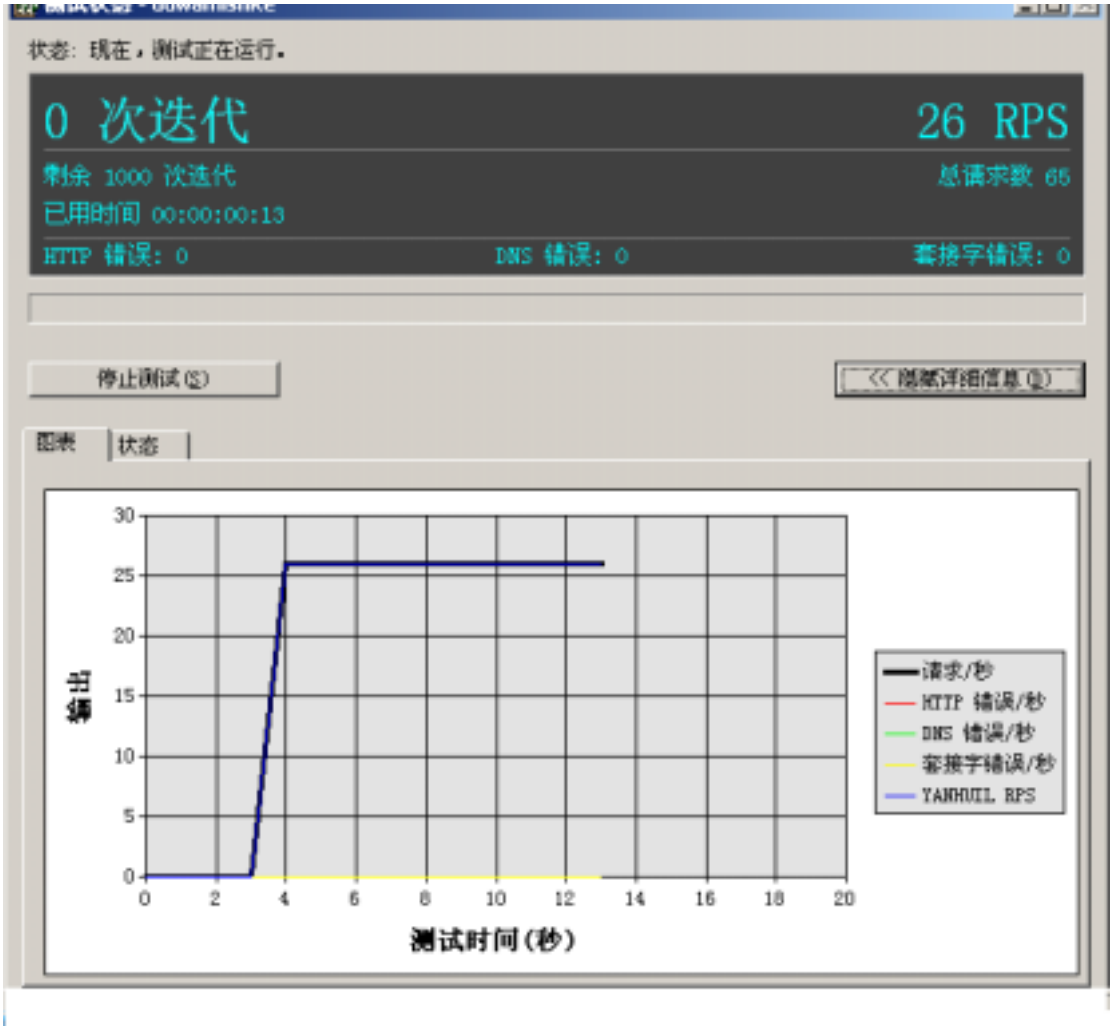
1. 设置测试的负载级别　浏览器同时连接数。在测试运行中，ACT 可以打开多个与 Web 服务器的连接，并可以在每个连接上发送请求。使用多个连接可以模拟同时有多个用户访问 Web 服务器的情况。
2. 准备时间：也就是我们在 WAS 文档中所说的“热身时间”，意思完全一样。在测试运行的前几秒内，Web 应用程序或服务器可能正在初始化组件或调整缓存数据。在准备时间内，ACT 不收集统计数据。
3. 运行时间：执行测试脚本要连续运行的时间，格式为：天数：小时数：分钟数：秒数。通常情况下，要进行持续的压力测试，运行时间至少 8 小时。
4. 迭代次数：在建立测试脚本的初期，我们常常为了验证测试脚本运行的正确性，通常需要执行运行的次数。

