软件测试教程 性能测试工具-LoadRunner（一）

工欲善其事，必先利其器，本节课我们就开始学习性能测试的工具-LoadRunner。

LoadRunner是一款开源桌面应用软件，可用来模拟用户负载完成性能测试工作，LoadRunner的功能在版本不断升级的过程中已经十分强大，现在很多互联网公司都在使用LoadRunner来完成产品或者

Loadrunner是业界公认的权威性能测试工具，被誉为工业级的性能测试工具，支持广泛的协议和平台。

本节主要介绍以下内容：

> LoadRunner安装
>
> LoadRunner的基本概念
>
> 开发测试脚本

LoadRunner的安装

现在以Loadrunner12为例，说明一下安装过程。

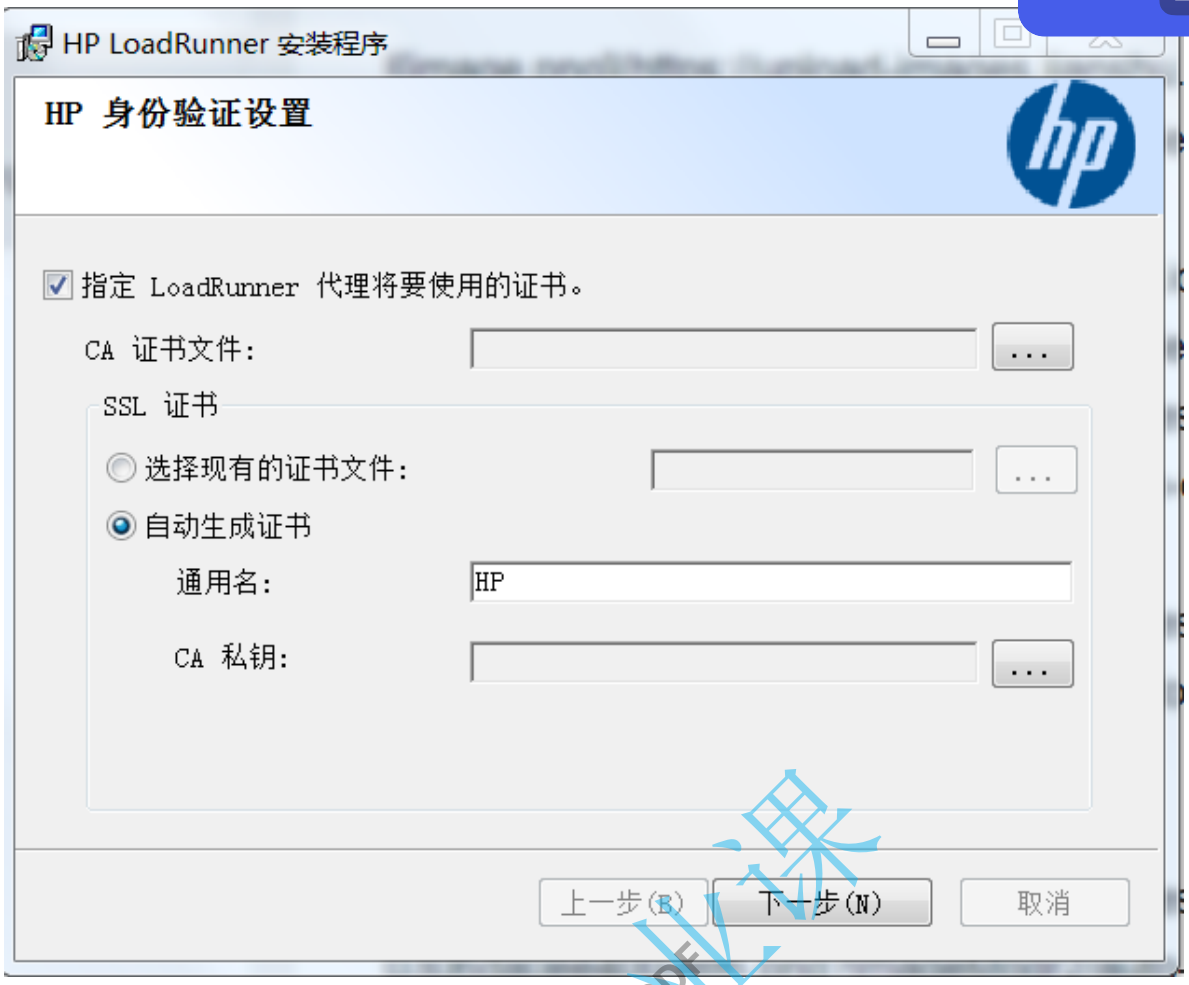
1、双击HP LoadRunner 12.53 Community Edition.exe启动安装程序

2、安装程序开始解压，选择默认路径即可，点击install

3、Loadrunner程序会使用到Visual C++的库，如果缺少这些库，安装程序会提示安装，这时选择“确定”进行安装即可，过程中如果提示重启，按要求重启即可，重启后会继续进行安装过程。

4、按照默认操作一步步进行安装

5、安装过程中会有如下的提示。若无指定代理使用的证书，则去掉勾选。



6、至此，Loadrunner已安装完毕。HP network Virtualization为可选项，可不安装。

7、双击HP_LoadRunner_12_Community_Edition_User_Interface_Packs_T7177-15057.exe，安装中文包。

系统将抽取语言包安装包，可选择抽取的语言包临时存放路径。建议直接默认即可，点击【Install】

8、抽取安装包完成后将自动关闭窗口（注此处只是把安装包抽取出来了，要到抽取的安装包中进行安装），此时需要到上一步中选择的路径中找到语言安装包。如未修改路径则在以下路径"C:\Temp\HP LoadRunner 12.02 Community Edition\DVD"打开该文件夹。点击"Setup"

