

# 软件测试技术实验报告

## 实验 7：练习使用 LoadRunner 测试工具进行性能测试

小组成员：刘志才（学号：23020013021）

曲泓勃（学号：23020007099）

翟一航（学号：23020011046）

完成日期：2025 年 1 月 8 日

# 实验目录

- 软件测试技术实验报告.....1
- 实验 7：练习使用 LoadRunner 测试工具进行性能测试.....1
  - 一．实验目的 .....1
  - 二．实验内容 .....1
  - 三．实验要求 .....1
  - 四．实验过程、重点内容及实验结果 .....2
    - 1. Vuser 脚本录制与回放 .....2
    - 2. 服务器设置修改 .....7
    - 3. 脚本优化与参数化 .....10
    - 4. 场景创建与负载模拟.....17
  - 五．实验中遇到的问题、难点及解决方案.....23
  - 六．感想和收获.....24

# 实验 7：练习使用 LoadRunner 测试工具进行性能测试

## 一．实验目的

熟练掌握 LoadRunner 工具的使用方法，学会使用 LoadRunner 进行性能测试。

## 二．实验内容

以下实验内容 2 次课内完成，至少完成下面步骤 1, 3, 4，按照完成度给分数。

1. 给 LoadRunner 的样例程序示例 HP web tours 建立一个 Vuser 脚本并回放该脚本，保证脚本能够正常运行。
2. 更改 HP web tours 的设置，让 HP Web Tours 服务器发射和请求唯一 session ID，并重新录制 Vuser 脚本，并保证此时脚本仍能够正确回放。
3. 更改 Vuser 脚本使之适合在场景中运行。
4. 用 Controller 创建一个场景，模拟 10 个代理同时登录，搜索航班，购买机票，检查行程，登出系统的行为。运行该场景，并观察系统在负载下的性能。

## 三．实验要求

1. 给 LoadRunner 的样例程序示例 HP web tours 建立一个 Vuser 脚本并回放该脚本，保证脚本能够正常运行。
2. 更改 HP web tours 的设置，让 HP Web Tours 服务器发射和请求唯一 session ID，并重新录制 Vuser 脚本，并保证此时脚本仍能够正确回放。{参考 Tutorial/lesson3}
3. 更改 Vuser 脚本使之适合在场景中运行 {参考 Tutorial/lesson4}
4. 用 Controller 创建一个场景，模拟 10 个代理同时登陆，搜索航班，购买机票，检查行程，登出系统的行为。运行该场景，并观察系统在负载下的性能。{参考 Tutorial/lesson5}
5. 本实验 2 次课内完成，按照完成度给分。组内成员可以相互讨论，每组提交一份实验报告并保留所有测试文件以备检查。

实验提交产物：

- 实验报告：除了写出实验目的、实验内容外，还要描述实验主要过程以及实验中遇到的问题及解决方案。每一个实验都要对实验关键步骤和实验结果截图展示。（与前一实验雷同的步骤不需要展示）
- 所有用户测试文件（通常保存在“我的文档”下相应目录中）：不用上交，但是请自行保存到期末考试结束，以备检查。测试文件包括：实验步骤 1, 2, 3 中的 Vuser 脚本，实验步骤 4 中保存的场景文件和运行结果. lrr 文件。
- 实验演示：本实验需要给老师演示，演示结果占分数。不演示实验只写实验报告，本实验成绩打 7 折。

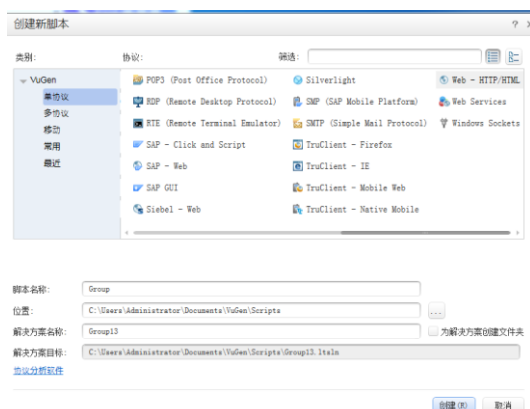
注意事项:

1. 如果在运行 sample 的时候, IE 页面出现错误, 尝试如下操作后重启 IE: 控制面板 → Java → 安全选项卡 → 例外站点列表 → 编辑站点列表, 将 <http://127.0.0.1> 加入到例外站点列表中。
2. LoadRunner 的所有用户测试文件保存在“我的文档”下相应目录中, 请每次实验后自行保存用户文件, 以便下次实验使用 (不保存电脑重启后 C 盘内容可能会重置)。
3. 实验参考资料为 HP\_LoadRunner\_12.02\_Tutorial\_T7177-88037.pdf, 大部分实验都可以根据 Tutorial 上面描述的过程完成, 少数实验需要综合 Tutorial 中的几个实验完成。实验要求中标注了每个实验应该参考 Tutorial 中的哪些部分完成。LoadRunner 学习手册基本为 Tutorial 的简易摘要版本, 可以对照 Tutorial 看 (单独看有可能看不懂), 供参考。
4. 如果在第 4 步实验中找不到 localhost, 请检查在创建新场景 (Scenario) 的时候, 是否清除了“百分比模式”, 是否选择了一个 Vuser 脚本添加到场景中去。