打开属性窗口的“用户”选项卡，可以选择在测试中选择需要的用户组，这里选择“A 系统用户组”。这里可以多选几个用户组，也可以让 ACT 自动生成用户（用于不需要进行登陆的系统）。关于自动生成用户和指定用户组的优缺点比较可以参考帮助中的“关于用户”。

打开属性窗口的“ 计数器 ”选项卡，可以添加性能计数器。关于计数器的作用以及添加计数器的必要性，可以参见 WAS 的使用文档。

2.4 执行测试

选择测试名称，右键，点“ 启动测试 ”，出现下图，执行测试页面



以上界面显示程序测试的时间、剩余时间、每秒提交的 Request 数量、提交的 Request 总量、错误数量等。点击按钮【Show details】可以查看详细数据。

3 理解测试结果

3.1 摘要

在导航栏中选择“ 结果 ”，在“ 测试运行 ”中选择测试运行名称，在“ 报告 ”中选择“ 概述 ”——“ 摘要 ”，这样就可以显示出测试结果的摘要信息。

在“ 摘要 ”中，包含以下元素：

- 简单说明
- 测试运行图形
- 属性
- 摘要
- 错误计数
- 网络统计数据
- 响应代码

3.2 性能计数器

在导航栏中选择“ 结果 ”，在“ 测试运行 ”中选择测试运行名称，在“ 报告 ”中选择“ 概述 ”——“ 计数器 ”，这样就可以显示出测试结果的计数器信息。

3.3 图表

在导航栏中选择“ 结果 ”，在“ 测试运行 ”中选择测试运行名称，在“ 报告 ”中选择“ 图表 ”，然后选择 X 轴和 Y 轴的代表的信息，这样就可以显示出图表信息。

ACT 可以使用测试运行过程中收集的数据创建图表。图表对于可视化分析结果以及找出数据的变化趋势很有帮助。可以将某个测试的多个报告甚至多个测试的报告中的结果叠加到一张图表中，这样可以查看一段时间内，对 Web 应用程序进行修改和调整时所引起的性能变化。

