LoadRunner 测试 BOSS 技巧

——山东移动 BOSS1.5

修订纪录

日期	修订版本	修改描述	作者
2005-10-30	V1.0	初稿完成	徐林林
2007-11-18	V1.1	修订与补充运行原理图以及组件关系图	徐林林

1. 概述

在山东 BOSS 性能压力测试过程中，发现脚本对于整个压力测试过程的重要性，一个压力测试脚本录制和编辑修改得怎么样直接影响后面压力测试的执行。通常情况下，脚本应尽可能的精简，就像写代码一样。针对 BOSS 系统的特点，个人认为把单一业务录制成一个 Action,并在脚本中添加 Transaction，Find 检查（可以采用 URL-based script 方式录制并事先设定），Rendezvous，参数化等基本元素，然而有时我们会发现光有这些基本元素还不能满足我们的要求。比如在 Controller 中运行我们的脚本时，一旦压力过大或某种原因导致某一业务失败，而此时我们很想尽快地找出错误的原因。当然此时我们第一想到的是，查找日志,但是有时发现查找日志很不方便,因此我们希望寻求一种更快捷的方式，希望能直接从 Controller 的 Errors 错误中找到出错的服务号码、在第几次 Iteration 的哪个 Transaction 出错。实现的方式，当然是通过简单的编程来调用错误日志里的信息，另外本文中还简单介绍了关于 LoadRunner 工具使用的一些常用注意事项、脚本处理技巧和一些常用性能参数的分析及性能测试中机器瓶颈的定义和查看机器瓶颈的相关命令。
下面再具体的一一介绍：

2. 一个规范的性能测试脚本就像一段规范的程序代码一样，需要基本的说明信息：

在下面要介绍的脚本中，我把这些信息以注释的形式放在 vuser_init 最前面：

```
/*
@Athour:徐林林
@Date:2005-09-18
@Name:异地缴费压力测试脚本
@Parameter:BOSSURL,LogName,PhoneNum,iteration,FanHui
@Data:BOSSURL:BOSSURL.dat;        //由于 BOSS 压力测试前台展现环境多,故将地址也参数化。
      LogName:LogName.dat;        //登录用操作员，选择具备异地缴费权限的操作员，这里选择的是德州操作员 300 个。
      PhoneNum:PhoneNum.dat;      //用于异地缴费的服务号码，这里选择的是烟台的正常在用的标准全球通号码 3000 个。
      iteration:iteration.dat;    //用于压力测试出错时，打印出错所在的循环次数。
@Description:此脚本用于测试异地缴费的性能及稳定性，选用德州的操作员对烟台的标准全球通号码进行异地缴费,目标是通过 vuser 模仿真实操作员进行异地缴费，达到验证或测试系统性能和稳定性的目的。
@Notes:脚本的录制使用的是 LoadRunner8.0 的 VU,采用的是 URL-based script
```

方式，需要特别注意的是 Recording Options(按 Ctrl+F7)的 Advanced 选项里的 Surport Charset 一般情况默认为不选，除非字符集合采用的是国际标准才选中 UTF-8 选项，否则会出现汉字乱码现象。
*/

3．通常情况下，任何业务必须在登陆成功后才能做，所以有必要对登陆成功与否进行判断：

下面我从脚本中取出相关部分进行简单介绍：

```
vuser_init()
{
    int status; //定义变量用于判断登陆是否成功
   web_reg_find("Text=山东移动 BOSS",
      LAST);
    …….
    …….
   web_submit_data("reguserAction.do",  //登陆提交数据 Action。
      "Action=http://{BOSSURL}/boss/reguserAction.do",
      "Method=POST",
      "RecContentType=text/html",
      "Referer=http://{BOSSURL}/boss/index.jsp",
      "Snapshot=t12.inf",
      "Mode=HTTP",
      ITEMDATA,
      "Name=logname", "Value={LogName}", ENDITEM,
      "Name=password", "Value=", ENDITEM,
      LAST);
    status = web_submit_data("reguserAction.do",    // 取成功与否标志
      "Action=http://{BOSSURL}/boss/reguserAction.do",
      "Method=POST",
      "RecContentType=text/html",
      "Referer=http://{BOSSURL}/boss/index.jsp",
      "Snapshot=t12.inf",
      "Mode=HTTP",
      ITEMDATA,
      "Name=logname", "Value={LogName}", ENDITEM,
      "Name=password", "Value=", ENDITEM,
      LAST);

    if (status == LR_FAIL)      //一旦登陆失败，脚本给出提示报错信息。
      {
```

```
        lr_error_message("错误信息: %s", "不能正常登陆！");
        return -1;
        }
}
```

4．事务的定义，很简单，也很有必要，尽量是每个定义的事物符合逻辑和小。

在下面的脚本中，在异地缴费这一业务中定义了两个 Transaction：准备异地缴费数据和提交异地缴费，见如下脚本代码：

```
lr_start_transaction("准备异地缴费数据");


web_set_max_html_param_len("4096");
……….
web_submit_data("chargeacc.do",

"Action=http://{BOSSURL}/boss/charge/commonbusiness/acccharge/chargeacc.do?act=queryaccount",
        "Method=POST",
        "RecContentType=text/html",

"Referer=http://{BOSSURL}/boss/charge/commonbusiness/acccharge/acccharge.jsp?act=first",
        "Snapshot=t74.inf",
        "Mode=HTTP",
        ITEMDATA,
        "Name=isconfirm", "Value=no", ENDITEM,
        "Name=chargetype", "Value=telnumber", ENDITEM,
        "Name=telnumber", "Value={PhoneNum}", ENDITEM,
        "Name=nowfee", "Value=0.0", ENDITEM,
        "Name=factfee", "Value=", ENDITEM,
        "Name=totalfee", "Value=0.0", ENDITEM,
        LAST);
lr_end_transaction("准备异地缴费数据", LR_AUTO);
```

5．增强脚本，对脚本进行简单的编程。

增强脚本，对脚本进行简单的编程，为性能或压力测试提供方便，这也是写本文的宗旨，下面对此做简单的介绍：

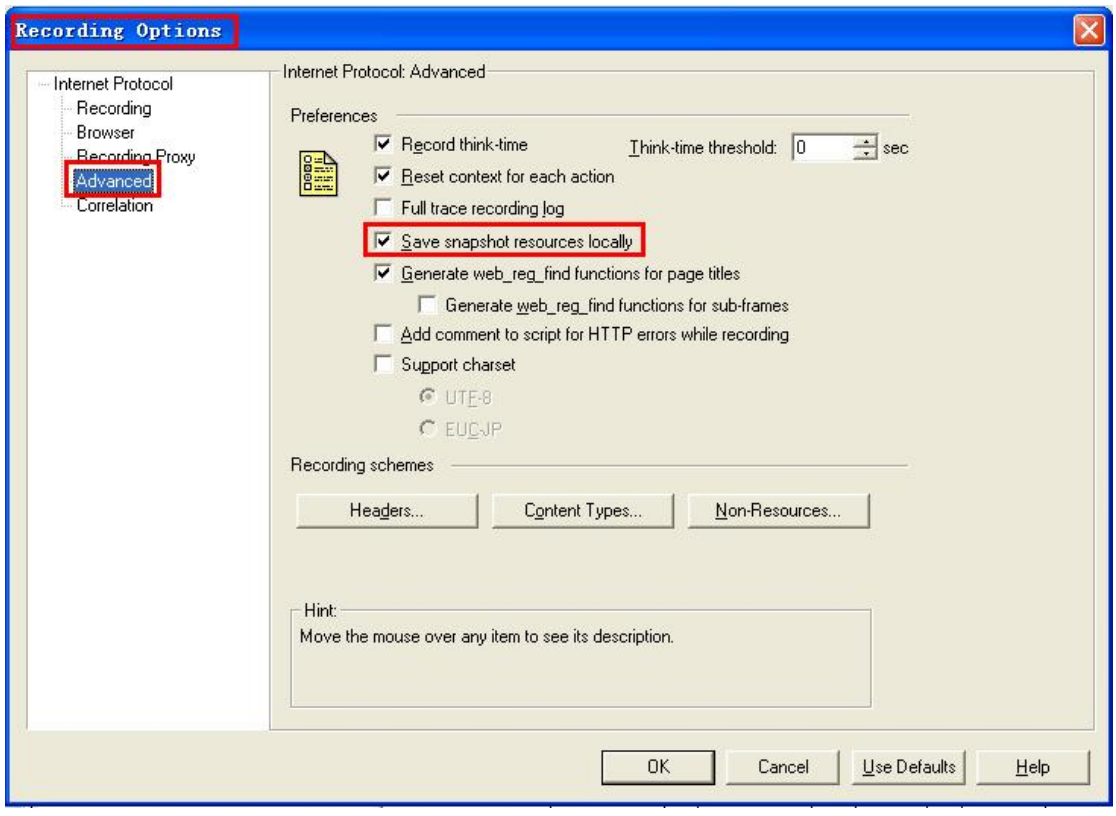
5．1 首先，定义成功与否的判断标志或字符串。

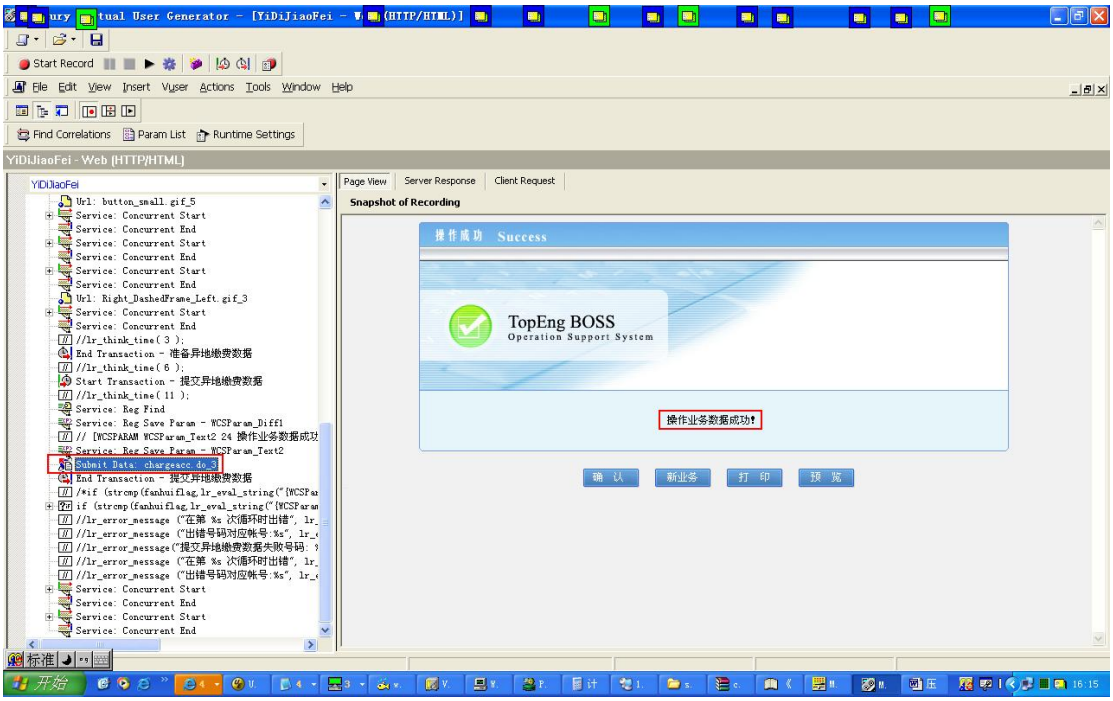
在此，我把判断成功与否的标志定义在异地缴费 Action 最前面，具体定义如下：char fanhuiflag[30]="操作业务数据成功！";