9、将自动打开安装目录，点击【语言】

10、打开选择语言文件夹，选择要安装的语言。本处依次打开如下文件【Chinese-Simplified】→【LoadRunner】→【LR_CHS】，点击【LR_CHS】将进行安装。（其实可以省略掉第8,9步，直接找到该文件安装即可）

Loadrunner的基本概念

功能：LoadRunner是一种适用于许多软件体系架构的自动负载测试工具，从用户关注的响应时间、吞吐量，并发用户和性能计数器等方面来衡量系统的性能表现，辅助用户进行系统性能的优化。

原理：LR启动以后，在任务栏会有一个Agent进程，通过Agent进程，监视各种协议的Client与Server端的通讯，用LR的一套C语言函数来录制脚本，所以只要LR支持的协议，就不会存在录制不到的，然后LR调用这些脚本向服务器端发出请求，接受服务器的响应。至于服务器内部如何处理，它不关心。

LoadRunner通过以模拟上千万用户实施并发负载及实时性能监测的方式来确认和查找问题，优化性能和加速应用系统的发布周期。

组成：LoadRunner主要包括三个前台功能组件，分别为VuGen（虚拟用户脚本生成器）、Cont
试控制器）和Analysis（结果分析器）。系统会自动调用后台功能组件LG（负载生成器）和Proxy（用户代理）来完成性能测试工作。

VuGen 是录制与便携脚本的地方。通过录制或编写脚本来模拟用户的行为。

Controller是执行负载测试管理和监控的中心。在这里指定具体的性能测试方案，执行性能测试，收集测试数据，监控测试指标。监控工具将测试过程中收集到的客户机、服务器和网络性能指标数据显示在监控页面上，便于测试人员对系统表现进行随时掌握。

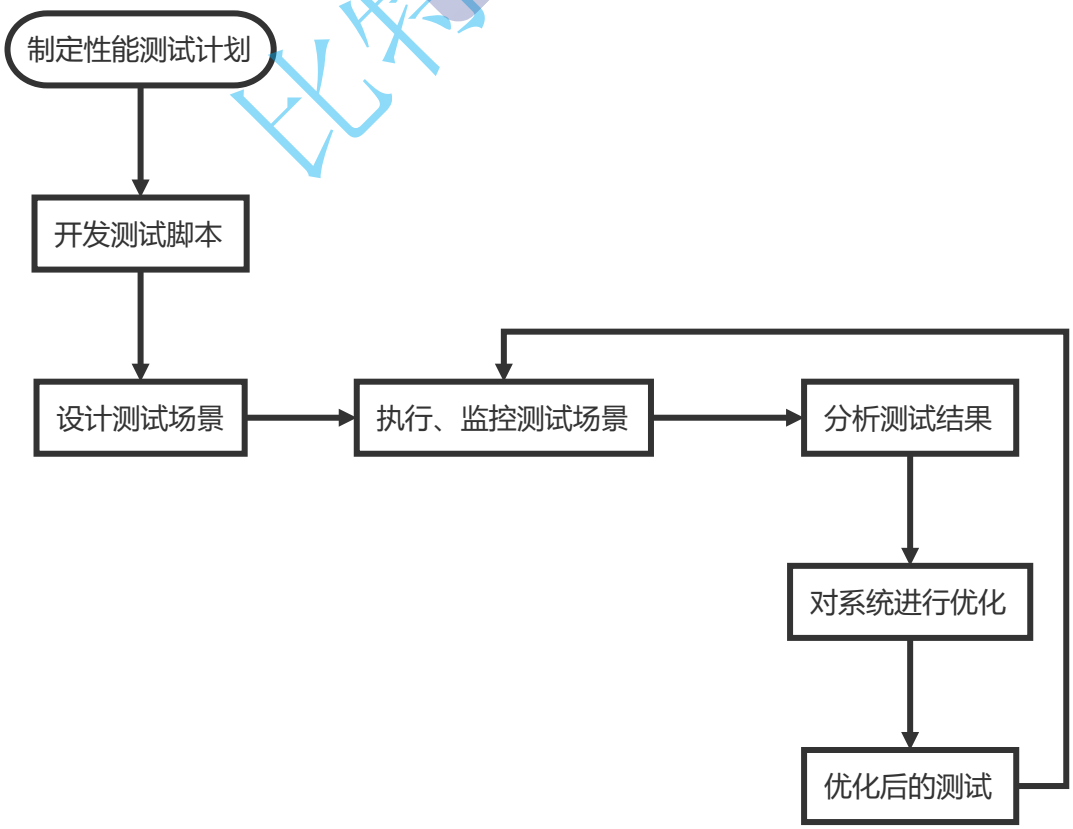
LG是模拟多用户并发访问被测试系统的组件。模拟多用户访问系统的前提是已经具备了虚拟用户脚本，VuGen是录制和编辑虚拟用户脚本的工具，录制好的脚本是不同语言表达的文本文件，在LG执行时被解析和执行。脚本录制和回放过程是在Proxy支持下完成的。

Analysis在测试完成后，对测试过程中收集到的各种性能数据进行计算、汇总和处理，生成各种图表和报告，为系统性能测试结果分析提供支持。

在使用loadrunner之前，先了解一下几个概念：

- Scenario：场景。所谓场景，是指在每一个测试过程中发生的事件。
- Vusers：虚拟用户。LoadRunner使用多线程或多进程来模拟用户对应用程序操作时产生的压力。一个场景可能包括多个虚拟用户，甚至成千上万个虚拟用户。
- Vuser Script:脚本。用脚本来描述Vuser在场景中执行的动作。
- Transactions：事务。事务代表了用户的某个业务过程，需要衡量这些业务过程的性能。
- rendezvous：集合。当我们测试多个用户并发时，每个用户执行到该事务脚本的先后顺序是不确定的，所以得到的测试结果也并不是一个完全 并发的极限测试结果。在开始事务之前，插入一个“集合点”，那么在多用户执行时，就可以将用户请求停下来，直到用户数量达到满足的条件（默认是100%的用户都到达集合点）。那么，所有的用户都将同时发出接下来的请求。