大多数分析方法都需要多次测试运行的结果。这些技术通常依赖于多次运行测试，通过逐渐增高负载级别来确定每秒的最大请求数或 Web 应用程序可以处理的最大同时连接。

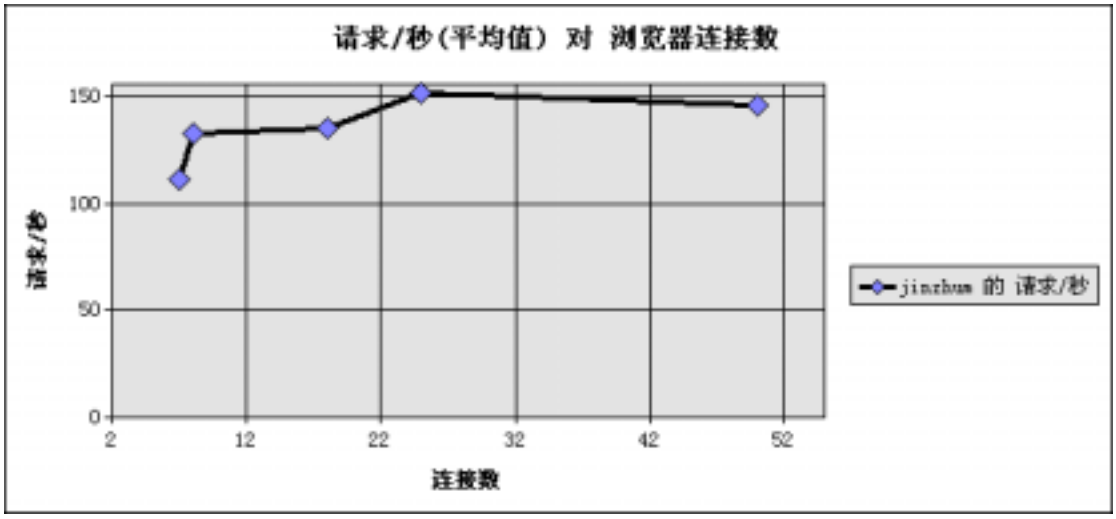
以下是一些常用图表值和比较的示例。

3.3.1 每秒连接数与请求数

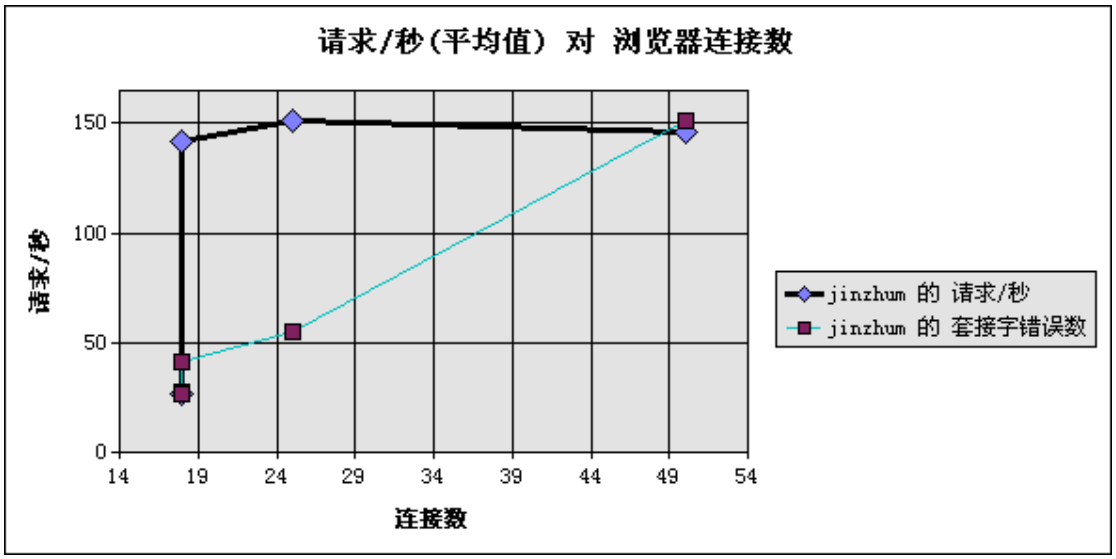
本图表有助于确定 Web 服务器可以处理的每秒最大请求数。

应该逐渐增加浏览器的同时连接数（如 1、2、5、10、20、50、100、500 和 1000）来多次运行测试。使用的最大数目可以大于也可以小于 1000。这取决于正在测试的 Web 服务器和 Web 应用程序。

完成所有测试运行之后，可以创建一张图表，在其横轴上绘制浏览器同时连接数，在竖轴上绘制每秒请求数（RPS）。



如果同时连接数很大，还可以显示测试运行过程中 TCP 错误的增长情况。因为可以在竖轴上绘制多重值，所以您可以在上面添加 TCP 错误，同时查看 TCP 错误和 RPS 值。图表刻度将反映“源”列表中当前选中的度量。



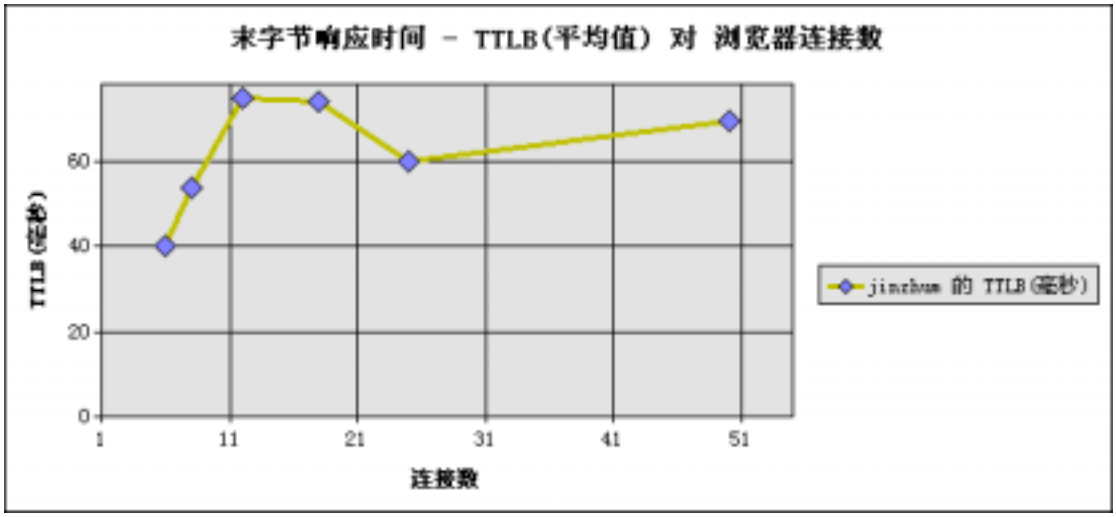
很多 Web 应用程序会增大 RPS 值直至某一特定值，在连接数超过了 Web 服务器可以处理的数量时，就会开始显示较低的 RPS 值。这样，就可以确定最佳每秒请求数对应的浏览器同时连接数。如果同时连接数超过了该最佳值，Web 站点每秒处理的请求数就会降低。

3.3.2 连接与接收最后一个字节的时间 (TTLB)