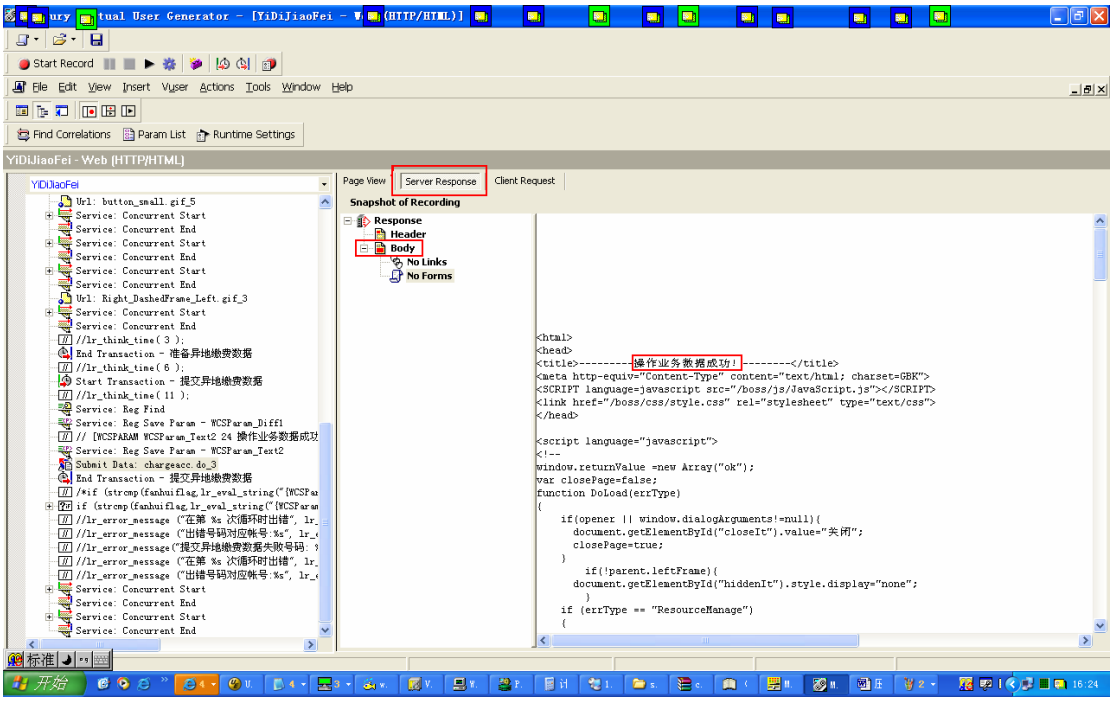
但是大家可能会问，字符串"操作业务数据成功！"从何处而来，可以肯定的不能凭空想象，成功标志可从两三种方式来取得：

第一种:也是最简单的一种,直接从脚本中取得,具体操作是以 View Tree 方式找到相关的界面,然后从 Server Response 的 Snapshot 的 Body 里去取。见下面的图片：

注：Snapshot 在录制前要将 Recording Options>Advanced 里的 Save snapshot resources locally 选项选中。



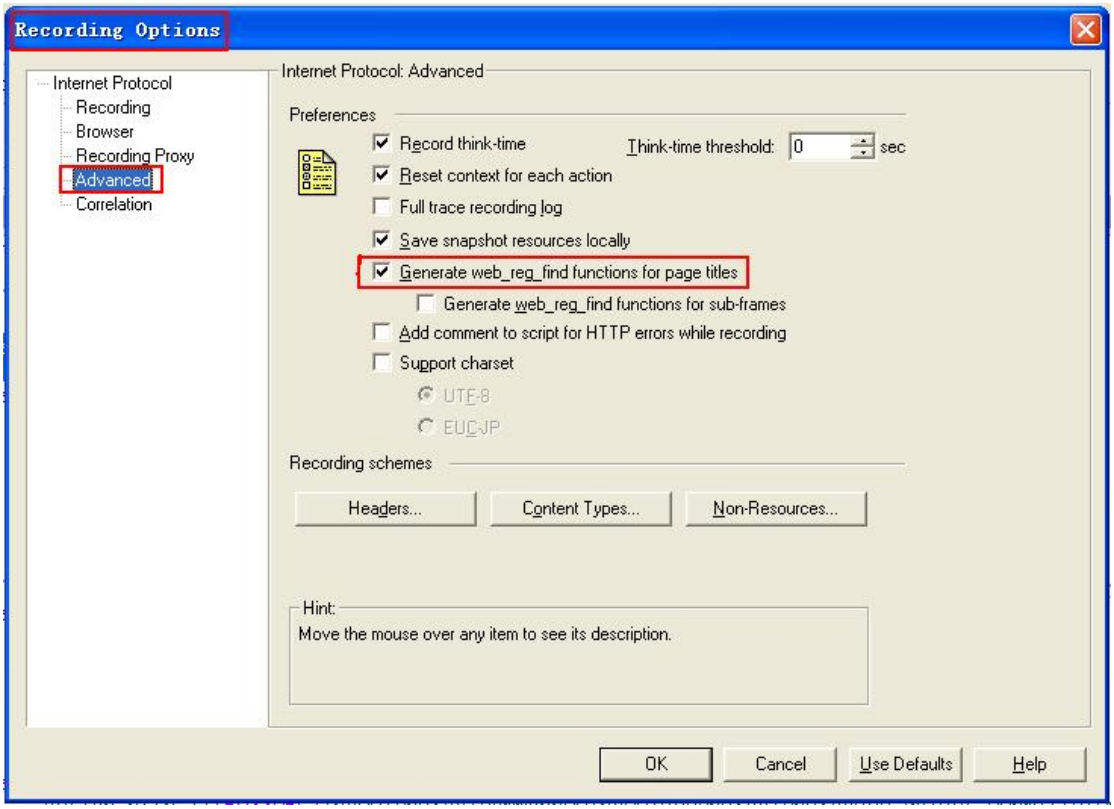




第二种方式，从脚本代码中去取，即取 find 函数中相关字符串，具体做法是，找到在提交事件前的 web_reg_find 函数，然后从中取相关字符串。

```
web_reg_find("Text=---------操作业务数据成功！--------",
LAST);
```

值得注意的是要有 web_reg_find 函数，可以在录制前选中 Recording Options>Advanced 里的 Generate web_reg_find functions for page titles 选项，见下图所示：



第三种方式，从本地的 snapshot 里去取，具体操作，首先找到提交数据事件相关脚本，找到 snapshot 文件的名称，然后从本地的 data 文件里去找这个 snapshot 文件，然后从中找到我们需要的字符串。

```
    web_reg_find("Text=---------操作业务数据成功！--------",
        LAST);
    ……
     web_submit_data("chargeacc.do_3",

    "Action=http://{BOSSURL}/boss/charge/commonbusiness/acccharge/chargeacc.do?act=submit&atype=commitdata",
        "Method=POST",
        "RecContentType=text/html",

    "Referer=http://{BOSSURL}/boss/charge/commonbusiness/acccharge/chargeacc.do?act=querycustomer",
        "Snapshot=t129.inf",
        "Mode=HTTP",
        ITEMDATA,
        "Name=isconfirm", "Value=no", ENDITEM,
        "Name=chargetype", "Value=telnumber", ENDITEM,
        "Name=telnumber", "Value=", ENDITEM,
        "Name=nowfee", "Value=8.8", ENDITEM,
        "Name=factfee", "Value=0.00", ENDITEM,
```

```
"Name=totalfee", "Value=8.8", ENDITEM,
"Name=accountno", "Value={WCSParam_Diff1}", ENDITEM,
"Name=factpay", "Value=8.8", ENDITEM,
"Name=grantpercent", "Value=", ENDITEM,
"Name=grantfee", "Value=0", ENDITEM,
"Name=takecash", "Value=8.8", ENDITEM,
"Name=zero", "Value=0", ENDITEM,
"Name=paytype", "Value=Cash", ENDITEM,
"Name=remark", "Value=", ENDITEM,
"Name=invoice", "Value=joininvoice", ENDITEM,
LAST);
```

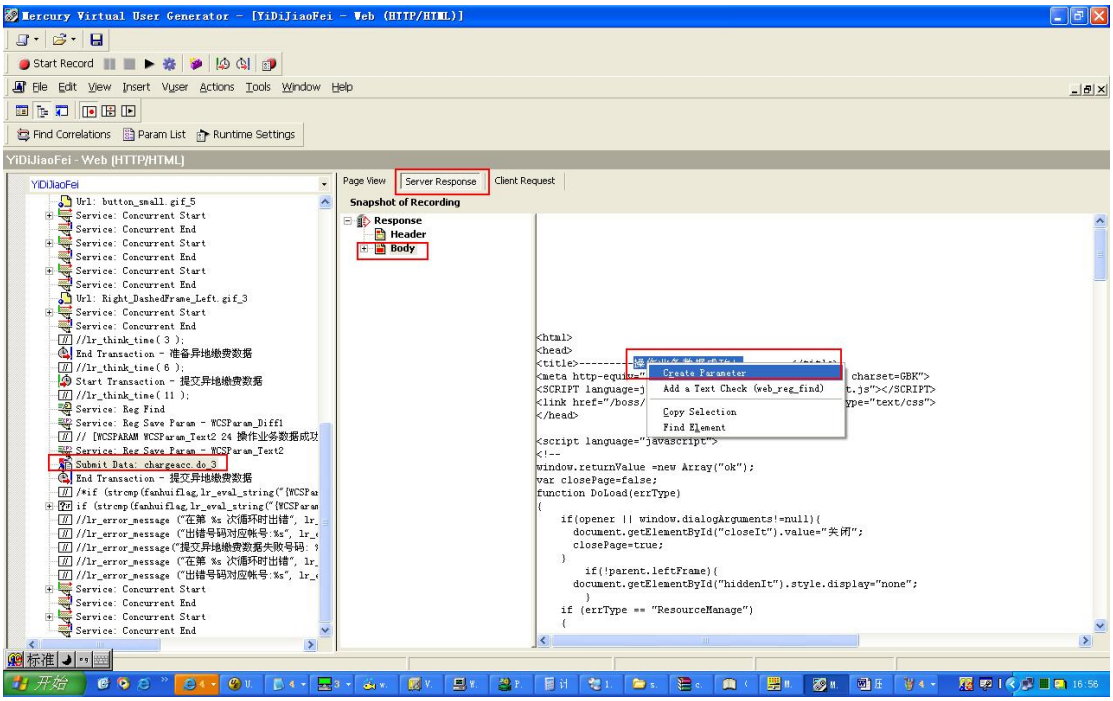
5．2 设置和保存判断成功与否的参数，这也是最关键的地方。

有一点大家都很清楚，业务成功返回的字符串和失败返回的字符串是不同的，我们所要做的是将返回的字符串做为参数保存下来，然后拿这个参数和我们事先定义好的成功的标志做比较，有两种方式可以设置和保存这一参数，下面简单介绍：