Loadrunner的性能测试过程：



制定性能测试计划

这部分内容已经在上一节进行了讲述。主要实现以下内容：

分析应用程序、确定测试目标、计划怎样执行

开发测试脚本

LoadRunner 使用虚拟用户的活动来模拟真实用户来操作Web 应用程序，而虚拟用户的活动就包含在测试脚本中，所以说测试脚本对于测试来说是非常重要的。

开发测试脚本要使用 VuGen 组件。测试脚本要完成的内容有：

- 每一个虚拟用户的活动
- 参数化
- 定义事物
- 定义检查点

设计运行场景

运行场景描述在测试活动中发生的各种事件。一个运行场景包括一个运行虚拟用户活动的Load Generator 机器列表，一个测试脚本的列表以及大量的虚拟用户和虚拟用户组。

运行、监视测试

一切配置妥当，开始运行测试。在运行过程中，需要监视各个服务器的运行情况（DataBase Server、Web Server 等）。

分析测试结果

所有前面的准备都是为了这一步。我们需要分析大量的图表，生成各种不同的报告，最后会得出结论。

LoadRunner用3个主要功能模块来覆盖性能测试的基本流程。

- Virtual User Generator
- Controller
- Analysis

其中Virtual User Generator使用在创建VU脚本阶段，Controller用在定义场景阶段和运 行场景阶段，Analysis用在分析结果阶段。

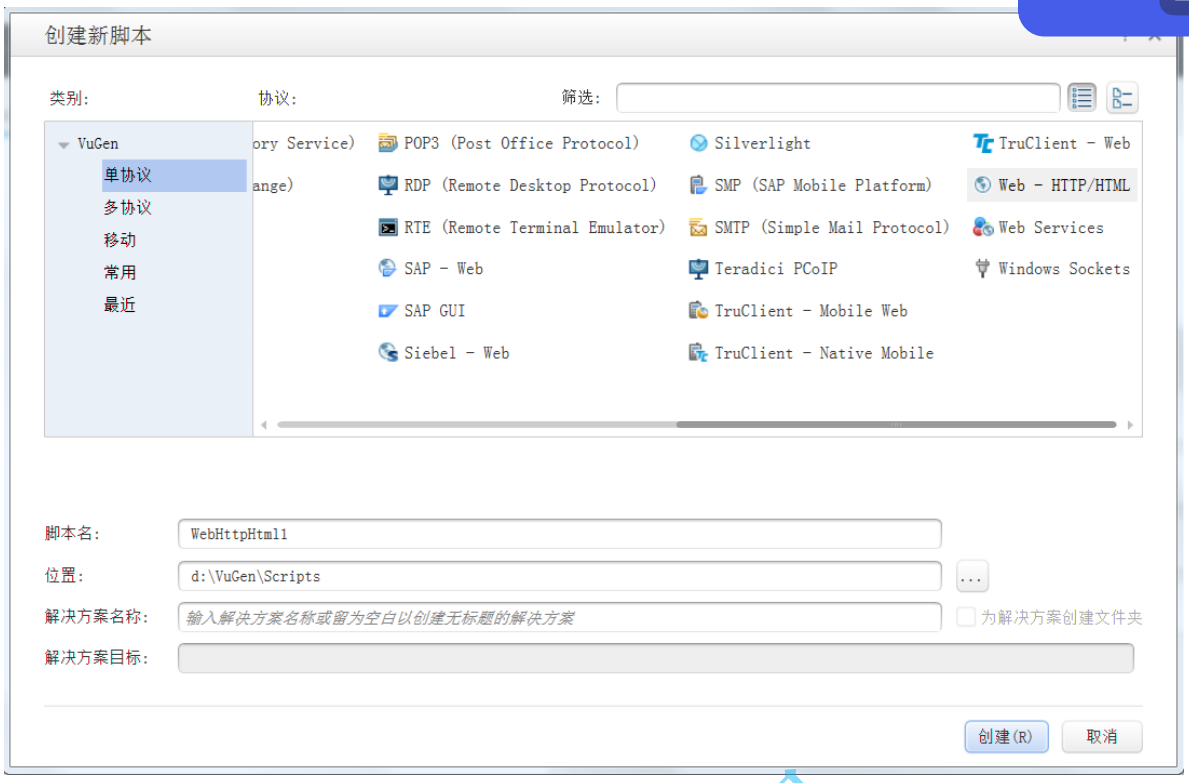
开发测试脚本

下面以Loadrunner安装时附带的样例程序Web Tours进行讲解。

C:\Program Files (x86)\HP\LoadRunner\WebTours，选择StartServer.bat启动服务。

录制基本的用户脚本

1、启动 Visual User Generator 后，选择新建脚本，因为要测试的是web项目，所以选择协议为Web-HTTP/HTML，点击创建后，进入主窗体