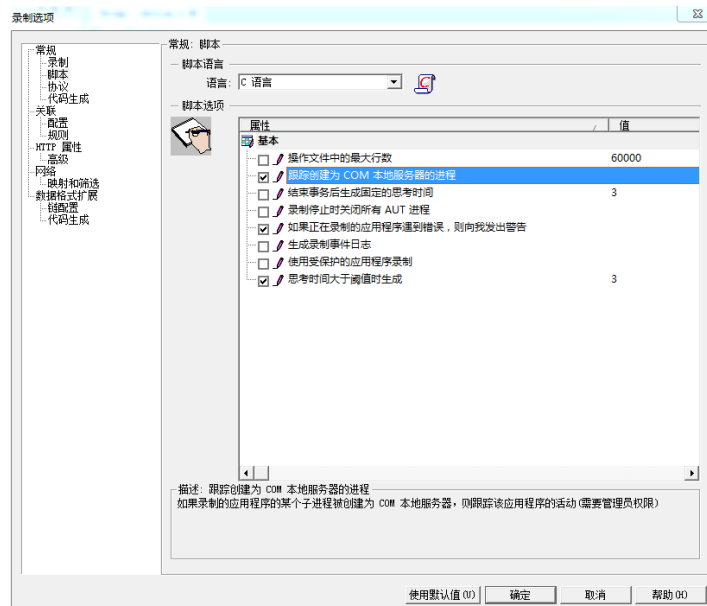
## 四. 实验过程、重点内容及实验结果

### 1. Vuser 脚本录制与回放

- a) HP web tours 是 LoadRunner 中内置的样例程序, 它是一个基于 Web 的旅行代理程序, HP web tours 的用户连接到 web 服务器, 可以搜索航班, 预订航班, 查看航班行程等。使用时先打开服务器 (开始 → 所有程序 → HP Software → HP LoadRunner → Samples → Web → Start HP Web Tours Server), 样例程序运行过程中, 请保证该服务器一直开启不关闭。开启服务器后可以打开 HP web tours 主页 (开始 → 所有程序 → HP Software → HP LoadRunner → Samples → Web → HP Web Tours Application 或者直接在 IE 浏览器中输入 <http://localhost:1080/WebTours>)。注意: 如果 IE 页面出现错误, 尝试如下操作后重启 IE: 控制面板 → Java → 安全选项卡 → 例外站点列表 → 编辑站点列表, 将 <http://127.0.0.1> 加入到例外站点列表中。
  - b) 使用 VuGen 为该样例程序建立一个 Vuser 脚本, 并且采用录制的方式增强该脚本。{ 参考 Tutorial/ lesson1 }
- 1) 创建一个空白的 Vuser 脚本: 打开 VuGen → 创建新脚本 → 选择单协议, 并选择 Web-HTTP/HTML 协议 → 创建了解决方案 Group13, 并在里面创建了一个虚拟用户脚本 test1 → 创建。



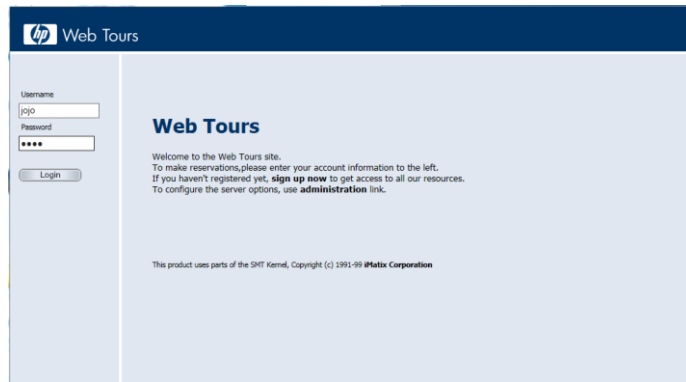
- 2) 录制 Vuser 脚本：本次录制的虚拟用户脚本模拟旅行代理的如下动作：预订一个从 Denver 到 Los Angeles 的航班，然后查看航班行程。录制之前，首先设置录制选项（录制→录制选项→常规→脚本→确保选上了“跟踪创建为 COM 本地服务器的进程”）；



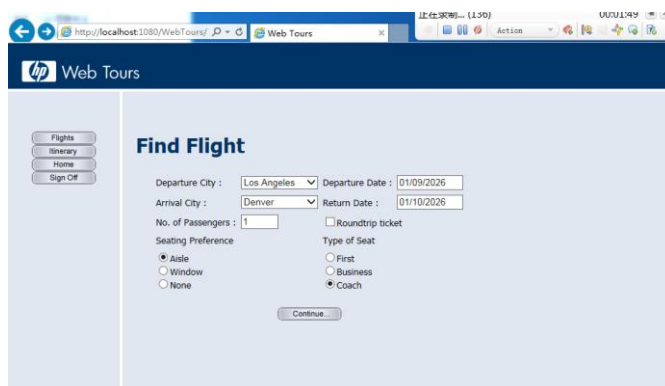
- 3) 开始录制：点录制按钮，录制到操作选择 Action，其他默认，应用程序选 IE，URL 为：http://localhost:1080/WebTours。点开始录制。录制工具栏打开并悬浮在桌面。录制内容如下：



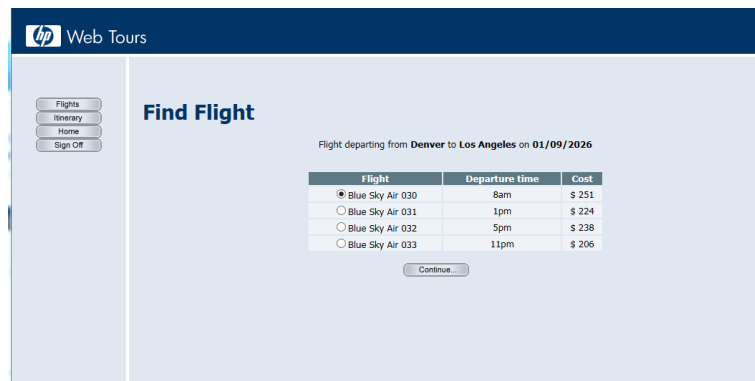
- i. 输入用户名 jojo，密码 bean，登录系统，进入航班详细信息。



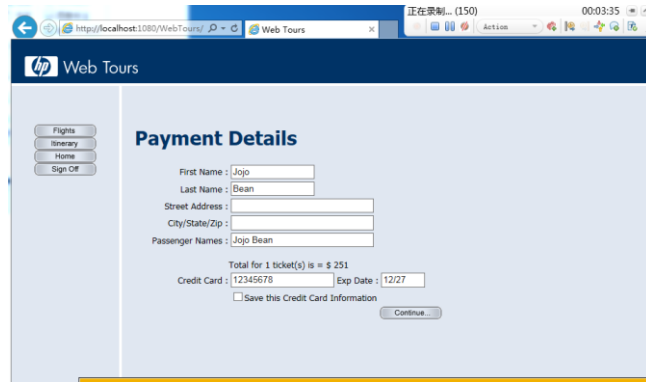
- ii. 点击 Flight，打开搜索航班页面。
- iii. 到达城市选 Los Angeles, Seating Preference 选择 Aisle, 其他默认, 点击 Continue。