本图表有助于量化由于同时连接数的增长而引起的性能降低。

接收最后一个字节的时间 (TTLB) 值测量 Web 服务器响应流的最后部分到达用户 Web 浏览器所花费的时间。随着 TTLB 的增加，站点的速度会降低，并且对用户的响应也会减慢。

本图表中的数据也是通过针对每次测试运行逐渐增大同时连接数、并经多次运行测试而得到的。在图表横轴上绘制同时连接数，在竖轴上绘制 TTLB。

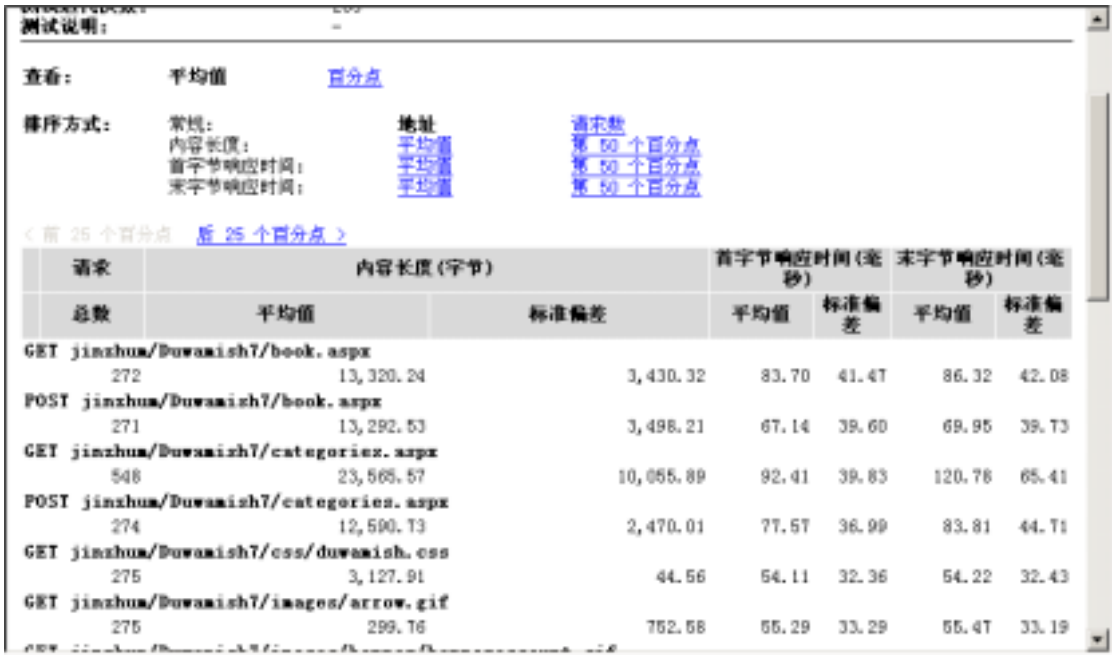


随着连接数的增加，TTLB 值通常会逐渐增长。对于 Web 站点来说，这意味着当用户数和同时连接数增长时，Web 服务器需要花费较长的时间来完成每次响应。

3.4 请求

3.4.1 摘要

显示了所有请求的平均 TTFB、TTLB 的值。通过分析每个请求的值，可以找出在相同的硬件软件环境下，很方便地地比较服务器上所有页面的性能，并找出速度最慢的页面。同时这些数据还可以用来用来计算页面的性能。



测试说明：

查看： 平均值 百分点

排序方式： 常规： 地址 请求数
内容长度： 平均值 第 50 个百分点
首字节响应时间： 平均值 第 50 个百分点
末字节响应时间： 平均值 第 50 个百分点

< 前 25 个百分点 后 25 个百分点 >

请求	内容长度(字节)			首字节响应时间(毫秒)		末字节响应时间(毫秒)	
	总数	平均值	标准偏差	平均值	标准偏差	平均值	标准偏差
GET jinzhun/Duwamish7/book.aspx							
	272	13,320.24	3,430.32	83.70	41.47	86.32	42.08
POST jinzhun/Duwamish7/book.aspx							
	271	13,292.53	3,498.21	67.14	39.60	69.95	39.73
GET jinzhun/Duwamish7/categories.aspx							
	548	23,565.57	10,055.89	92.41	39.83	120.78	65.41
POST jinzhun/Duwamish7/categories.aspx							
	274	12,590.73	2,470.01	77.57	36.99	83.81	44.71
GET jinzhun/Duwamish7/css/duwamish.css							
	275	3,127.91	44.56	54.11	32.36	54.22	32.43
GET jinzhun/Duwamish7/images/arrow.gif							
	275	299.76	752.58	55.29	33.29	55.47	33.19