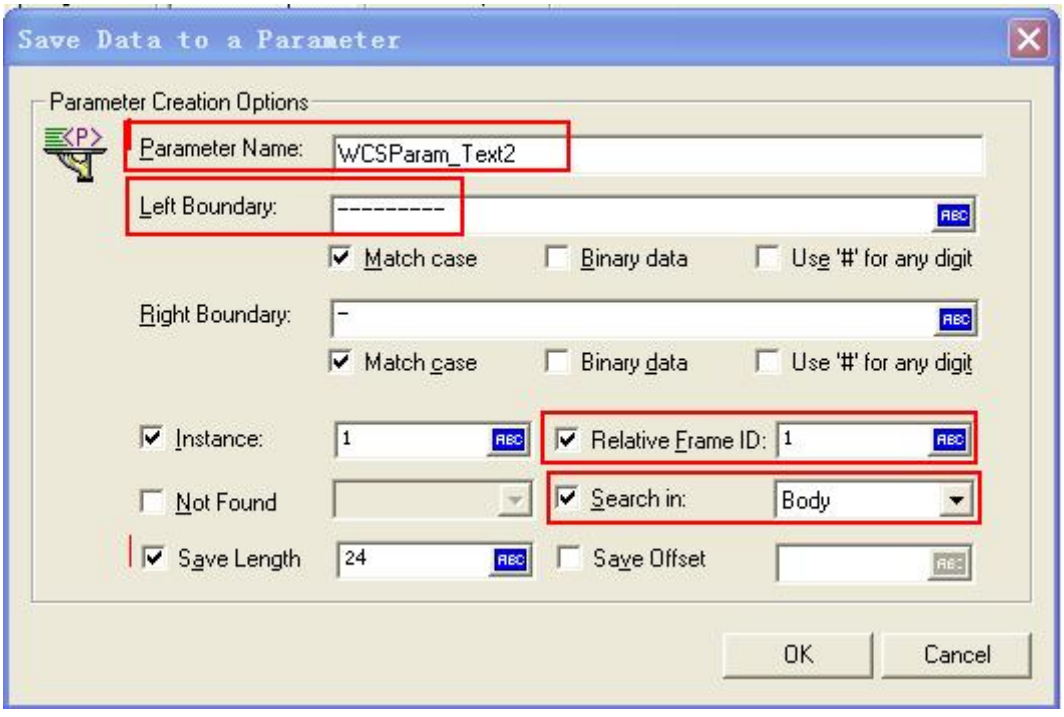
第一种方式也是最准确的方式，是以 View Tree 方式找到相关的界面,然后从 Server Response 的 Snapshot 的 Body 里的成功的返回标志，然后对其进行 Create Parameter,这样 LoadRunner 会自动在脚本中添加 **web_reg_save_param** 函数，具体如下：

```
    // [WCSPARAM WCSParam_Text2 24 操作业务数据成功！ _Text2]
Parameter {WCSParam_Text2} created by Correlation Studio
    web_reg_save_param("WCSParam_Text2",
        "LB=---------",
        "RB=-",
        "Ord=1",
        "RelFrameId=1",
        "Search=Body",
        LAST);
```

操作流程见下图：



第二种方式是在提交事件前添加 web_reg_save_param 函数，具体操作是在提交事件前右击鼠标选择 Insert Before 选项，然后在弹出的对话框中选择 Services 里的 web_reg_save_param 选项，单击 OK 按钮，然后在弹出的对话框中输入相关的数据，见下图：



5．3 编写相关判断代码段。

在已经定义好判断字符串和设置和保存好成功与否的标志字符串参数后，编写相关判断代码段，这也是最关键的地方，具体代码段如下：

```
if (strcmp(fanhuiflag,lr_eval_string("{WCSParam_Text2}"))!=0)
  {
  lr_error_message("消息: %s,在第 %s 次循环时出错,出错号码:%s", "提交异地缴费数据失败!",  lr_eval_string("{iteration}"), lr_eval_string("{PhoneNum}"));
  }
```

简单解释如下：

fanhuiflag：前面已经定义好的成功标志字符串的数组名，当然前面也可以用指针来实现，这里不做介绍。

WCSParam_Text2：为实现设置和保存好的成功与否返回的字符串的参数；

PhoneNum：服务号码的参数化，具体为电话号码。关于参数化，这里不做分析和解释部分。

Iteration：为了定位具体的循环而设置的参数。

Strcmp 函数：LoadRunner 自带的字符串比较函数，相等时返回 0

lr_eval_string 函数：LoadRunner 自带求字符串函数，函数格式为

char * lr_eval_string (const char * *instring*);

（另一种判断方式：先事先定义好 int rc=1; char *fanhuiflag="操作业务数据成功！"; 然后在定义事务的结尾进行判断：

```
rc=strcmp(str_tip,lr_eval_string("{re_str_tip}"));
    if(rc==0)
      {
        lr_end_transaction("异地缴费_提交", LR_PASS);
      }
    else
      {
         lr_error_message("异地缴费_提交失败，号码为:%s",lr_eval_string("{msisdn}"));
         lr_end_transaction("异地缴费_提交", LR_FAIL);
      }
   //lr_end_transaction("异地缴费_提交",LR_AUTO);)
```

5．4 验证我们的需求是否实现

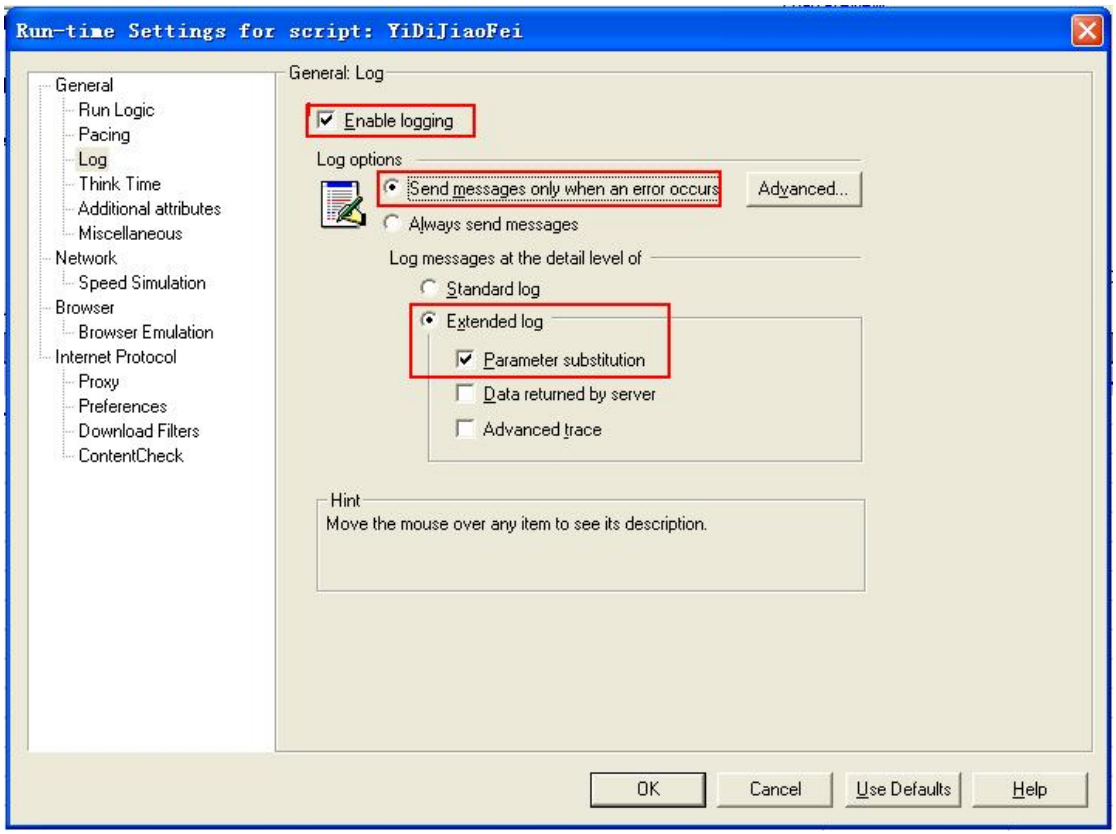
下面简单介绍，整个验证过程，在确保脚本正确和测试环境正常的情况下，我们先在 VU 里验证下是否真正实现了我们想要的功能，为了方便，我特地将判断条件该为==而不是!=，通过查看日志来检查。