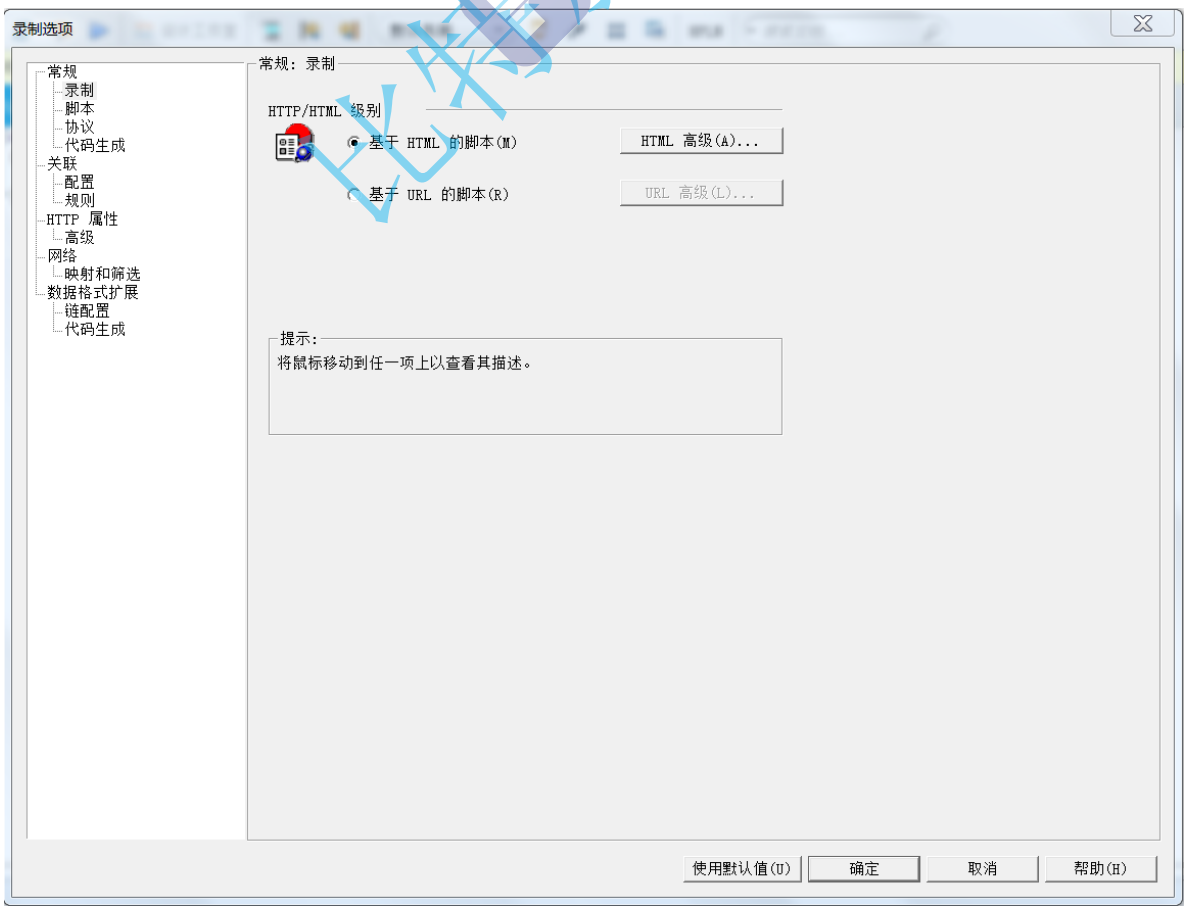
2、在解决方案资源管理器中可以看到该脚本的组成部分。简单说明一下：VuGen 中的脚本分为三部分：vuser_init、vuser_end 和Action。vuser_init用于用户初始化，vuser_end用于用户清理工作。Action用于具体的需要测试的操作。类似于unittest等测试框架。

> 举例说明：
> 一个测试场景为：用户登录系统，进行搜索操作，再进行退出系统。
> 这里，一般将登陆放置到`vuser_init`，退出放置到`vuser_end`，搜索放置到`Action`。

注意：在重复执行测试脚本时，vuser_init 和vuser_end 中的内容只会执行一次，重复执行的只是 Action 中的部分。

3、选择录制操作，可以开始一次录制操作，在录制中需要填写URL，这里以http://127.0.0.1:1080/WebTours/为例。录制到操作说明是将脚本放置到哪里。在录制中也可以修改脚本放置的地方。 已注册的用户名和密码查看地址：xxx\HP\LoadRunner\WebTours\cgi-bin\users 开始录制中，“立即”默认情况下是选中的，说明应用程序一旦启动，VuGen 就会开始录制脚本；如果没有选中，应用程序启动后，VuGen 出现对话框，待确认后才开始录制。一般默认即可。

4、在以上页面上有录制选项可以进行一些高级选项的设置，这里一般不需要改动。



Recording 标签页：

- 基于HTML的脚本：这种方式录制的代码只生成了一个函数，所有请求全放在这个函数里面，每一个操作(可能包含多个请求)会生成一个函数，这种方式看起来比较简洁

  基于HTML的脚本模式下-->高级选项两种方式的区别：

  描述用户操作的脚本(e.g. web_link,web_submit_form)：上下文相关的，依赖上下文才能提交，比较符合人们的操作习惯。

  仅包含明确URL的脚本(e.g. web_url, web_submit_data)：上下文不相关，每个函数都指定了具体的url地址，可以直接提交成功，如果只关注协议，不需要关注页面，可使用这种方式录制。

- 基于URL的脚本：这种方式会生成很多函数，它将每个请求都单独成一个函数，这种方式更接近请求-响应的本质

选择哪种方式录制，有以下参考原则：

- 基于浏览器的应用程序推荐使用HTML-based Script
- 不是基于浏览器的应用程序推荐使用 URL-based Script
- 如果基于浏览器的应用程序中包含了 JavaScript 并且该脚本向服务器产生了请求，比如DataGrid的分页按钮等，也要使用URL-based 方式录制
- 基于浏览器的应用程序中使用了 HTTPS 安全协议，使用URL-based 方式录制

5、点击开始录制，loadrunner会自动调用IE，并开始录制操作。这里以注册为例进行录制，录制完毕后，点击停止，录制停止，返回到脚本界面，可以看到已录制的脚本。