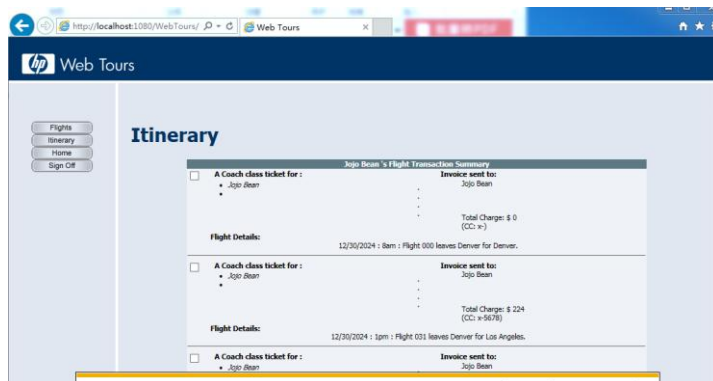
- iv. 选择第一个航班：点击 Continue 接受默认航班选择，进入付款详细信息页面。



- v. 输入付款信息：信用卡号 12345678, Exp Date 12/27, 点击 Continue, 进入收据页面。



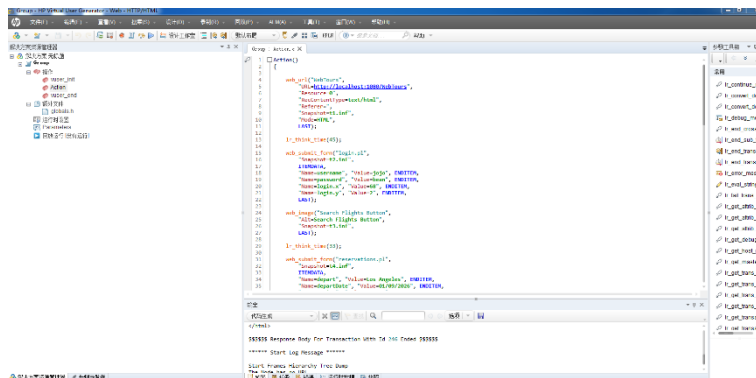
vi. 左侧面板点击 Itinerary，行程页面打开。



vii. 左侧面板点击 sign off，退出。

viii. 关闭浏览器

录制的结果如下：



c) 设置运行环境为：迭代 2 次，迭代时间间隔 5-10 秒之间随机，扩展日志，参数替换，忽略思考时间。以该设置运行刚才录制的脚本，保证脚本运行成功。{参考 Tutorial/lesson2}

- 1) 打开 Vuser 脚本。
- 2) 打开设置运行时环境对话框：回放→运行时设置
- 3) 设置运行逻辑：在运行时环境对话框中，左边面板→常规→运行 逻辑。运行逻辑可以设置 Vuser 脚本的迭代次数。设置迭代次数为 2。



- 4) 设置节奏: 左边面板→常规→节奏。节奏设置允许你设置迭代间的间隔时间。设置成随机时间, 这比较符合真实环境。设置为第三项: 新迭代开始于随机间隔, 5 秒到 10 秒之间。