首先，选中菜单栏里的 Run-time Settings 子菜单，设置里面的 Log 选项，选

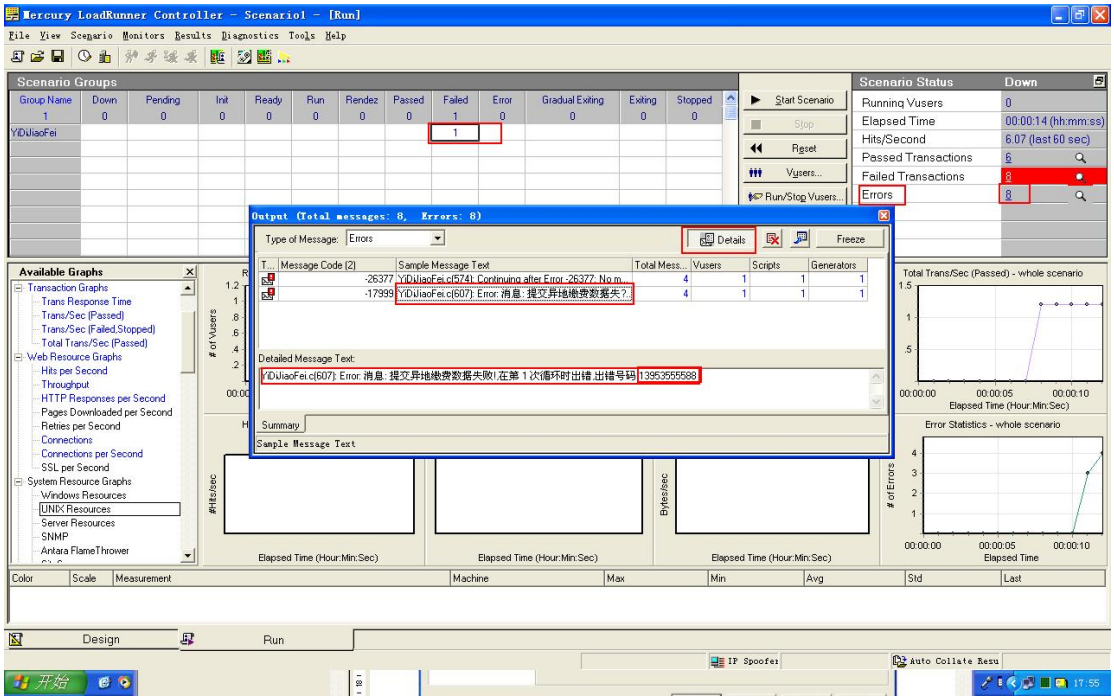
中 Enable Logging 和 Always send messages 和 Extended log 及 Parameter substitution。
设置 Run Logic 里的 Number of Iterations 为 2，然后运行，并查看日志。可以得到如下的日志（只取了关键部分的）：
第一次循环关键部分：
YiDiJiaoFei.c(598): Notify: Transaction "提交异地缴费数据" ended with "Fail" status (Duration: 4.8459 Wasted Time: 0.0060).
YiDiJiaoFei.c(605): Notify: Parameter Substitution: parameter "WCSParam_Text2" = "操作业务数据成功！"
YiDiJiaoFei.c(607): Notify: Next row for parameter iteration = 1 [table = iteration].
YiDiJiaoFei.c(607): Notify: Parameter Substitution: parameter "iteration" = "1"
YiDiJiaoFei.c(607): Notify: Parameter Substitution: parameter "PhoneNum" = "13953555588"
YiDiJiaoFei.c(607): **Error: 消息: 提交异地缴费数据失败!,在第 1 次循环时出错,出错号码:13953555588**

第二次循环关键部分：
YiDiJiaoFei.c(598): Notify: Transaction "提交异地缴费数据" ended with "Fail" status (Duration: 4.2347 Wasted Time: 0.0064).
YiDiJiaoFei.c(605): Notify: Parameter Substitution: parameter "WCSParam_Text2" = "操作业务数据成功！"
YiDiJiaoFei.c(607): Notify: Next row for parameter iteration = 2 [table = iteration].
YiDiJiaoFei.c(607): Notify: Parameter Substitution: parameter "iteration" = "2"
YiDiJiaoFei.c(607): Notify: Parameter Substitution: parameter "PhoneNum" = "13953572390"
YiDiJiaoFei.c(607): **Error: 消息: 提交异地缴费数据失败!,在第 2 次循环时出错,出错号码:13953572390**

下面验证执行时，能正确找到出错的号码信息，将脚本加入到场景后，同样设置好 Run-time Settings,具体见下图：



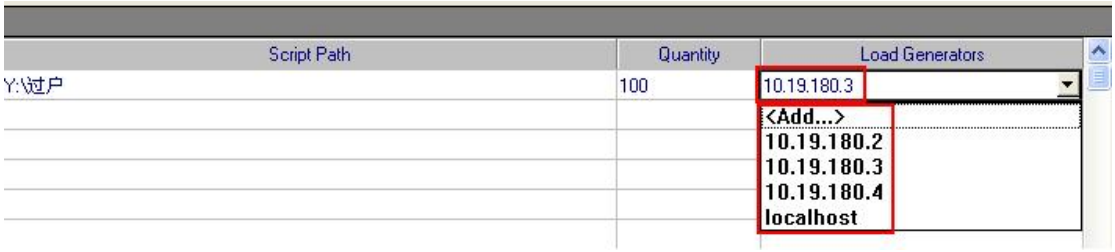
设置结果保存路径等相关信息，按 Start Scenario 执行场景，等出错是单击右上角的 Errors，然后选中错误信息再按 Details 按纽查看错误信息见下图。



6．怎么样使多台产生 vuser 的测试机均匀地对被测试的系统施加压力？

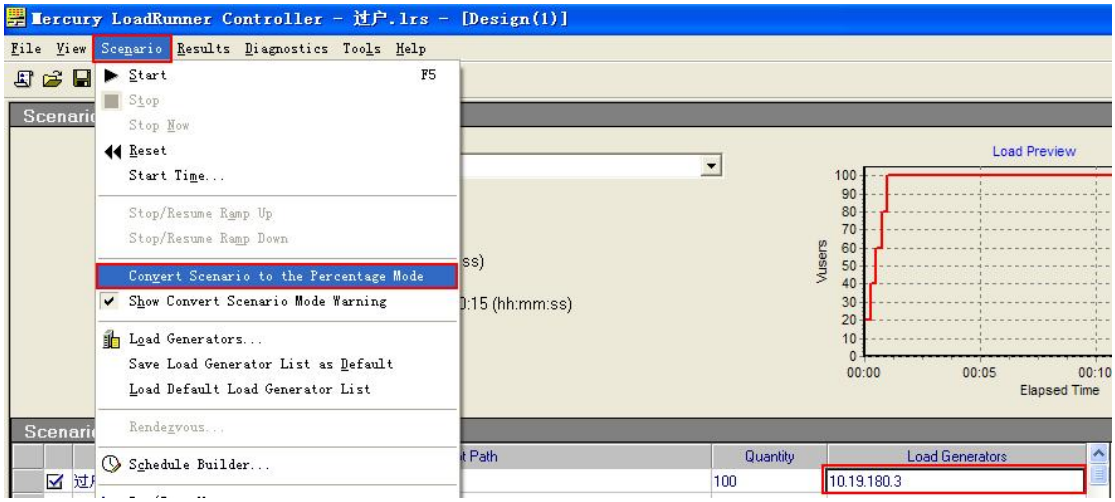
在测试的过程中，为了尽可能减少或者避免本身的测试机成为测试过程中的瓶颈，需要使用多台测试机产生 vuser 对被测试系统施加压力，下面对操作步骤做简单介绍：

在默认模式下使用 controller 添加多台 Generators 机器时，不管你怎么加，最终能真正起作用的只有一台（10.19.180.2/3/4 或 localhost），见下图：

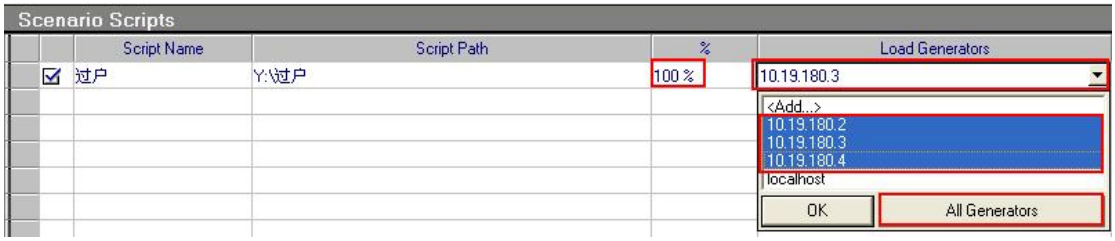


为了让 10.19.180.2/3/4 机器同时能真正被添加进去，我们需要做以下几步工作：

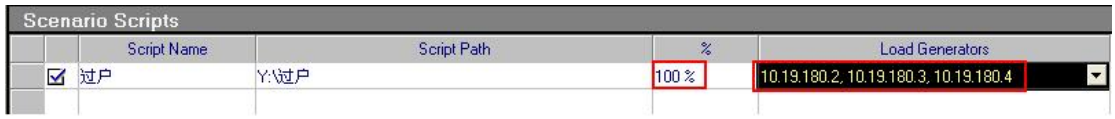
改变场景模式，将组模式（）改变为百分比模式（percentage mode），具体做法是，选择 Scenario 菜单下的 Convert Scenario to the Percentage Mode，见下图所示：



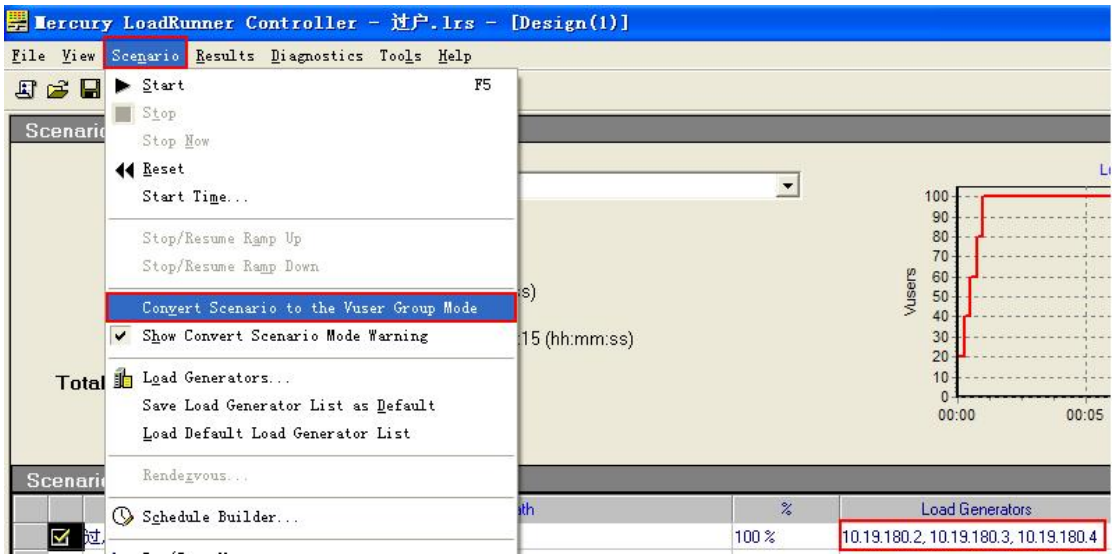
然后，在已经添加好的 Load Generators 机器列表中同时选择你想选择的机器，见下图：



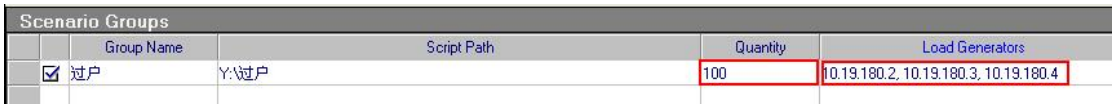
最后，点 OK 按钮就可以得到我们所要的结果了，见下图：



当然，如有必要我们还可以把场景模式改为 Vuser Group Mode，具体做法如下：选择 Scenario 菜单下的 Convert Scenario to the Vuser Group Mode，见下图