4 录制脚本时可能会遇到的问题以及解决方法

4.1 录制脚本时如何设置多用户？

在需要身份验证的 Web 程序中，录制完成后，直接执行测试脚本，会发现多种连接用的都是在录制过程中的用户。

现在我们假设录制过程中，我们登陆用户名为 was1@mycom.com，密码为 tmptmp

1 为 Web 程序建立用户组（详细过程参考 2.3.1）

2 首先在“属性”设置“用户”选项卡中，指定特定的用户组（参考 2.3.2）。

3 编辑脚本

3.1 定义全局变量 Dim g_oUser

3.2 在脚本中查找到 was1@mycom.com 所在的过程，然后在该过程开始给 g_oUser 赋值 g_oUser = Test.GetCurrentUser

3.3 在脚本中查找到 was1@mycom.com ，确认就是登陆的用户名，用 “ & g_oUser.Name & “替换

3.4 在脚本中查找到 tmptmp，确认就是登陆的密码，用 ” & g_oUser.Password & “替换

这样就可以了。

5 使用 VBScript 编写 ACT 测试脚本

使用 VBScript 直接编写测试脚本，只要在 2.1 中直接“创建空测试”即可。

直接编写运行脚本，有以下好处：

- 脚本代码逻辑比较清楚，易维护
- 一旦 Web 程序界面、逻辑变化时，容易升级

缺点：

- 需要比较多的时间编写脚本
- 必须对 Web 程序的内部逻辑非常清楚
- 如果没有脚本编程的经验和不熟悉 HTTP 协议，比较难上手

下面将主要提炼出能够在大多数测试脚本中重用的代码

5.1 需要定义的全局变量以及初始化信息

```
Option Explicit
On Error Resume Next      " Comment out this line when debugging.

Dim g_oConnection
Dim g_oRequest
Dim g_oUser
Dim g_oResponse
Dim g_oCookies          'Web 应用使用 cookies 时才需要定义
Dim g_sPage
Dim g_HTTPServer
Dim g_HTTPPath
Dim g_HTTPPort
Dim g_UseSSL
Dim g_HTTPVersion
Dim g_UseKeepAlive
Dim g_ViewState

'-------------------------------------------------------------------------
' 根据实际配置情况，初始化以下全局变量的值
'-------------------------------------------------------------------------
g_HTTPServer      = "localhost"          ‘需要测试的 Web Server 名称或者 IP
g_HTTPPath        = "/duwamish7/"        ‘需要测试的虚拟目录的名字
g_HTTPVersion     = "HTTP/1.1"           ‘一般情况下不需要更改
g_HTTPPort        = 80                   ‘使用的端口号，默认是 80，一般不需要更改
g_UseSSL          = False                ‘是否使用了加密的安全连接。使用了请设成 True
g_UseKeepAlive    = True                 ‘？？？？？一般情况下都设置成 True

'-------------------------------------------------------------------------
' 初始化测试过程中可能发生的各种情况
'-------------------------------------------------------------------------
Const L_ErrHomePage_Text          = "Error: Page not found. Make sure site is configured and setup properly."
Const L_ErrRequest_Text           = "Error: Invalid request or host not found."
Const L_ErrConnect_Text           = "Error: Unable to create connection."
Const L_ErrResponse_Text          = "Error: Request failed for page: "
Const L_ErrPageNotFound_Text      = "Error: Page not found: "
```

```
Const L_NotLoggedIn_Text            = "not logged in successfully. Make sure proper users are setup."


'-------------------------------------------------------------------------
' Web Pages，列出本次测试要访问到的所有的网页的文件名
'-------------------------------------------------------------------------
Const BOOK_PAGE                 = "book.aspx"
Const CATEGORIES_PAGE       = "categories.aspx"
Const DEFAULT_PAGE              = "default.aspx"
Const SHOPPINGCART_PAGE    = "shoppingcart.aspx"
Const SEARCHRESULTS_PAGE = "searchresults.aspx"
Const ACCOUNT_PAGE             = "/secure/account.aspx"
Const LOGON_PAGE                = "/secure/logon.aspx"
Const CHECKOUT_PAGE           = "/secure/checkout.aspx"
Const ORDER_PAGE                = "/secure/order.aspx"
```

5.2 把测试过程日志写入 ACTTrace.LOG 文件

```
'-------------------------------------------------------------------
' ActTrace
'
' 参数说明
'
' [in] strText : 要写入日志文件的字符串.
'
' 功能
'
' 把字符串写入 ACTTrace.LOG 日志文件.
'-------------------------------------------------------------------
Sub ActTrace(strText)
    Test.Trace strText & vbCrLF
End Sub
```