录制过程中，在屏幕上会有一个工具条出现。录制提供了暂停、停止、新增操作，增加事务、增加集合点等操作。

插入事务

当录制完一个基本的用户脚本后，在正式使用前我们还需要完善测试脚本，增强脚本的灵活性。一般情况下，我们通过以下方法来完善测试脚本。

事务（Transaction）：为了衡量服务器的性能，我们需要定义事务。比如：我们在脚本中有一个数据查询操作，为了衡量服务器执行查询操作的性能，我们把这个操作定义为一个事务，这样在运行测试脚本时，LoadRunner 运行到该事务的开始点时，LoadRunner 就会开始计时，直到运行到该事务的结束点，计时结束。这个事务的运行时间在结果中会有反映。插入事务操作可以在录制过程中进行，也可以在录制结束后进行。LoadRunner 可以在脚本中插入不限数量的事务。

通过菜单设计---在脚本中插入---开始事务、结束事务来进行事务的添加。

事务的状态默认情况下是 LR_AUTO。一般情况下，我们也不需要修改，除非在手工编写代码时，有可能需要手动设置事务的状态。可以通过步骤导航器来查看步骤的参数选项。

以上述注册为例，在实际生成的脚本中含有打开首页、注册、退出登录等多项操作。而我们实际需要关注的是注册这一个事务的性能，那么就需要在注册前后来加入事务。

```
lr_start_transaction("register");

    web_submit_form("login.pl_2",
        "Snapshot=t8.inf",
        ITEMDATA,
        "Name=username", "Value=test", ENDITEM,
        "Name=password", "Value=123456", ENDITEM,
```

```
        "Name=passwordConfirm", "Value=123456", ENDITEM,
        "Name=firstName", "Value=", ENDITEM,
        "Name=lastName", "Value=", ENDITEM,
        "Name=address1", "Value=", ENDITEM,
        "Name=address2", "Value=", ENDITEM,
        "Name=register.x", "Value=59", ENDITEM,
        "Name=register.y", "Value=10", ENDITEM,
        LAST);

    web_image("button_next.gif",
        "Src=/WebTours/images/button_next.gif",
        "Snapshot=t9.inf",
        LAST);
lr_end_transaction("register", LR_AUTO);
```

插入集合点

插入集合点是为了衡量在加重负载的情况下服务器的性能情况。在测试计划中，可能会要求系统能够承受1000 人同时提交数据，在LoadRunner 中可以通过在提交数据操作前面加入集合点，这样当虚拟用户运行到提交数据的集合点时，LoadRunner 就会检查同时有多少用户运行到集合点，如果不到1000 人，LoadRunner 就会命令已经到集合点的用户在此等待，当在集合点等待的用户达到1000 人时，LoadRunner 命令1000 人同时去提交数据，从而达到测试计划中的需求。

注意：集合点经常和事务结合起来使用。集合点只能插入到Action 部分，vuser_init和vuser_end 中不能插入集合点。

具体的操作方法如下：在需要插入集合点的前面，通过菜单操作：菜单设计---在脚本中插入---集合

```
lr_rendezvous("index");
```

参数化输入

如果用户在录制脚本过程中，填写提交了一些数据，比如要增加数据库记录。这些操作都被记录到了脚本中。当多个虚拟用户运行脚本时，都会提交相同的记录，这样不符合实际的运行情况，而且有可能引起冲突。为了更加真实的模拟实际环境，需要各种各样的输入。

参数化输入是一种不错的方法。

用参数表示用户的脚本有两个优点：

① 可以使脚本的长度变短。

② 可以使用不同的数值来测试你的脚本。例如，如果你企图搜索不同名称的图书，你仅仅需要写提交函数一次。在回放的过程中，你可以使用不同的参数值，而不只搜索一个特定名称的值。

参数化包含以下两项任务：

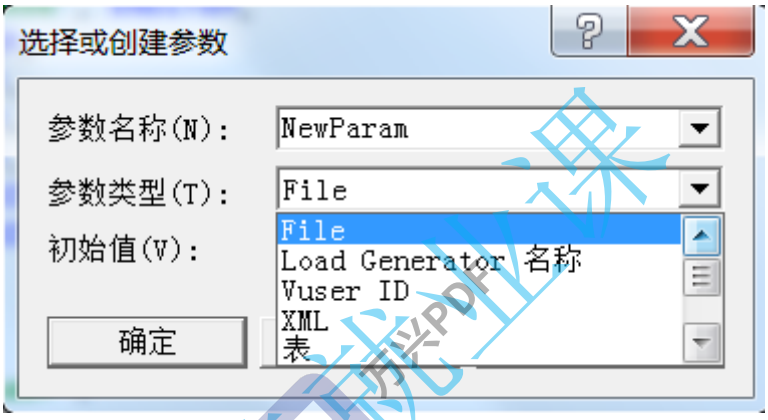
① 在脚本中用参数取代常量值。

② 设置参数的属性以及数据源。

参数化仅可以用于一个函数中的参量。你不能用参数表示非函数参数的字符串。

例如在上面的注册的例子中，我们已经注册了test用户，那么再次注册就会失败。也就是说[illegible]脚本再次运行就会失败。找到以下的代码块，在username中选中“test”字符串点击右键选择“使用参数替换”，就可以进行参数设置。

```
web_submit_form("login.pl_2",
        "Snapshot=t8.inf",
        ITEMDATA,
        "Name=username", "Value=test", ENDITEM,
        "Name=password", "Value=123456", ENDITEM,
        "Name=passwordConfirm", "Value=123456", ENDITEM,
        "Name=firstName", "Value=", ENDITEM,
        "Name=lastName", "Value=", ENDITEM,
        "Name=address1", "Value=", ENDITEM,
        "Name=address2", "Value=", ENDITEM,
        "Name=register.x", "Value=59", ENDITEM,
        "Name=register.y", "Value=10", ENDITEM,
        LAST);
```