然后在弹出的对话框中，单击 Yes 按钮可以得到如下结果，见下图：



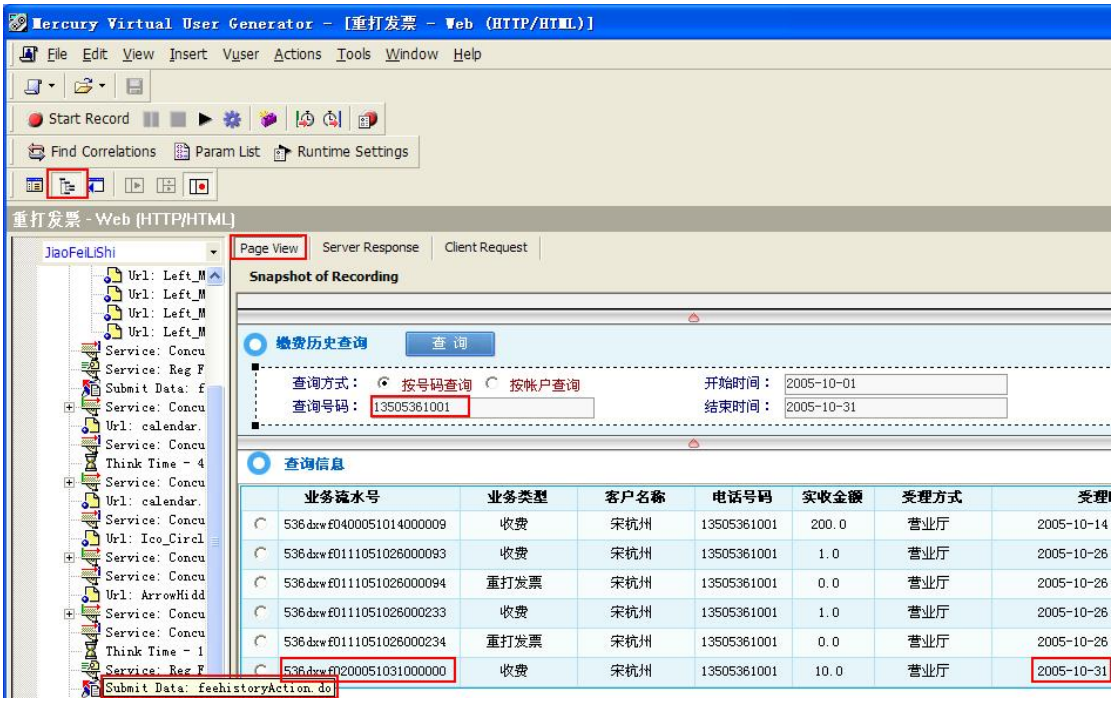
到此为止，添加多台 Load Generators 测试机整个过程就完成了，其实很简单，关键是你发现了没有。

7. 怎么样在关联时取列表的最后一个值（在测试重打发票取流水号时需要）？

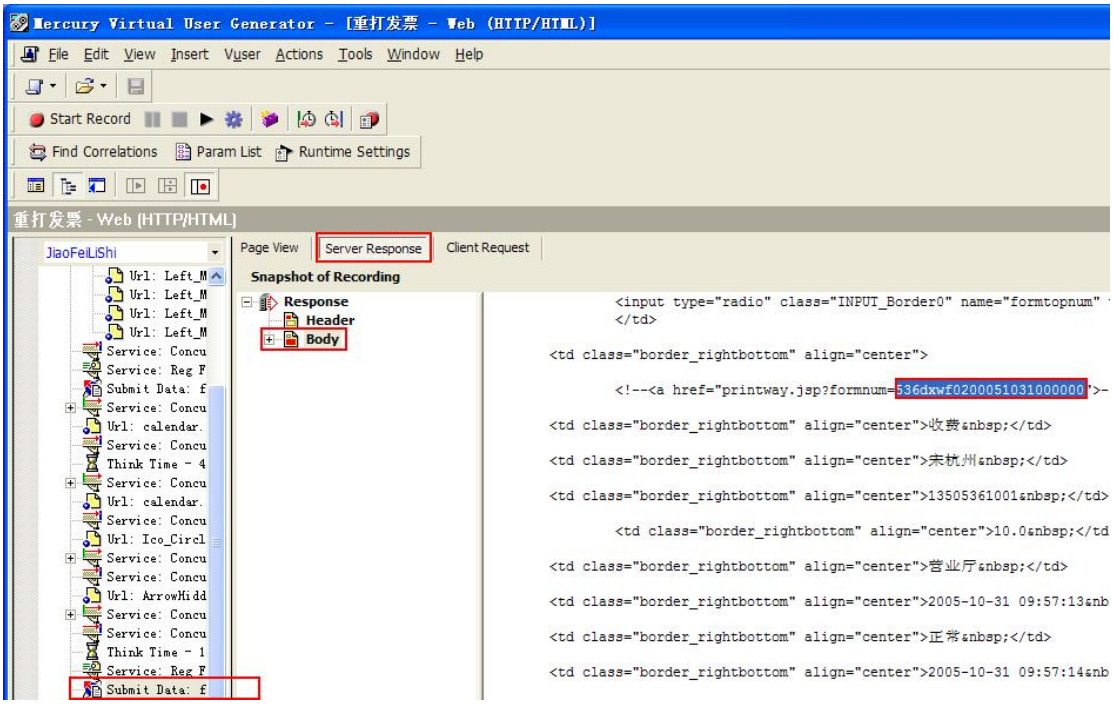
在压力测试脚本的关联过程中，我们有时可能需要关联最新的值（如最新的流水号，通常情况下，最新的流水号放在列表的最下方），所以找最新的流水号就是最列表最下方，如果保存在数组里，那就是找 index 值最大的那个元素。下面以重打发票（注：具体流程为先缴费，然后查询缴费历史，然后从缴费历史里找到最新的流水号，然后使用此流水号进行重打发票）为例对整个过程做详细的介绍：