5.3 判断 HTTP 响应是否成功

```
'-------------------------------------------------------------------
' IsSuccessful
'
' 参数说明:
'
' [in] g_oResponse : Response 对象
'
' 功能:
```

```
'
' 检查服务器的返回代码，验证请求是否成功
'--------------------------------------------------------------------
Function IsSuccessful(g_oResponse)
    Dim lStatusCode : lStatusCode = 0
    Dim bIsSuccessful

    lStatusCode = g_oResponse.ResultCode

    bIsSuccessful = ((lStatusCode >= 200) And (lStatusCode <= 299))

    If Not bIsSuccessful Then
        ActTrace L_ErrResponse_Text & g_oResponse.Path & " : " & lStatusCode
    End If

    IsSuccessful = bIsSuccessful

End Function
```

5.4 产生一个请求（Request）

```
'-----------------------------------------------------
' PopulateRequest
'
' 功能:
'
' 产生一个请求对象，并且已经初始化成默认信息
'-----------------------------------------------------
Sub PopulateRequest()
    Set g_oRequest = Test.CreateRequest
    g_oRequest.HTTPVersion = g_HTTPVersion
    g_oRequest.ResponseBufferSize = 32768    ' 设定请求对象的缓冲区大小,可以不设定

    ' If g_UseKeepAlive is True then use Keep-Alive else Keep-Alive is off.
    If g_UseKeepAlive = True Then
        Call g_oRequest.Headers.Add("Connection", "Keep-Alive")
    End If
End Sub
```

5.5 设定_VIEWSTATE 的值（只针对 ASP.NET 开发的 WEB 程序）

```
'--------------------------------------------------------------------
' SetViewState
'
' 参数说明:
'
' [in]   g_oResponse : The Response 对象
'
' 功能:
'
' 在 Response.Body 正文中查找 __VIEWSTATE 并且设定它的值.
'--------------------------------------------------------------------
Sub SetViewState(g_oResponse)
    Dim Pos, PosStart, PosEnd
    Dim res, vState

    If (g_oResponse Is Nothing) Then
        ActTrace L_ErrRequest_Text
    Else
        Pos = InStr(g_oResponse.Body, "__VIEWSTATE")
        If Pos > 0 Then
            PosStart = InStr(Pos, g_oResponse.Body, "value=""")
            PosStart = PosStart + Len ("value=""")

            PosEnd = InStr(PosStart, g_oResponse.Body, """")

            res = Mid(g_oResponse.Body, PosStart, PosEnd - PosStart)
            Test.SetGlobalVariable "vState",res
            g_ViewState = Test.GetGlobalVariable("vState")
        End If
    End If
End Sub
```

5.6 检查 Response.Body 是否包含指定的字符串

```
'--------------------------------------------------------------------
' CheckBody
'
'参数:
'
```

```
' [in] sTarget : 要在 Response.Body 中查找的字符串
'
' 功能:
'
' 验证返回的 Response.Body 中是否包含 sTarget 字符串
'---------------------------------------------------------------------
Function CheckBody(sTarget)
    Dim oRegExp
    Dim oMatches
    Dim sBody
    Dim bFoundTarget

    bFoundTarget = False

    sBody = g_oResponse.Body

    Set oRegExp = New RegExp
    oRegExp.Pattern = sTarget

    ' run the search
    Set oMatches = oRegExp.Execute(sBody)

    If oMatches.Count > 0 Then
        bFoundTarget = True
    Else
        ActTrace L_ErrPageNotFound_Text & " " & sTarget
    End If

    CheckBody = bFoundTarget
End Function
```

5.7 请求指定的网页名

```
'---------------------------------------------------------------------
' GetPage
'
' 参数说明:
'
' [in] sGetPage      : 要请求的网页名
' [in] sPageString :    在请求的网页中需要验证的关键字
'             '
' 功能:
'
' 请求指定的网页 sGetPage，并且验证该网页中是否包含 sPageString 字符串
```

```
'---------------------------------------------------------------------
Sub GetPage(sGetPage, sPageString)
    Call PopulateRequest()

    g_sPage = g_HTTPPath & sGetPage
    g_oRequest.Path = g_sPage
    g_oRequest.Verb = "GET"

    Set g_oResponse = g_oConnection.Send(g_oRequest)
    If (g_oResponse Is Nothing) Then
        ActTrace L_ErrRequest_Text
    Else
        If IsSuccessful(g_oResponse) Then
            Call SetViewState(g_oResponse)
            If Not CheckBody(sPageString) Then
                ActTrace L_ErrPageNotFound_Text & " " & sPageString
            End If
        End If
    End If

    Set g_oRequest = Nothing
End Sub
```