下面介绍一下参数的类型。

日期/时间：很简单，在需要输入日期/时间的地方，可以用DateTime 类型来替代。其属性设置也很简单，选择一种格式即可。当然也可以定制格式。

组名：在实际运行中，LoadRunner使用该虚拟用户所在的Vuser Group 来代替。但是在VuGen 中运行时，Group Name将会是None

Load Generator Name：在实际运行中，LoadRunner 使用该虚拟用户所在LoadGenerator 的机器名来代替。

迭代编号：在实际运行中，LoadRunner 使用该测试脚本当前循环的次数来代替。

随机数字：随机数。很简单。在属性设置中可以设置产生随机数的范围

唯一编号：唯一的数。在属性设置中可以设置第一个数以及递增的数的大小（每个Vuser的块大小）。

注意：使用该参数类型必须注意可以接受的最大数。例如：某个文本框能接受的最大数为99。当使用该参数类型时，设置第一个数为1，递增的数为1，但100 个虚拟用户同时运行时，第100 个虚拟用户输入的将是100，这样脚本运行将会出错。

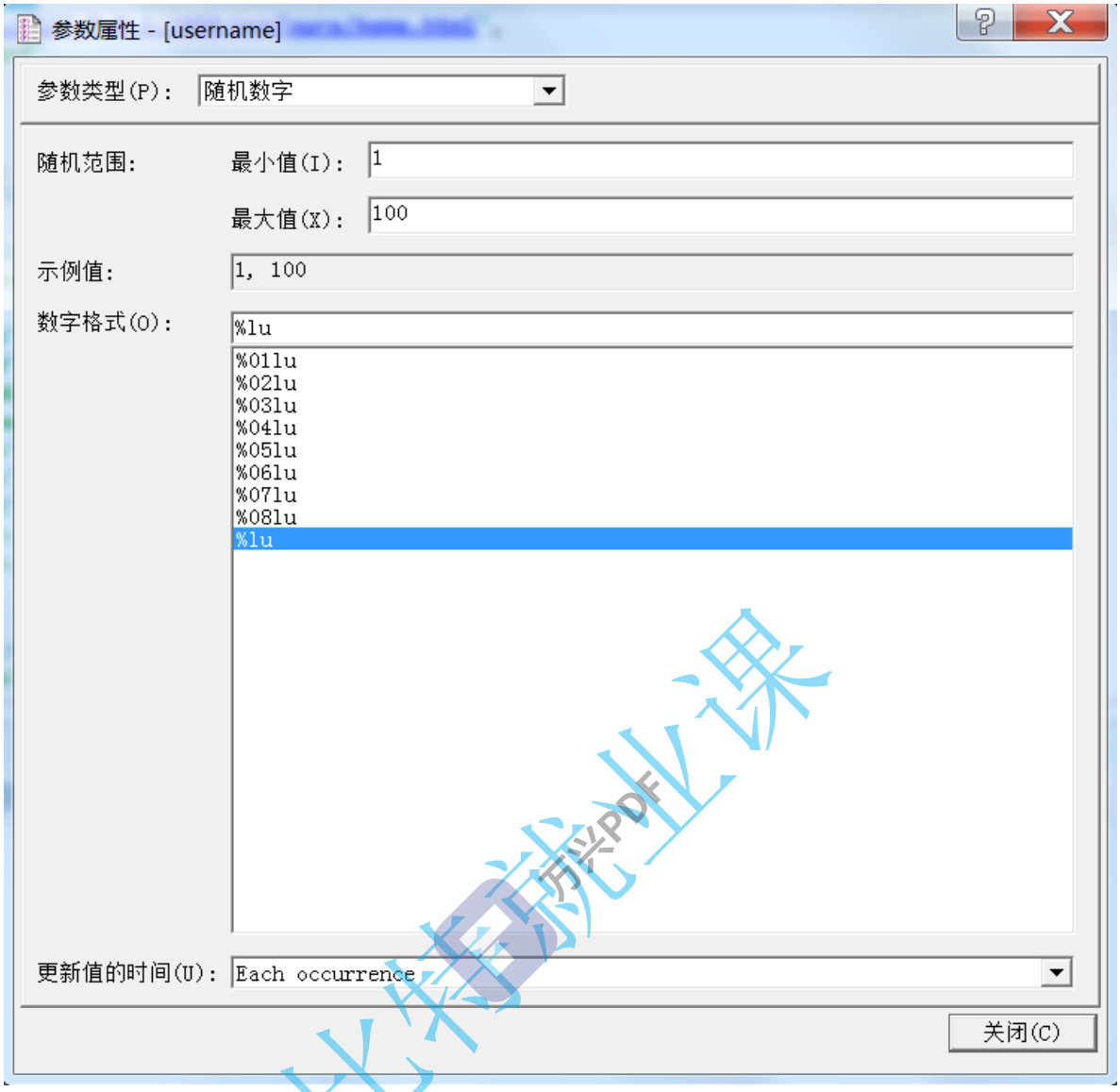
注意：这里说的递增意思是各个用户取第一个值的递增数，每个用户相邻的两次循环之间的差值为1。举例说明：假如起始数为1，递增为5，那么第一个用户第一次循环取值1，第二次循环取值2；第二个用户第一次循环取值为6，第二次为7；依次类推。

Vuser ID：设置比较简单。在实际运行中，LoadRunner 使用该虚拟用户的ID 来代替，该ID 是由Controller 来控制的。但是在VuGen 中运行时，Vuser ID 将会是-1。

File：需要在属性设置中编辑文件，添加内容

用户定义的函数：从用户开发的dll 文件提取数据。

在上面的例子中，我们取随机数就可以了。选择随机数后，点击“属性... ..”按钮，进行属性设置窗口



添入随机数的取值范围为（1-50），选择一种数据格式。在更新值的时间中有以下几个选项：

- Each Occurrence：在运行时，每遇到一次该参数，便会取一个新的值
- Each iteration：运行时，在每一次循环中都取相同的值, 不同的循环中值不一样。
- Once：运行时，在所有的循环中，该参数只取一次值