- 5) 设置日志: 日志的设置指示运行 Vuser 脚本时记录多少信息。选择扩展日志, 并选上参数替换。

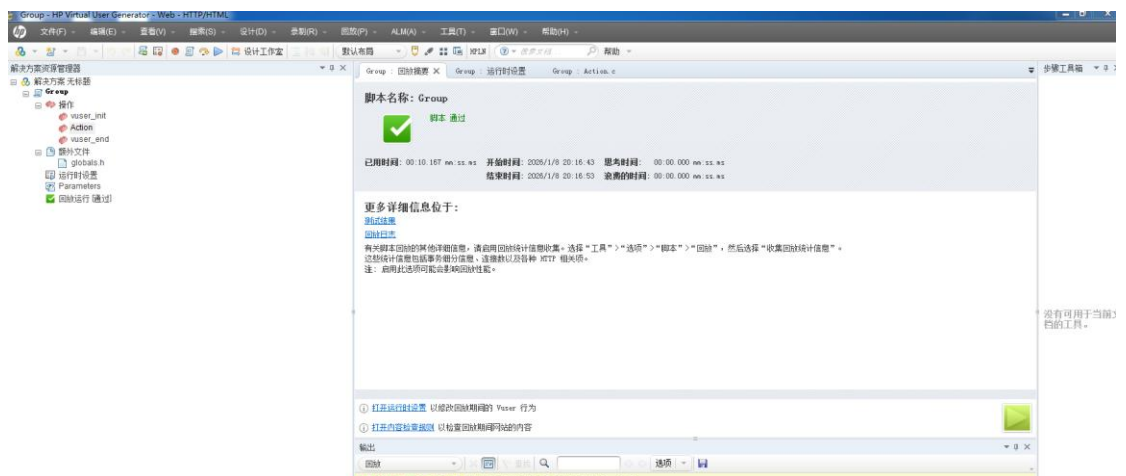


- 6) 观察思考时间设置: 常规→思考时间。保持缺省值忽略思考时间。

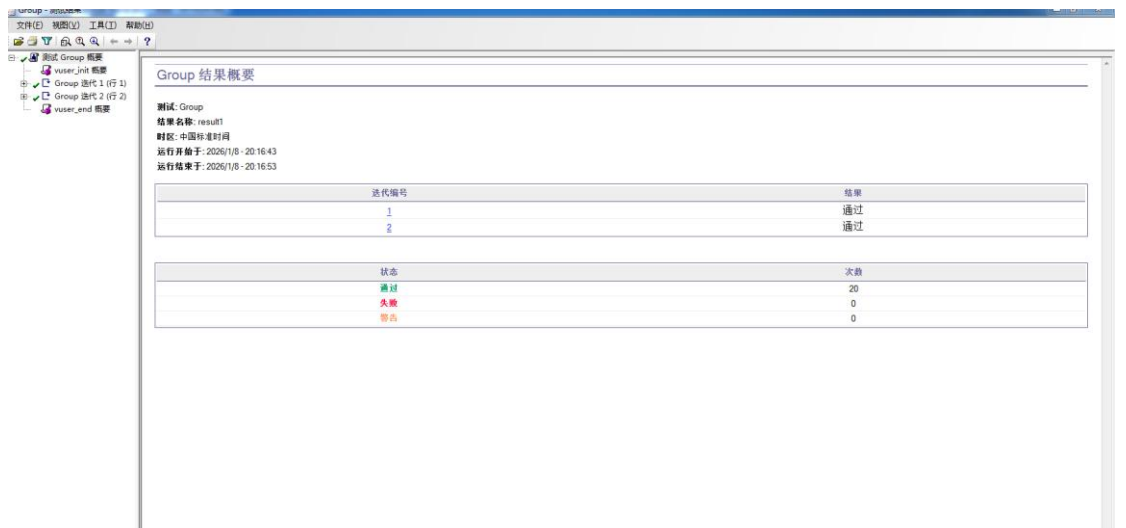




- 7) 保存脚本。
- 8) 点击工具栏上的回放按钮运行脚本
- 9) 执行结果如下：



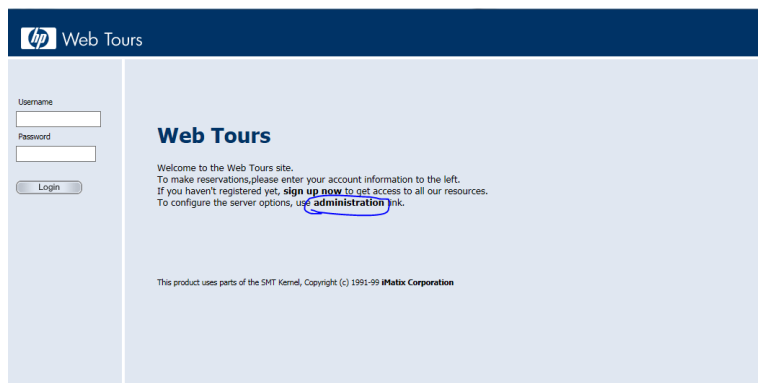
查看测试结果概要并展开，均通过：



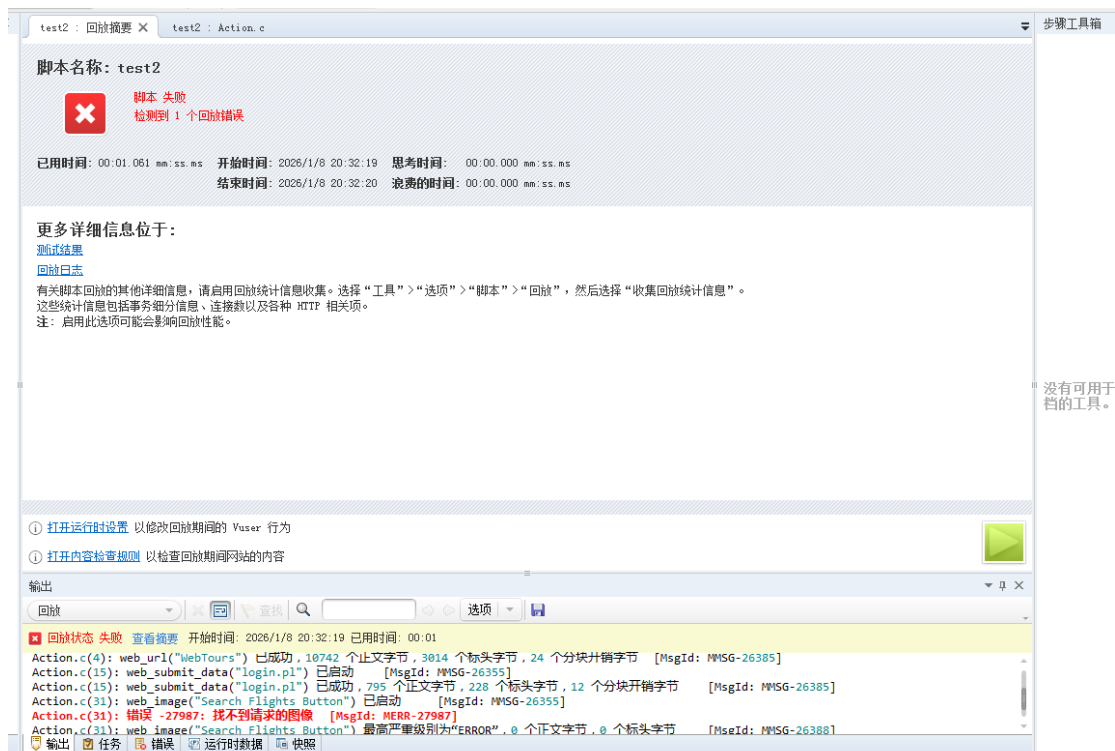
## 2. 服务器设置修改

更改 HP web tours 的设置，让 HP Web Tours 服务器发射和请求唯一 session ID，并重新录制 Vuser 脚本，并保证此时脚本仍能够正确回放。

- a) 打开 HP Web Tours 程序，更改服务器选项为：“Set LOGIN form’ s action tag to an error page”。