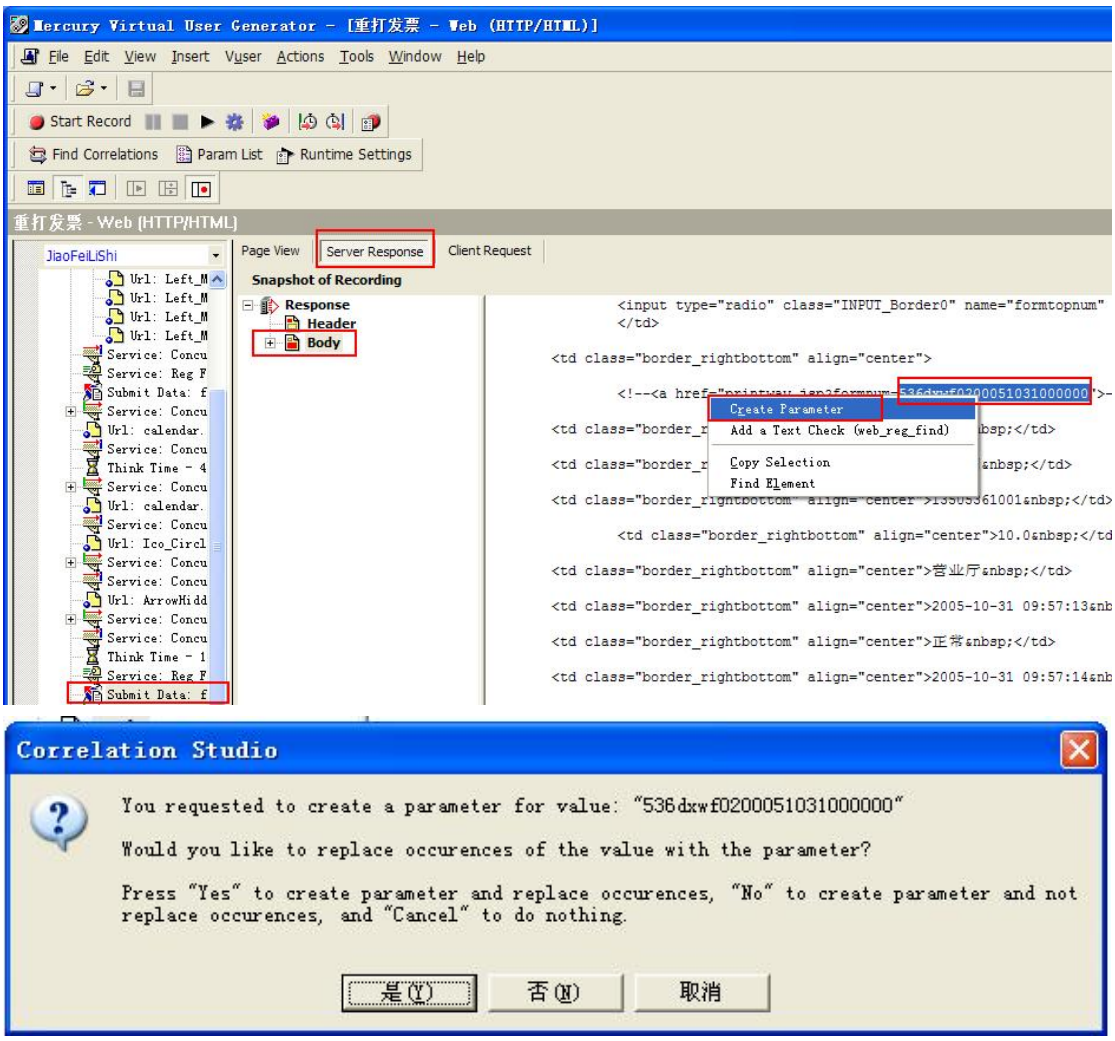
首先，在缴费历史里找到需要关联的流水号并关联之，具体做法如下，

7．1 以 Tree View 方式打开脚本并在对应事件的 Page View 里找到最新的流水号，见下图所示：

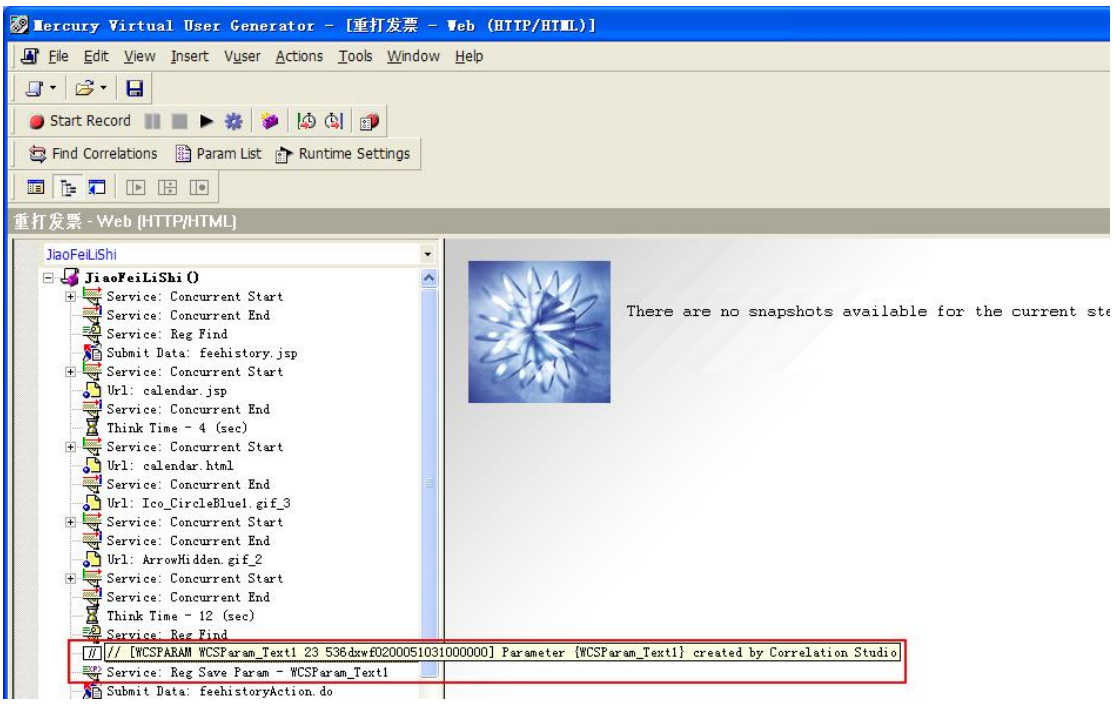


1．找到我们需要关联的流水号（这里为 536dxwf0200051031000000）后，需要把它给关联，（因为返回的值是事后才知道的，且对于不同的电话号码，对应的返回值不同，所以对于这样的值是需要关联的。）具体做法是打开 Server Response 页面并在 Body 里找到需要关联的流水号，然后选中此流水号并在右键弹出的菜单中使用 Create Parameter 关联之，见下面的图：

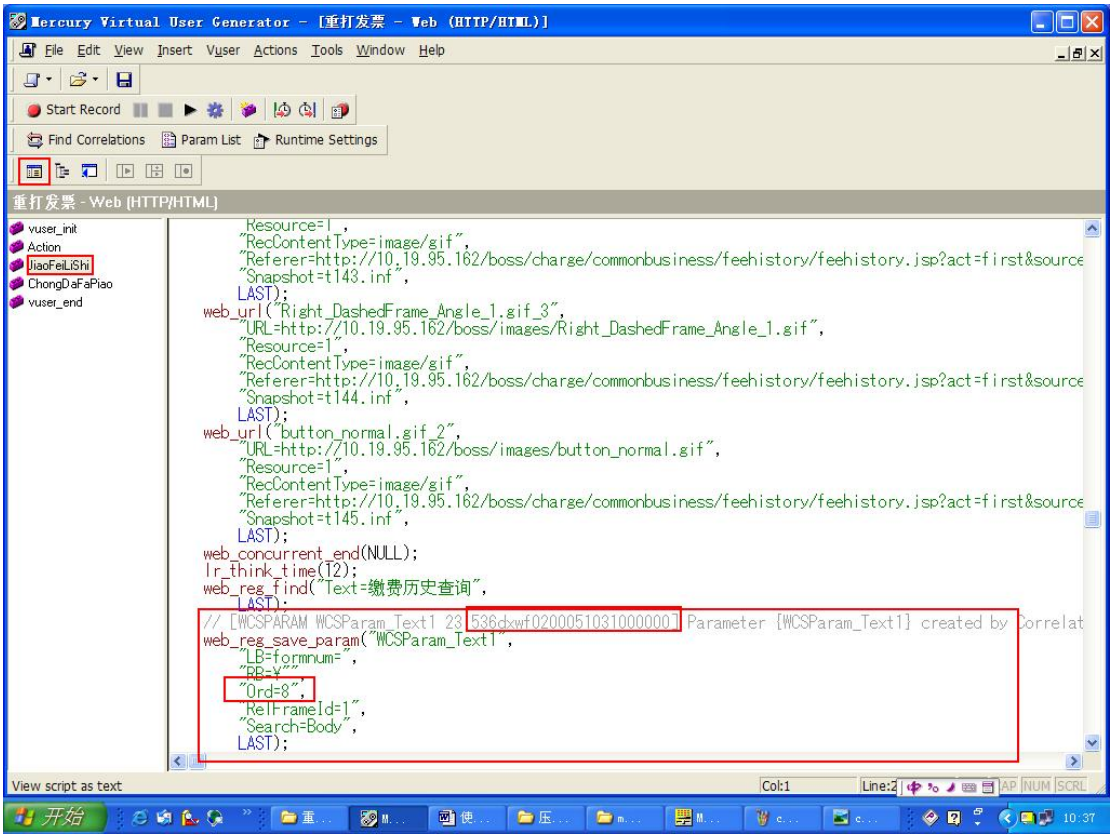



Mercury Virtual User Generator - [重打发票 - Web (HTTP/HTML)]
File Edit View Insert Vuser Actions Tools Window Help
Start Record
Find Correlations
Param List
Runtime Settings
重打发票 - Web (HTTP/HTML)
JiaoFeiLiShi
Page View
Server Response
Client Request
Snapshot of Recording
Response
Header
Body
Create Parameter
Add a Text Check (web_reg_find)
Copy Selection
Find Element
Correlation Studio
You requested to create a parameter for value: "536dxwf0200051031000000"
Would you like to replace occurences of the value with the parameter?
Press "Yes" to create parameter and replace occurences, "No" to create parameter and not replace occurences, and "Cancel" to do nothing.
是(Y)
否(N)
取消

7．2 单击是（Y）按钮，对应的脚本中会增加如下内容，见如下图片描述：



2．单击 View Script 图标以 Script 模式查看关联情况，见下图所示：



7．3 到此为止脚本中多出如下代码段，见下面红色文字部分，下面对它做一定的分析：

```
web_reg_find("Text=缴费历史查询",
    LAST);
// [WCSPARAM WCSParam_Text1 23 536dxwf0200051031000000] Parameter {WCSParam_Text1} created by Correlation Studio
web_reg_save_param("WCSParam_Text1",
    "LB=formnum=",
    "RB=\"",
    "Ord=8",
    "RelFrameId=1",
    "Search=Body",
    LAST);
```

//后面的内容为注释部分，说明流水号 536dxwf0200051031000000 已经关联并保存到 WCSParam_Text1 参数中。

web_reg_save_param（）为 LoadRunner 的保存参数所用的函数，其作用是将返回流水号保存到 WCSParam_Text1 参数中，以便使用不同的号码进行缴费历史查询出来的流水号能随着时间的变化流水号也跟着变化，而不是录制脚本时的 536dxwf0200051031000000，这样可以避免在重打发票时不会报诸如此流水号不存在等类似错误信息。然而现在的问题是怎么将此流水号对应到重打发票对应的地方。另一个问题是，不是所有的号码缴费后查询到的流水号数量都和录制脚本时查询到的流水号相同，事实上每做一笔除查询类的操作都会有一个流水号。而我们关注的是怎么取到最新的缴费的流水号，下面详细介绍相关步骤：

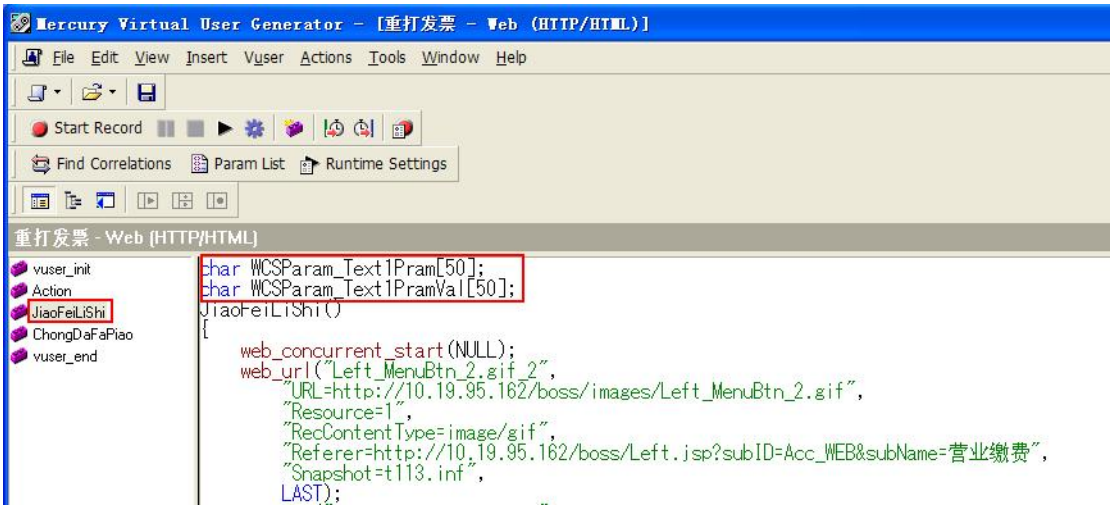
5．1 修改 web_reg_save_param（）函数相关部分，很简单，把"Ord=8",改为"Ord=ALL",目的是找最 TOP 的那个参数值。

5．2 光保存和取参数还不够，我们需要把参数能正确传递到重打发票对应的地方，为此我采取的做法如下：

在缴费历史事件脚本最前面定义两个变量，目的是为了将流水号以字符串的形式保存在变量里（因为 LoadRunner 不支持在 web_submit_data（）函数里直接使用变量）：具体做法是:

```
char WCSParam_Text1Pram[50];          //保存取到的流水号
char WCSParam_Text1PramVal[50];       //保存以"Value=流水号"取到的流水号。
```

见下图所示：



将关联好的流水号存到变量里，在此的做法是在对应的 web_submit_data（）函数后添加如下代码段：

lr_message("WCSParam_text1:%s",lr_eval_string("{WCSParam_Text1}"));

//打印出关联的参数 WCSParam_Text1 的值。

sprintf(WCSParam_Text1Pram,"{WCSParam_Text1_%s}",lr_eval_string("{WCSParam_Text1_count}"));