这里我们用的是随机数，选择默认的Each Occurrence 就非常合适。

File等类型中有以下参数：
“选择下一行 ”有以下几种选择：
Sequential：按照顺序一行行的读取。每一个虚拟用户都会按照相同的顺序读取
Random：在每次循环里随机的读取一个，但是在循环中一直保持不变
Unique ：唯一的数。注意：使用该类型必须注意数据表有足够多的数。比如Controller 中设定20 个虚拟用户进行5 次循环，那么编号为1 的虚拟用户取前5个数，编号为2 的虚拟用户取6-10 的数，依次类推，这样数据表中至少要有100个数据，否则Controller 运行过程中会返回一个错误。

为了避免不正确的参数导致不可用，在设置后，可以选择“模拟参数”来试运行

插入函数

VuGen 中可以使用C 语言中比较标准的函数和数据类型，语法和C 语言相同。下面简单介绍一下比较常用的函数和数据类型。

在脚本页面，通过右键-插入-新建步骤可以查看函数列表

1. 控制脚本流程

```
if { } else { }
for{ }
while{ }
```

..... 总之 C 语言的控制流程的语句这里都可以直接使用

2. 字符串函数

由于在 VuGen 脚本中使用最多的还是字符串，所以字符串函数在脚本中使用非常频繁。具体的语法请参考帮助说明。

```
strcmp 比较两个字符串
strcat 连接两个字符串
strcpy 拷贝字符串
```

..... 注意：在VuGen 中，以char*声明的字符串是只读的，如果试图给char*类型的字符串赋值的话，编译会通过，但在运行时会产生"Access Violation"的错误。解决这类问题，就是把字符串声明为字符数组，比如char[100]。

3. 输出函数

输出函数在调试脚本时非常有用。

```
lr_output_message 输出一条消息
```

4. LoadRunner 提供的标准函数

```
lr_eval_string 该函数功能是得到参数（参数化输入中）当前的值
exg: lr_output_message("temp = %s", lr_eval_string("{WCSParam2}"));
lr_save_string 该函数功能是把一个字符串保存到参数中
exg: lr_save_string("439","WCSParam3");
```

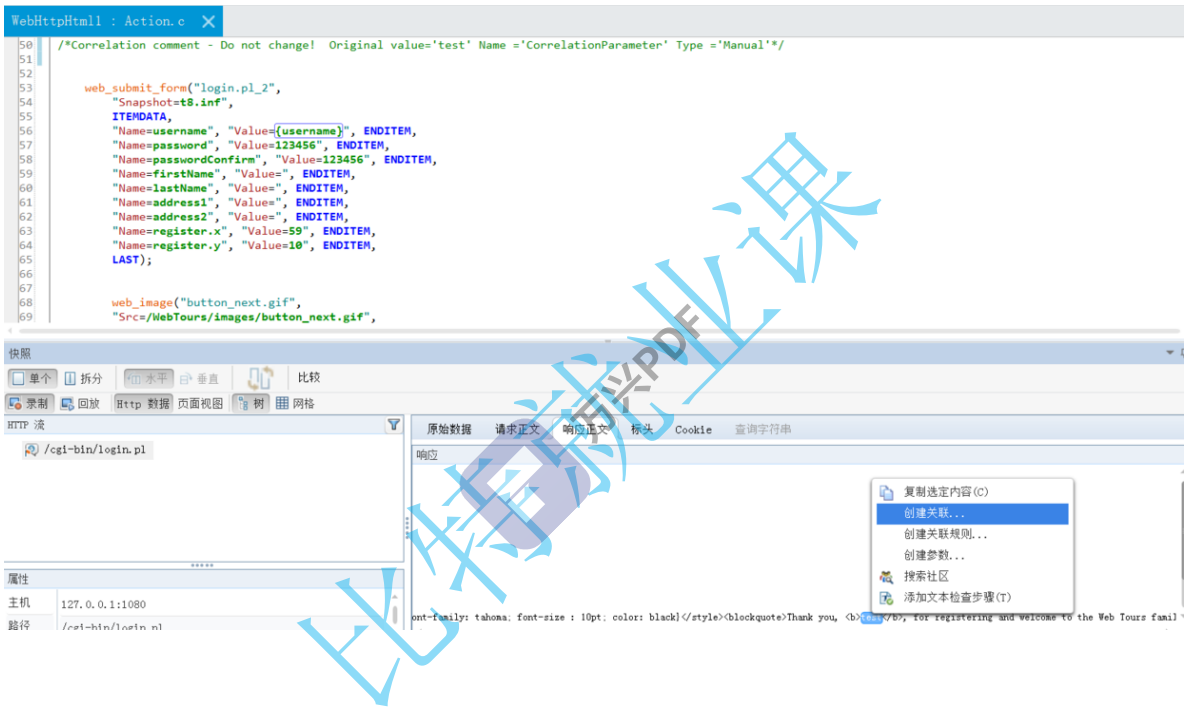
```
web_reg_save_param("BODY",
        "LB=\"MESSAGE\":{\"",
        "RB=\":",
        LAST);


if(strcmp(lr_eval_string("{BODY}"),"msginfo")==0)
{
    lr_end_transaction("柜员登陆",LR_PASS);
     }
    else
     {
        lr_output_message("BODY=[%s]",lr_eval_string("{BODY}"));
         lr_end_transaction("柜员登陆",LR_FAIL);
         }
```