- b) 重新回放实验一的脚本，出现错误：

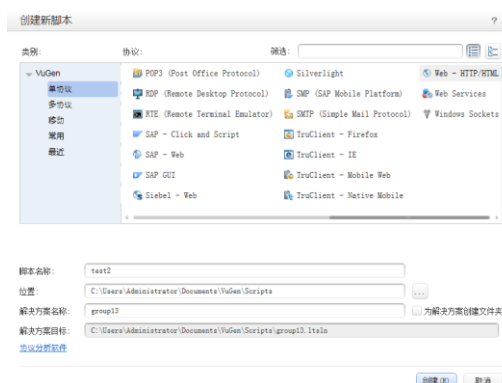


这是因为更改设置后，服务器会为每个 Vuser 赋一个唯一 session ID，每次执行时值都会改变，此时回放 Vuser 脚本会失败。

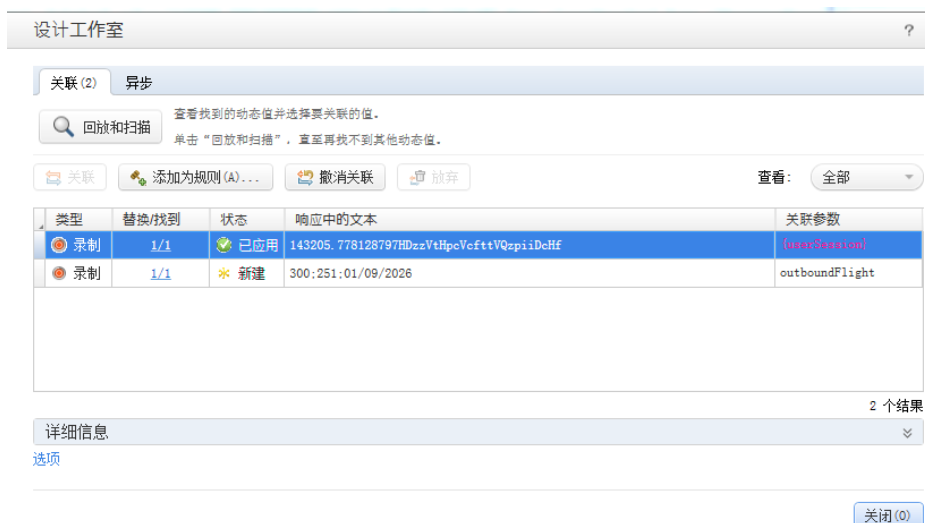
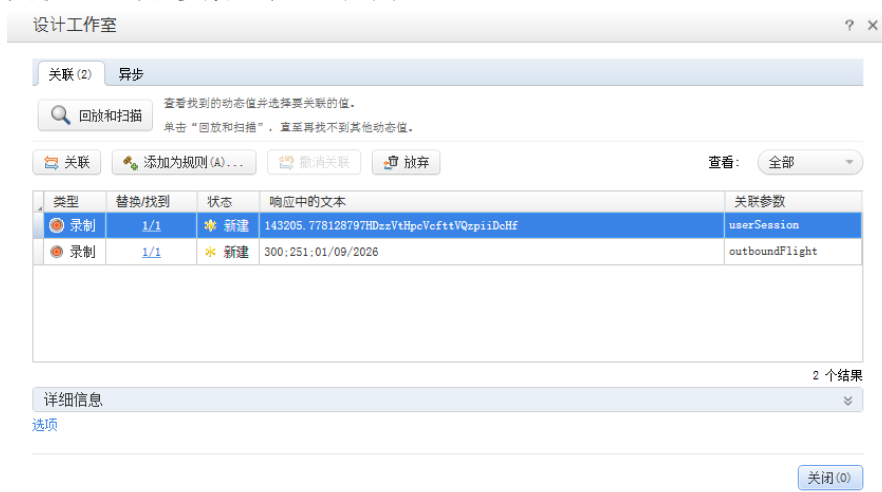
- c) 重新录制与实验步骤 1 相同的脚本（请重新命名脚本名称，与步骤 1 不要相同），看看怎样才能让录制的脚本正确回放。

为了解决这个问题，我们可以使用 VuGen 检测如何能关联到 session ID，可以指示 VuGen 插入一个步骤保存原来的 session ID 到一个参数中。以后每次回放，VuGen 保存新的唯一 session ID 到参数中。当 Vuser 执行脚本中的步骤时，都使用保存的 session ID 而不是原来记录的那个值。过程如下：

- 1) 重新录制一个新的 Vuser 脚本用于处理动态值。另存该脚本为 test2。



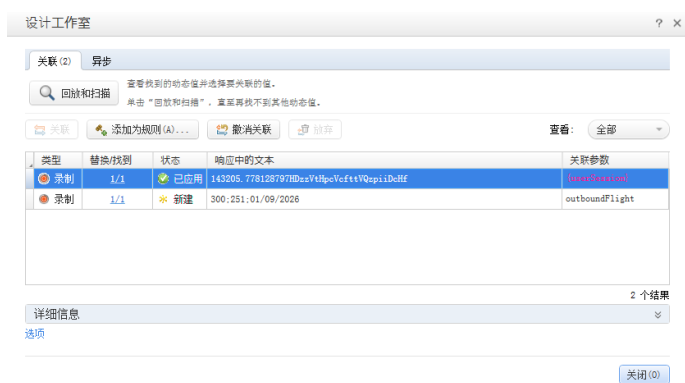
- 2) 录制成功后自动弹出设计工作室，上面显示可关联的值。选择 userSession，点选关联，此时该参数显示“已应用”。



3) 点击回放，运行多次，测试均通过



如果不关联 userSession 则回放不会成功，若这里将 userSession 取消关联，则会回访失败：



此时打开设计→设计工作室，关联 userSession 参数，即可再次运行成功。

### 3. 脚本优化与参数化

更改 Vuser 脚本使之适合在场景中运行

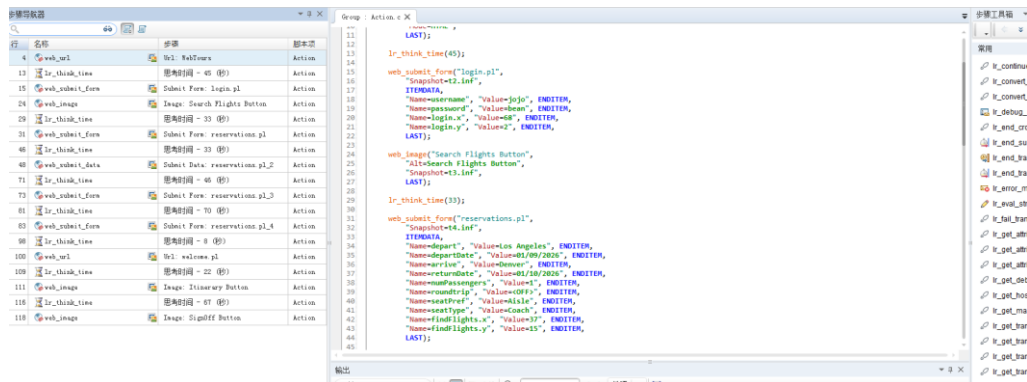
- 本步骤中更改的是步骤 1 中的脚本，请注意如果在步骤 2 中更改了 HP Web Tours 的设置，请更改回原来的设置。