//把取到流水号保存到 WCSParam_Text1Pram 里，具体形式为

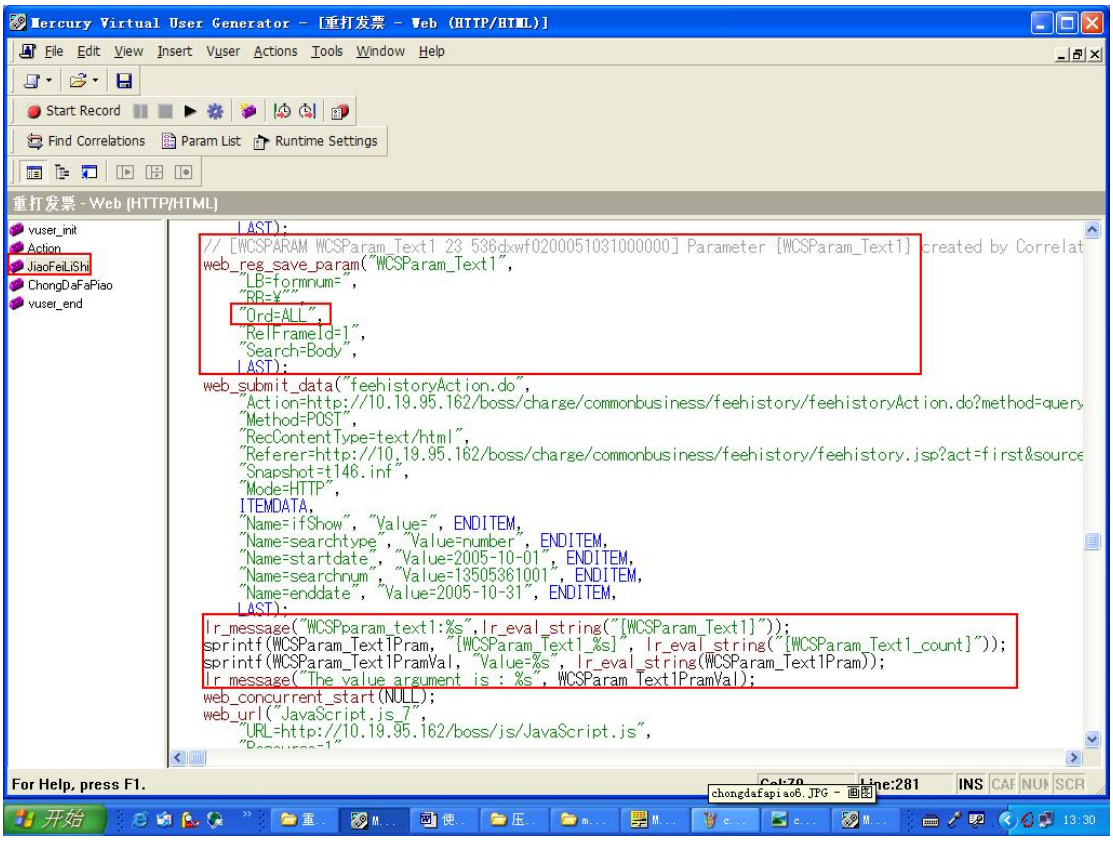
sprintf(WCSParam_Text1PramVal,"Value=%s",lr_eval_string(WCSParam_Text1Pram));

//组合流水号和”Value＝”并保存到 WCSParam_Text1PramVal 变量中。

lr_message("The value argument is : %s", WCSParam_Text1PramVal);

//打印出字符串变量 WCSParam_Text1PramVal 的值。

详细描述见下图所示：



5．3 找到重打发票中响应的流水号，并把其中的"Value=536dxwf0200051031000000"替换成 WCSParam_Text1PramVal，在这里总共有两处，见下面的图：

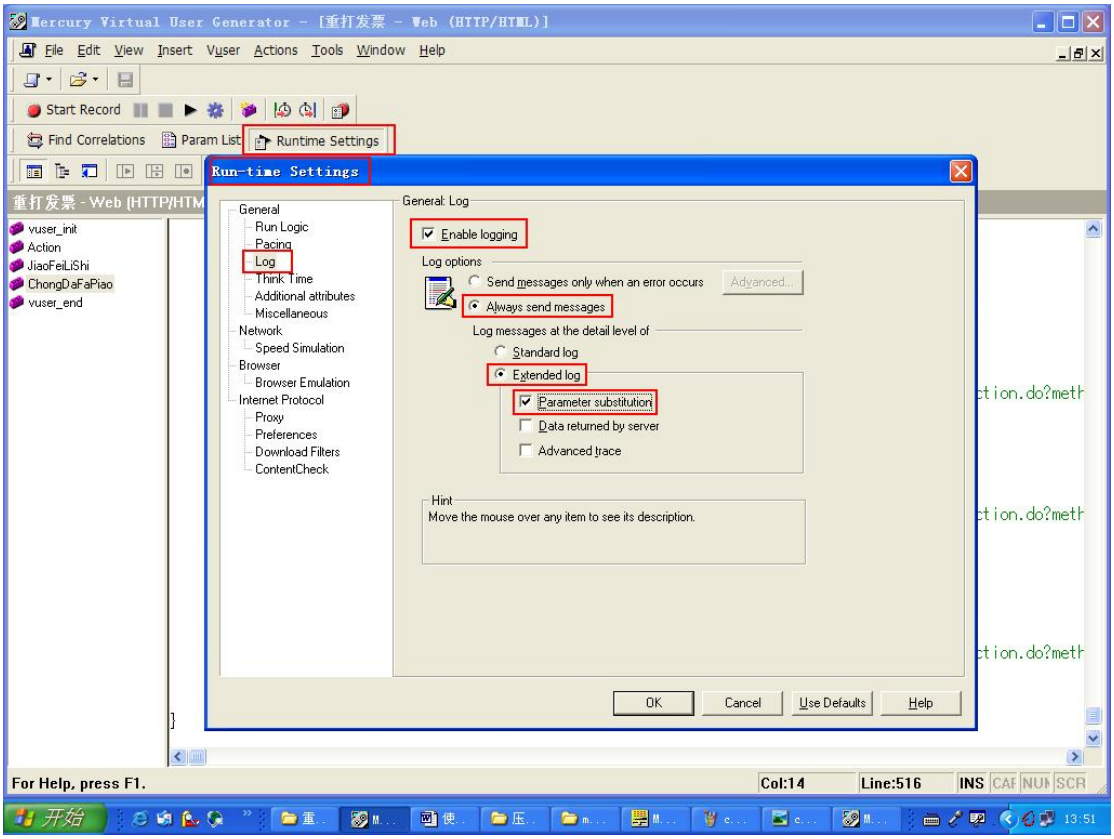
```
web_submit_data("reprintInvoiceAction.do",
    "Action=http://10.19.95.162/boss/charge/otherbusiness/reprintInvoice/reprintInvoiceAction.d
    "Method=POST",
    "RecContentType=text/html",
    "Referer=http://10.19.95.162/boss/charge/otherbusiness/reprintInvoice/reprintInvoice.jsp?ac
    "Snapshot=t203.inf",
    "Mode=HTTP",
    ITEMDATA,
    "Name=formtopnum", "Value=536dxwf0200051031000000", ENDITEM,
    "Name=glideNumber", "Value=", ENDITEM,
    LAST);

web_submit_data("reprintInvoiceAction.do_2",
    "Action=http://10.19.95.162/boss/charge/otherbusiness/reprintInvoice/reprintInvoiceAction.c
    "Method=POST",
    "RecContentType=text/html",
    "Referer=http://10.19.95.162/boss/charge/otherbusiness/reprintInvoice/reprintInvoiceAction.
    "Snapshot=t226.inf",
    "Mode=HTTP",
    ITEMDATA,
    "Name=formtopnum", "Value=", ENDITEM,
    "Name=glideNumber", "Value=536dxwf0200051031000000", ENDITEM,
    "Name=printType", "Value=1", ENDITEM,
    LAST);
```

替换后的分别为：

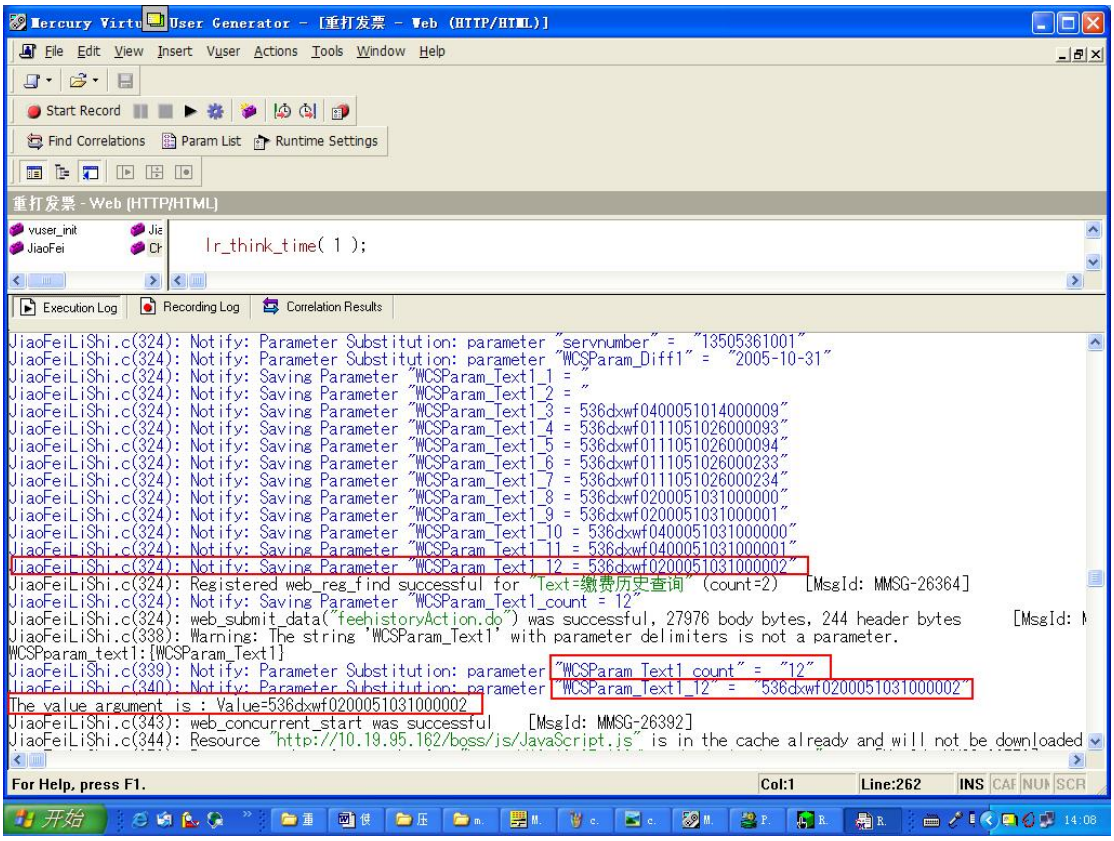
```
web_submit_data("reprintInvoiceAction.do",
    "Action=http://10.19.95.162/boss/charge/otherbusiness/reprintInvoice/reprintInvoiceAction.c
    "Method=POST",
    "RecContentType=text/html",
    "Referer=http://10.19.95.162/boss/charge/otherbusiness/reprintInvoice/reprintInvoice.jsp?ac
    "Snapshot=t203.inf",
    "Mode=HTTP",
    ITEMDATA,
    "Name=formtopnum", WCSParam_Text1PramVal, ENDITEM,
     Name=glideNumber ,  Value= , ENDITEM,
    LAST);

web_submit_data("reprintInvoiceAction.do_2",
    "Action=http://10.19.95.162/boss/charge/otherbusiness/reprintInvoice/reprintInvoiceAction.d
    "Method=POST",
    "RecContentType=text/html",
    "Referer=http://10.19.95.162/boss/charge/otherbusiness/reprintInvoice/reprintInvoiceAction.
    "Snapshot=t226.inf",
    "Mode=HTTP",
    ITEMDATA,
    "Name=formtopnum",  "Value=", ENDITEM,
    "Name=glideNumber", WCSParam_Text1PramVal, ENDITEM,
     Name=printType",  Value=1", ENDITEM,
    LAST);
```