6 ACT 与 WAS 的比较

6.1 相同点

1 在功能是都可以对 Web 程序进行性能测试和压力测试
2 提供报表的数据种类都是一样的
3 都是微软的产品，同时又都是微软推荐的产品

6.2 不同点

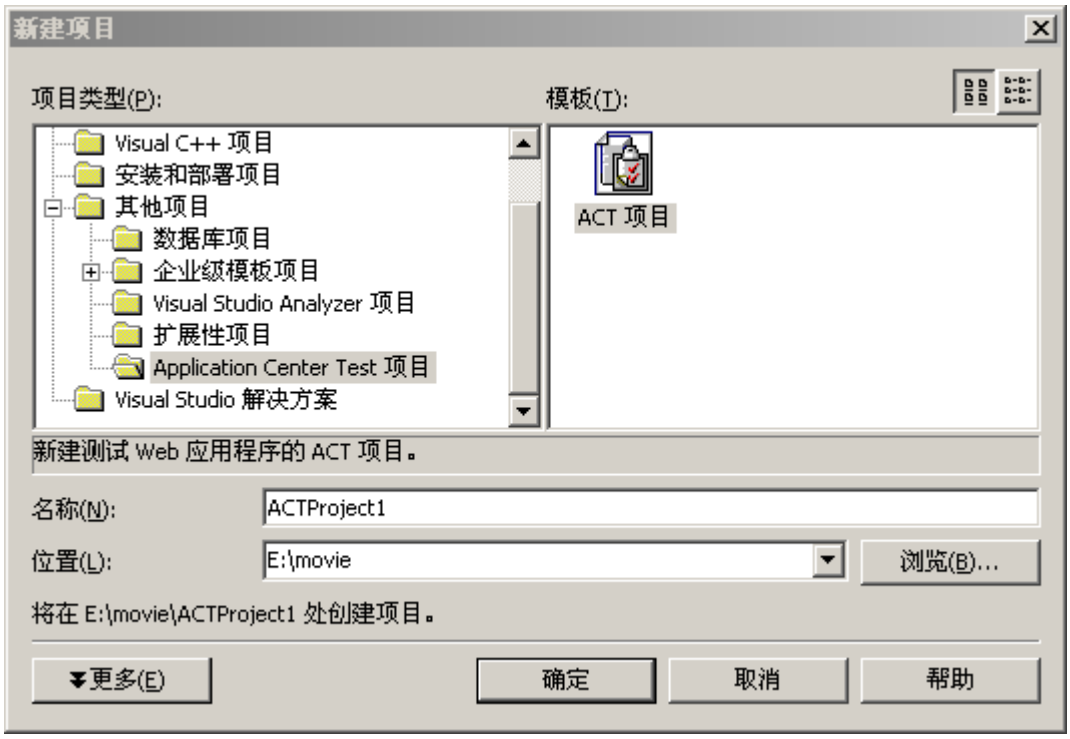
1. ACT 产生的报表是 HTML，其中包含非常直观的图表，同时可以根据需要定制符合自己要求的图表，并且对不同设置的测试结果可以进行非常直观的比较；WAS 产生的是比较普通的文本文件，不方便比较，也不够直观。
2. WAS 支持在测试运行过程中使用不限数量的协同客户端，这样可以充分利用分布式计算的能力。ACT 的 VS.NET 版本就不具备这种功能。如果要多台计算机共同工作，就必须在每一台计算上运行 ACT 的脚本。如果在一台计算机上模拟很多连接，ACT 本身就会有很多线程，占用很多 CPU 和网络资源，可能会导致测试结果的不准确。而 WAS 会自动把很多线程均匀分配到不同的计算机上，更加符合实际