b) 用插入事务的方法测量实验步骤 1 中 Vuser 脚本中的“用户查找和确认机票订单”的时间

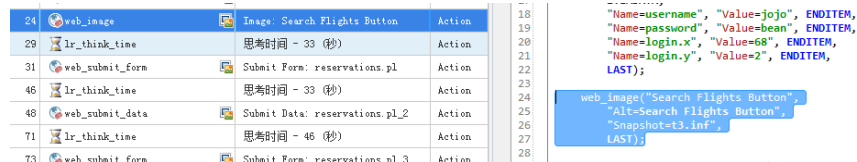
当我们准备应用进行部署时，需要测量特定业务流程的持续时间——登陆、订票需要多长时间等等。每一个业务流程由脚本中的一个或多个步骤或操作组成。在一个 Vuser 脚本中，可以通过把一系列操作包含在一个事务（transaction）中来指定我们想测量的那些操作。当运行一个包含事务的 Vuser 脚本时，LoadRunner 收集运行该事务的时间信息，并将结果显示在图表报告中。我们可以使用这些信息帮助自己决定是否这些应用符合性能需求。我们可以在 Vuser 脚本的任何地方手动插入一个事务。为了把一系列步骤标记为一个事务，可以在第一个步骤之前插入一个事务开始标记，并且在最后一个步骤之后插入一个事务结束标记。

- 1) 打开 test1 脚本。
- 2) 打开步骤导航器，显示步骤导航。

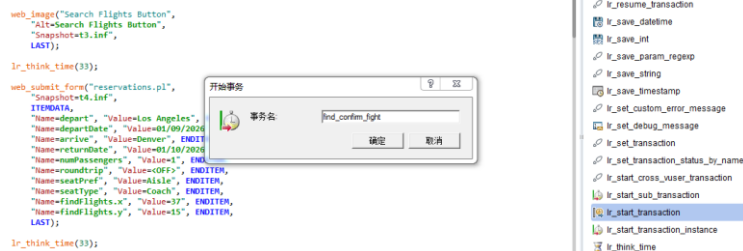


3) 插入事务开始标记

- i. 在步骤导航器中，定位步骤 Image:Search Flights Button。
- ii. 双击该步骤，使得在编辑器中显示该步骤的对应代码 web\_image。



- iii. 点击查看→步骤工具箱，打开右侧步骤工具箱。
- iv. 步骤工具箱，常用下，选择 lr\_start\_transaction，拖拽到编辑器中 web\_image 之前，并为事务起名为 find\_confirm\_flight。



v. 插入效果如下：

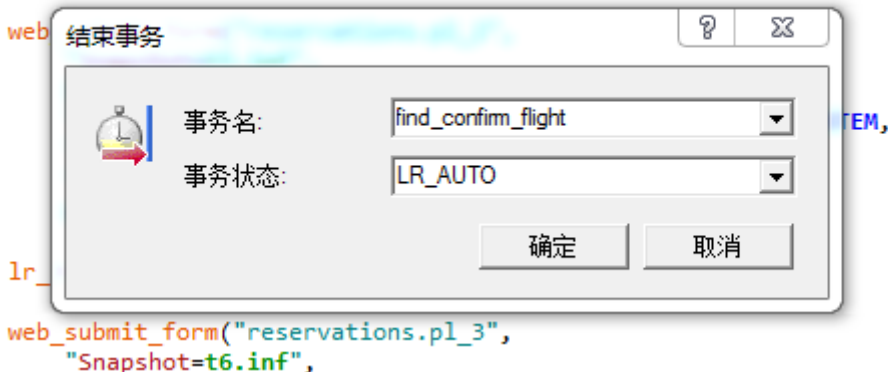
```
lr_start_transaction("find_confirm_flight");

web_image("Search Flights Button",
"Alt=Search Flights Button",
"Snapshot=t3.inf",
LAST);
```

4) 插入事务结束标记

- i. 在步骤导航器中定位步骤 Submit Form:reservations.pl\_2
- ii. 双击该步骤，使得在编辑器中显示该步骤的对应代码 web\_submit\_form
- iii. 在右侧步骤工具箱中，选择 lr\_end\_transaction，拖拽到编辑器中 web\_submit\_form 之后，并确保事务名为 find\_confirm\_flight。VuGen 创建事务结束。

```
lr_think_time(4);
```



```
web_submit_form("reservations.pl_3",
"Snapshot=t6.inf",
```

iv. 插入效果如下：

```
Snapshot=t4.inf",
ITEMDATA,
"Name=depart", "Value=Denver", ENDITEM,
"Name=departDate", "Value=01/09/2026", ENDITEM,
"Name=arrive", "Value=Los Angeles", ENDITEM,
"Name=returnDate", "Value=01/10/2026", ENDITEM,
"Name=numPassengers", "Value=1", ENDITEM,
"Name=roundtrip", "Value=<OFF>", ENDITEM,
"Name=seatPref", "Value=Aisle", ENDITEM,
"Name=seatType", "Value=Coach", ENDITEM,
"Name=findFlights.x", "Value=61", ENDITEM,
"Name=findFlights.y", "Value=9", ENDITEM,
LAST);

lr_think_time(4);

web_submit_form("reservations.pl_2",
"Snapshot=t5.inf",
ITEMDATA,
"Name=outboundFlight", "Value=030;251;01/09/2026", ENDITEM,
"Name=reserveFlights.x", "Value=25", ENDITEM,
"Name=reserveFlights.y", "Value=5", ENDITEM,
LAST);

lr_end_transaction("find_confirm_flight", LR_AUTO);

lr_think_time(20);

web_submit_form("reservations.pl_3",
"Snapshot=t6.inf",
ITEMDATA,
"Name=firstName", "Value=Jojo", ENDITEM,
"Name=lastName", "Value=Bean", ENDITEM,
"Name=address1", "Value=", ENDITEM,
"Name=address2", "Value=", ENDITEM,
"Name=pass1", "Value=Jojo Bean", ENDITEM,
"Name=creditCard", "Value=12345678", ENDITEM,
"Name=expDate", "Value=12/27", ENDITEM,
"Name=saveCC", "Value=<OFF>", ENDITEM,
```