到此为止，流水号的关联已经基本上处理完毕，下面我们执行脚本，来验证我们想要的是不是真的有效。（注，参数化的问题在本文中不做具体介绍）

为了看到明显的效果，我们需要将日志的处理做简单设置，具体如下图：



然后执行脚本，查看相关执行日志，可以得到类似下面得消息。



8．使用 LoadRunner 一些常用的注意事项：

Note1：VuGen 仅能录制 Windows 平台上的会话，但是，录制的 Vuser 脚本既可以在 Windows 平台上运行，也可以在 UNIX 平台上运行。
通用 Vuser 函数和特定于协议的函数，它们共同构成了 LoadRunner API，并使 Vuser 能够直接与服务器通信。

Note2:用于运行 Vuser 脚本的 C 解释器仅支持 ANSI C 语言。它不支持 Microsoft 对 ANSI C 的任何扩展。
通常情况下，可以将登录到服务器的活动录制到 vuser_init 部分中、将客户端活动录制到 Actions 部分中，并将注销过程录制到 vuser_end 部分中。

Note3：只能向 Action 部分（而不是 init 或 end 部分）添加集合。
Note：不要从事务内部发送消息，因为这可能使事务执行时间变长，并扭曲事务结果。

Note4：如果使用日志运行时设置修改脚本的调试级别，则 lr_message、lr_output_message 和 lr_log_message 函数的行为将不会更改，它们将继续发送消息。

Note5：录制 Java Vuser 脚本时， Vuser 脚本中将不生成 lr_think_time 语句。

Note6：VuGen 新建参数，但不会自动替换任何在脚本中选定的字符串。

Note7：不要将参数命名为 unique，因为该名已被 VuGen 使用。

Note8：如果在常规运行时设置文件夹中将“错误处理”设置为“出现错误时仍继续”，则错误消息仍将被发送到输出窗口。

Note9：因为生成的服务器消息很长，而且日志记录会降低系统的运行速度，所以请仅为脚本中特定的代码块激活服务器消息日志记录功能。

Note10：启用“出现错误时仍继续”功能时，将覆盖 0 严重级别；即使发生数据库错误，也将继续执行脚本。然而，如果禁用了“出现错误时仍继续”功能，但将严重级别指定为 1，则当发生数据库错误时仍将继续执行脚本。

Note11：下列协议不是线程安全协议：Sybase-Ctlib、Sybase-Dblib、Informix、Tuxedo 和 PeopleSoft-Tuxedo。

Note12：对于支持树视图的协议（如“视图”菜单所示），在树视图中运行 Vuser 脚本时， VuGen 将从 Vuser 脚本中的第一个图标开始运行该脚本。

Note13：要显示运行时查看器，必须安装 Microsoft Internet Explorer 4.0 或更高版本。
Note：Vuser 生成“结果摘要”报告时，事务时间可能会增加。Vuser 可以仅在从 VuGen 运行时才生成“结果摘要”报告。使用 Controller 运行 Web Vuser 脚本时， Vuser 不能生成报告。

Note14：当使用 JavaScript 和 VBScript Vuser 时，在脚本中用到的 COM 对象必须完全的兼容。这使下列情况成为了可能：一个应用程序操纵另一个应用程序中的对象，或者公开对象以便操纵它们。

9．性能参数解析：

WEB 资源参数

每秒点击次数：中 Vuser 每秒向 Web 服务器提交的 HTTP 请求数，依据点击次数来评估 Vuser 产生的负载量。

吞吐量：案运行过程中服务器上每秒的吞吐量。吞吐量的度量单位是字节，表示 Vuser 在任何给定的某一秒上从服务器获得的数据量，依据服务器吞吐量来评估 Vuser 产生的负载量。

每秒 HTTP 响应数：中每秒从 Web 服务器返回的 HTTP 状态代码号（表示 HTTP 请求的状态，例如“the request was successful”、“the page was not found”）

每秒下载页面数：每秒钟从服务器下载的网页数，依据下载的页面数来评估 Vuser 产生的负载量。

每秒重试次数：中每秒钟内服务器尝试的连接次数，在下列情况下将重试服务器连接：初始连接未经授权、要求代理服务器身份验证、服务器关闭了初始连接、初始连接无法连接到服务器或者服务器最初无法解析负载生成者的 IP 地址。

连接数：每个时间点上打开的 TCP/IP 连接数。

每秒 SSL 连接数：每秒打开的新的以及重新使用的 SSL 连接数。当对安全服务器打开 TCP/IP 连接后，浏览器将打开 SSL 连接，因为新建 SSL 连接需要消耗大量的资源，所以应该尽量少地打开新的 SSL 连接。

网页细分图

注意：由于要从客户端测定服务器时间，因此，如果发送初始 HTTP 请求到发送第一次缓冲这一段时间内网络性能发生变化，则网络时间可能会影响此测定。因此，所显示的服务器时间是一个估计值，可能不太精确。

DNS 解析：显示使用最近的 DNS 服务器将 DNS 名称解析为 IP 地址所需的时间。“DNS 查找”度量是指示 DNS 解析问题或 DNS 服务器问题的一个很好的指示器。

连接：显示与包含指定 URL 的 Web 服务器建立初始连接所需的时间。连接度量是一个很好的网络问题指示器。此外，它还可表明服务器是否对请求作出响应。

第一次缓冲：显示从初始 HTTP 请求（通常为 GET）到成功收回来自 Web 服务器的第一次缓冲时为止所经过的时间。第一次缓冲度量是很好的 Web 服务器延迟和网络滞后指示器。注意：由于缓冲区大小最大为 8K，因此第一次缓冲时间可能也就是完成元素下载所需的时间。

SSL 握手：显示建立 SSL 连接（包括客户端 hello、服务器 hello、客户端公用密钥传输、服务器证书传输和其他部分可选阶段）所用的时间，自此点之后，客户端与服务器之间的所有通信都将被加密。SSL 握手度量仅适用于 HTTPS 通信。

接收：显示从服务器收到最后一个字节并完成下载之前经过的时间。“接收”度量是很好的网络质量指示器（查看用来计算接收速率的时间/ 大小比率）。

FTP 验证：显示验证客户端所用的时间。如果使用 FTP，则服务器在开始处理客户端命令之前，必须验证该客户端。“FTP 验证”度量仅适用于 FTP 协议通信。

客户端时间：显示因浏览器思考时间或其他与客户端有关的延迟而使客户机上的

请求发生延迟时，所经过的平均时间。

错误时间：显示从发出 HTTP 请求到返回错误消息（仅限于 HTTP 错误）这期间经过的平均时间。

系统资源（UNIX 资源参数）：
CPU utilization :CPU 的使用时间百分比
Disk rate :磁盘传输速率
Paging rate :每秒钟读入物理内存或写入页面文件中的页数
Page-in rate :每秒钟读入到物理内存中的页数
Page-out rate :每秒钟写入页面文件和从物理内存中删除的页数
Collision rate :每秒钟在以太网上检测到的冲突数
Context switches rate :每秒钟在进程或线程之间的切换次数
Average load :上一分钟同时处于“就绪”状态的平均进程数
Swap-in rate :正在交换的进程数
System mode CPU utilization :在系统模式下使用 CPU 的时间百分比
User mode CPU utilization :在用户模式下使用 CPU 的时间百分比

网络监视参数
网络延迟时间：源计算机与目标计算机（例如，数据库服务器和 Vuser 负载生成器）之间的整个路径的延迟。

网络子路径时间：从源计算机到路径上每个节点的延迟。注意：从源计算机到每个节点的延迟是同时而又独立地度量的。因此，从源计算机到其中一个节点的延迟可能大于源计算机与目标计算机之间的整个路径上的延迟。

网络段延：路径上每个段的延迟。

验证网络是否是瓶颈：可以合并各种图来确定网络是否是瓶颈。例如，通过使用网络延迟时间图和运行 Vuser 图，可以确定 Vuser 的数量如何影响网络延迟。网络延迟时间图指示在方案运行期间的网络延迟。

数据库服务器
User Calls :在每次登录、解析或执行时， Oracle 会分配资源（Call State 对象）以记录相关的用户调用数据结构。在确定活动时，用户调用与 RPI 调用的比指明了，因用户发往 Oracle 的请求类型而生成的内部工作量

Total file opens :由实例执行的文件打开总数。每个进程需要许多文件（控制文件、日志文件、数据库文件）以便针对数据库进行工作

Opened cursors current :当前打开的光标总数

DB block changes :由于与一致更改的关系非常密切，此统计计算对 SGA 中所有块执行的、作为更新或删除操作一部分的更改总数。这些更改将生成重做日志项，

如果事务被提交，将是对数据库的永久性更改。此统计是一个全部数据库作业的粗略指示，并且指出（可能在每事务级上）弄脏缓冲区的速率。

10．AIX 操作系统机器性能瓶颈定义：

瓶颈定义
CPU bound : vmstat : when %user+%sys greater than 80%；
Disk I/O bound : vmstat : when %iowait greater than 40%(AIX4.3.3 or later)；
Application disk bound :vmstat : when %tm_act greater than 70%；
Paging space low : lsps -a : when used paging space greater than 70% active；
Paging bound : iostat vmstat : paging logical volumes %tm_act greater than 30% of the I/O(iostat) and paging activity greater than 10* the number of CPUs(vmstat)；
Thrashing : vmstat sar :rising page outs, CPU wait and run queue；

11．系统性能分析命令：

cpu : vmstat,iostat,topas,nmon,ps,sar,time,timex,netpmon,trace,trcrpt;
内存：vmstat,topas,nmon,ps,svmon,lsps,filemon,trace,trcrpt;
磁盘：iostat,topas,nmon,lvmstat,iostat -d, lvmstat, lsps,filemon,lsattr,lsdev;
网络：netstat,topas,nmon,entstat,nfsstat,ifconfig,iptrace,ipreport,trace,trcrpt;
监视 **CPU** 使用情况：vmstat 2 ; iostat -t 2 6;sar -P ALL 2 3;
监视内存使用情况： vmstat 2 10;ps aux;svmon -G;svmon -Pau 10;
监视 **I/O** 使用情况： iostat 5;sar -d 3 3;
监视网络使用情况： netstat -i ;netstat -m;netstat -v;

12．LR 运行原理图以及组件关系图：