插入检查点

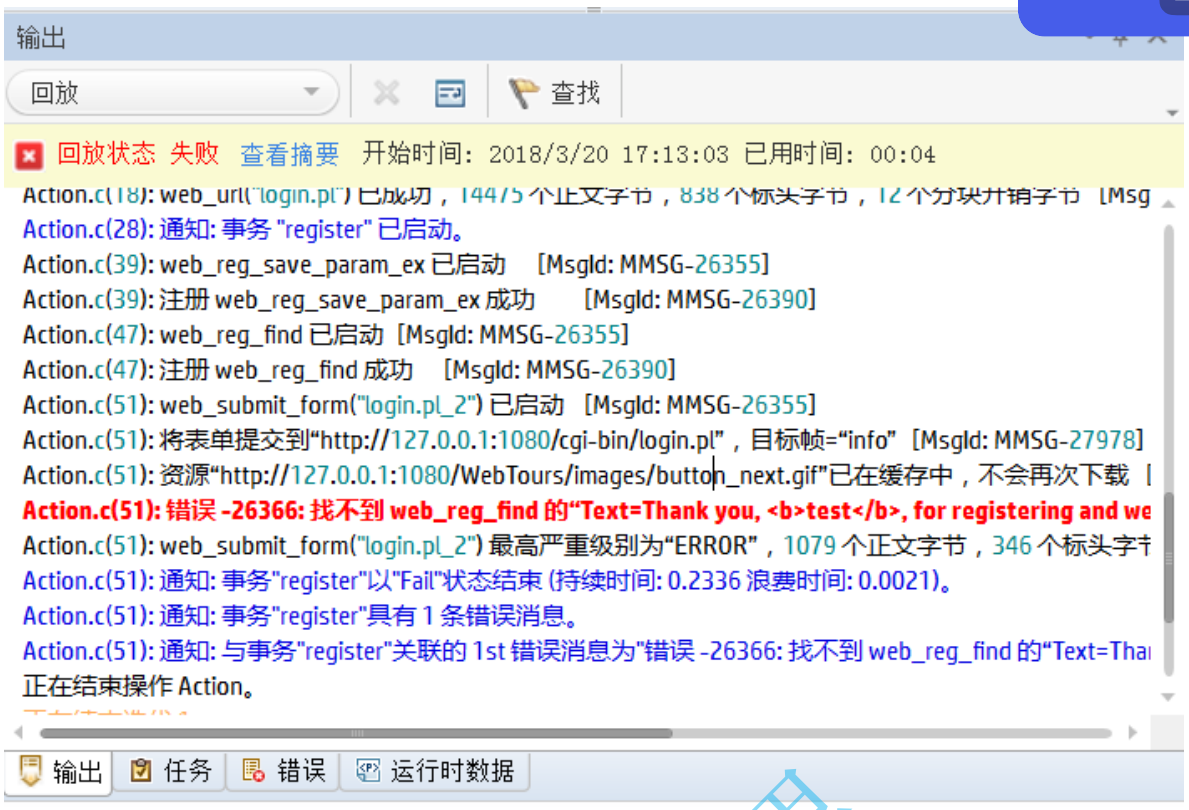
在进行压力测试时，为了检查Web 服务器返回的网页是否正确，VuGen 允许我们插入Text 检查点，这些检查点验证网页上是否存在指定的Text，还可以测试在比较大的压力测试环境中，被测的网站功能是否保持正确。检查点的含义和unittest中的断言功能基本上一致。

通过菜单---查看---快照，可以查看到http数据视图，选择检查的文本，选择添加文本检查步骤，即可添加一个检查点。

```
web_find("Text=Thank you, <b>test</b>, for registering and welcome to the Web
Tours family.",LAST);
web_reg_find("Text=Welcome","Search=Body",LAST);
Welcome--登录成功
Thank you--注册成功
```



此时选择“回放”，可以看到错误提示：



这是因为username我们已经做了参数化，在注册成功后的欢迎页面，已经不是test了。我们做如下改造：

```
web_find("Text=Thank you, <b>{username}</b>, for registering and welcome to the
Web Tours family.",LAST);
再回放一下试试，失败还是成功？为什么？想一想参数中有什么需要修改的？
```

###关联

很多时候，一个项目的请求所以带的参数会来源于前面请求返回的结果，而我们录制的内容，则只是完整地记录当时的请求参数，这通常不是我们想要的。

例如上面的例子中的检查点文本中的test。就是根据输出来显示的，我们可以获取该值来供后续使用。

在上图中，选择test，通过创建关联，可以创建一个关联，创建完毕后如下图：

```
web_reg_save_param_ex(
        "ParamName=CorrelationParameter",
        "LB=you, <b>",
        "RB=<",
        SEARCH_FILTERS,
        "Scope=Body",
        LAST);
//添加输出，进行验证
lr_output_message("BODY=[%s]", lr_eval_string("{CorrelationParameter}"));
```

运行时设置

当完善了测试脚本后，需要对VuGen 的"运行时设置" 进行配置。

在"解决方案资源管理器"视图中选择"运行时设置"，常用的设置内容如下：

1、常规-其他-错误处理：

一般不需要改动，但是在Controller运行时，可以设置"出现错误时仍继续"，来统计错误率

2、常规-其他-自动事务

当我们手工设置了事务时，建议取消该项，以免Controller运行时事务太多

3、常规-运行逻辑-迭代数

根据需要变动，一般在调试脚本时可以设置为多次迭代

4、常规-日志

在调试阶段可勾选“启动日志记录”，脚本稳定时可取消该项

5、常规-思考时间

忽略时会对服务器造成更大的压力，而增加思考时间可以更好的模拟用户使用

6、internet协议-首选项-启用图像或文本检查

不勾选“启用图像或文本检查”，web_find和web_image_check设置的检查点在运行时无效

7、工具-选项-脚本-回放

如果勾选“回放期间显示运行时查看器”，则在运行时会启动浏览器

单机运行测试脚本

经过以上的各个步骤后，脚本就可以运行了。运行脚本可以通过菜单或者工具栏来操作。

执行“运行”命令后，VuGen 先编译脚本，检查是否有语法等错误。如果有错误，VuGen将会提示错误。双击错误提示，VuGen 能够定位到出现错误的那一行。为了验证脚本的正确性，我们还可以调试脚本，比如在脚本中加断点等。

如果编译通过，就会开始运行。然后会出现运行结果。