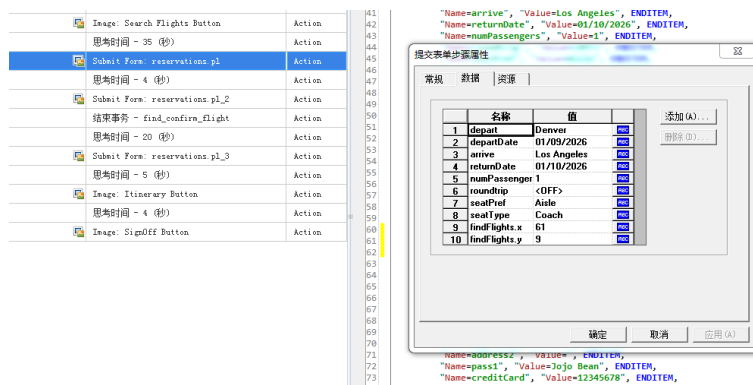
- c) 用参数化的方法更改步骤 1 中的 Vuser 脚本，使得脚本回放时每次迭代选择不同的 Seating Preference 值（例如，第一次迭代 Seating Preference 会使用 Aisle，第二次迭代使用 Window，第三次迭代使用 None 等等）

1) 找到需要参数化数据的地方：

- i. 打开步骤导航器
- ii. 定位到 Submit Form:reservations.pl 步骤
- iii. 右键该步骤，选择显示参数，则该步骤的属性对话框打开。ABC 图标表示每个值都是固定值。

2) 改变固定值为可变值。

- i. 在属性对话框中，选择 seatPref



- ii. 点击 seatPref 的 ABC 图标，参数创建对话框打开，参数名改为 seat，其他不变，oK。图标由 ABC，变为参数图标。

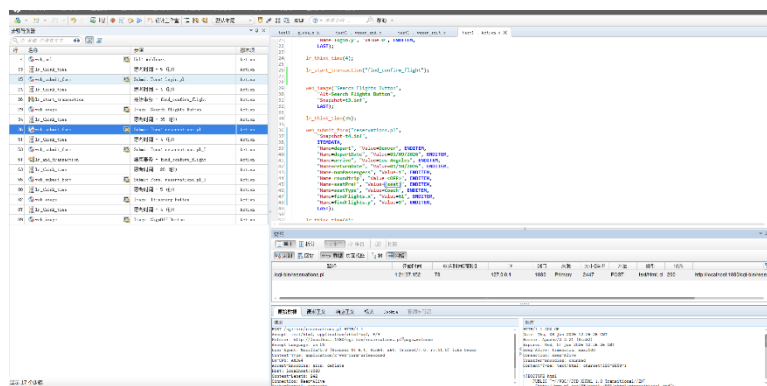


- iii. 点击参数图标，选择参数属性，打开参数属性对话框。

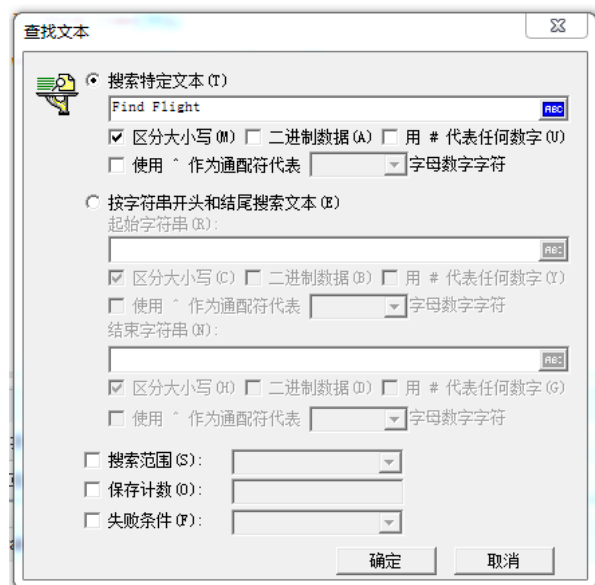
- iv. 为参数指定值：在参数属性对话框中，添加行，值改为 Window. 再添加行，值改为 None。其他默认。



- v. 定义测试如何改变数据
  1. 保持每个迭代更新值（默认），关闭参数属性对话框，OK.
  2. 此时再次回放时，Vuser 在第一次迭代会使用 Aisle，第二次迭代使用 Window，第三次迭代使用 None。
- d) 在脚本中插入文本检查点检查是否“Find Flight”出现在 HP web tours 的预订页面上。
  - 1) 打开步骤导航器
  - 2) 定位到 Submit Form:reservations.pl 步骤
  - 3) 在 VuGen 工具栏，点击显示快照窗格按钮。显示快照面板。
  - 4) 步骤导航器中双击 Submit Form:reservations.pl 步骤，对应快照在快照面板打开。
  - 5) 在步骤工具箱搜索框中输入 web\_reg，从结果中找出 web\_reg\_find 步骤。
  - 6) 将该步骤拖拽到编辑器 web\_submit\_form 函数之前，搜索文本对话框打开。