的环境。

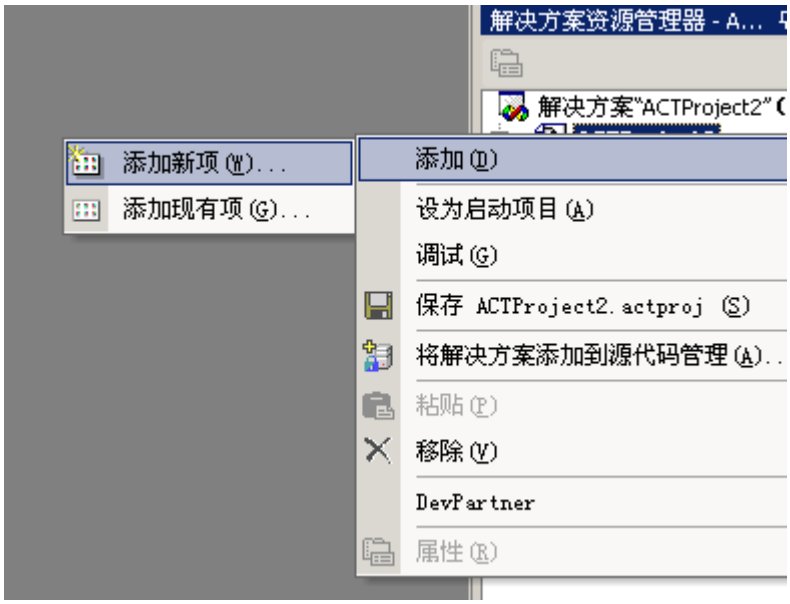
3. WAS 有导入 IIS 日志的功能，ACT VS.NET 版本则没有这个功能。该功能在测试一个已经投入试运行或者软件的升级版本的 Web 工程时会非常的有用。因为它记录的就是真实用户的实际操作记录。
4. WAS 目前的版本是 1.1.288.1 在微软社区中，微软内部的人员说 WAS 已经不再更新了（事实上 WAS 当前的版本感觉也还只是试用版，因为帮助中有非常明显的错误）。而 ACT 可能就是用来替代 WAS 的工具，只不过 VS.NET 中附带的 ACT 功能作了限制。
5. ACT 可以和 VS.NET 开发环境集成，可以不退出开发环境直接运行 ACT 测试脚本；WAS 不能和 VS.NET 开发环境集成

7 在 VS.NET IDE 中使用 ACT

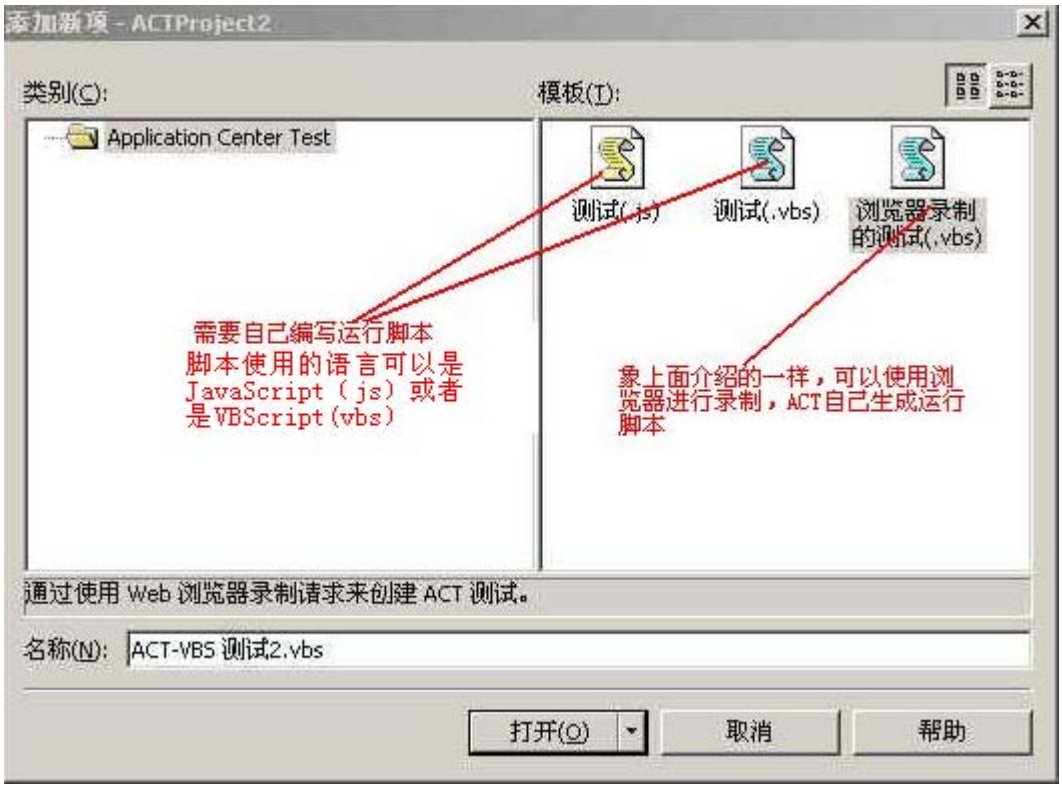
在当前的解决方案中添加 ACT 项目，见下图



然后，添加新项，如下操作



出现以下窗口，可以选择不同的脚本语言（VBS 或者 JS）



添加完后，其他编辑，运行操作可以参考前面的内容，这里不再重复说明。