- 7) 在搜索文本对话框中键入 Find Flight，点击 OK。

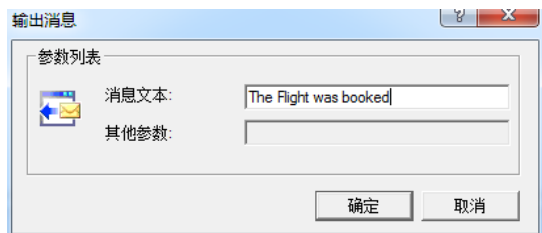


- 8) VuGen 插入搜索文本步骤。  
再次回放脚本时，VuGen 将会搜索文本 Find Flight 并在回放日志中指示是否找到该文本。





- e) 在脚本中退出登录后输出一条消息: The flight was booked.
- 1) 打开步骤导航器
  - 2) 在步骤导航器中, 定位到步骤 Image:SignOff Button
  - 3) 双击 Image:SignOff Button 步骤, 在编辑器中显示对应的函数 web\_image。
  - 4) 找到步骤工具箱→常用→Ir\_output\_message, 将该函数拖拽到编辑器中 web\_image 函数之后。这样会打开输出信息对话框。
  - 5) 在对话框中将信息内容填写为: The flight was booked。点击 OK。



- 6) 保存脚本
- f) 回放修改了的脚本, 查看上述 b, c, d, e 步骤结果是否正确。
- 1) 设置迭代数为 3。
    - i. 打开设置运行时环境对话框: 回放→运行时设置
    - ii. 在运行时环境对话框中, 左边面板→常规→运行逻辑。运行逻辑可以设置 Vuser 脚本的迭代次数。设置迭代次数为 3。
  - 2) 允许图像和文本检查。
    - i. 回放→运行时设置, 打开运行环境对话框
    - ii. Internet 协议→首选项
    - iii. 选择启用图片和文本检查



- 3) 运行脚本，测试通过：
- 4) 定位文本检查：
  - i. 打开输出面板，选择“回放”，在其中使用 CTRL+F 查找“事务”，可以找到事务的相关信息  
三次迭代的事务信息如下图所示：



```
输出
回放
[MsgId: MMSG-26659]
images/home.gif"
Action.c(29): 资源"http://localhost:1080/WebTours/images/signoff.gif"已在缓存中, 不会再次下载 [MsgId: MMSG-26655]
Action.c(29): web_image("Search Flights Button") 已成功, 8516 个正文字节, 1616 个标头字节, 43 个分块开销字节 [MsgId: MMSG-26385]
Action.c(36): web_reg_find 已启动 [MsgId: MMSG-26355]
Action.c(36): 注册 web_reg_find 成功 [MsgId: MMSG-26390]
Action.c(39): web_submit_form("reservations.pl") 已启动 [MsgId: MMSG-26355]
Action.c(39): 通知: 参数替换: 参数"seat"="Aisle" [MsgId: MMSG-27978]
Action.c(39): 将表单提交到"http://localhost:1080/cgi-bin/reservations.pl", 目标帧="" [MsgId: MMSG-27978]
Action.c(39): 资源"http://localhost:1080/WebTours/images/button_next.gif"已在缓存中, 不会再次下载 [MsgId: MMSG-26655]
Action.c(39): 注册的 web_reg_find 对于"Text=Find Flight"成功(计数=1) [MsgId: MMSG-26364]
Action.c(39): web_submit_form("reservations.pl") 已成功, 2447 个正文字节, 255 个标头字节, 12 个分块开销字节 [MsgId: MMSG-26385]
Action.c(56): web_submit_form("reservations.pl_2") 已启动 [MsgId: MMSG-26355]
Action.c(56): 将表单提交到"http://localhost:1080/cgi-bin/reservations.pl", 目标帧="" [MsgId: MMSG-27978]
Action.c(56): 资源"http://localhost:1080/WebTours/images/button_next.gif"已在缓存中, 不会再次下载 [MsgId: MMSG-26655]
Action.c(56): web_submit_form("reservations.pl_2") 已成功, 2699 个正文字节, 255 个标头字节, 12 个分块开销字节 [MsgId: MMSG-26385]
Action.c(64): 通知: 事务"find_confirm_flight"以"Pass"状态结束 (持续时间: 0.5644 浪费的时间: 0.0046)。
Action.c(68): web_submit_form("reservations.pl_3") 已启动 [MsgId: MMSG-26355]
Action.c(68): 将表单提交到"http://localhost:1080/cgi-bin/reservations.pl", 目标帧="" [MsgId: MMSG-27978]

[MsgId: MMSG-26385]
Action.c(36): web_reg_find 已启动 [MsgId: MMSG-26355]
Action.c(36): 注册 web_reg_find 成功 [MsgId: MMSG-26390]
Action.c(39): web_submit_form("reservations.pl") 已启动 [MsgId: MMSG-26355]
Action.c(39): 通知: 参数替换: 参数"seat"="Window" [MsgId: MMSG-27978]
Action.c(39): 将表单提交到"http://localhost:1080/cgi-bin/reservations.pl", 目标帧="" [MsgId: MMSG-27978]
Action.c(39): 资源"http://localhost:1080/WebTours/images/button_next.gif"已在缓存中, 不会再次下载 [MsgId: MMSG-26655]
Action.c(39): 注册的 web_reg_find 对于"Text=Find Flight"成功(计数=1) [MsgId: MMSG-26364]
Action.c(39): web_submit_form("reservations.pl") 已成功, 2448 个正文字节, 255 个标头字节, 12 个分块开销字节 [MsgId: MMSG-26385]
Action.c(56): web_submit_form("reservations.pl_2") 已启动 [MsgId: MMSG-26355]
Action.c(56): 将表单提交到"http://localhost:1080/cgi-bin/reservations.pl", 目标帧="" [MsgId: MMSG-27978]
Action.c(56): 资源"http://localhost:1080/WebTours/images/button_next.gif"已在缓存中, 不会再次下载 [MsgId: MMSG-26655]
Action.c(56): web_submit_form("reservations.pl_2") 已成功, 2700 个正文字节, 255 个标头字节, 12 个分块开销字节 [MsgId: MMSG-26385]
Action.c(64): 通知: 事务"find_confirm_flight"以"Pass"状态结束 (持续时间: 0.4931 浪费的时间: 0.0062)。
Action.c(68): web_submit_form("reservations.pl_3") 已启动 [MsgId: MMSG-26355]
Action.c(68): 将表单提交到"http://localhost:1080/cgi-bin/reservations.pl", 目标帧="" [MsgId: MMSG-27978]

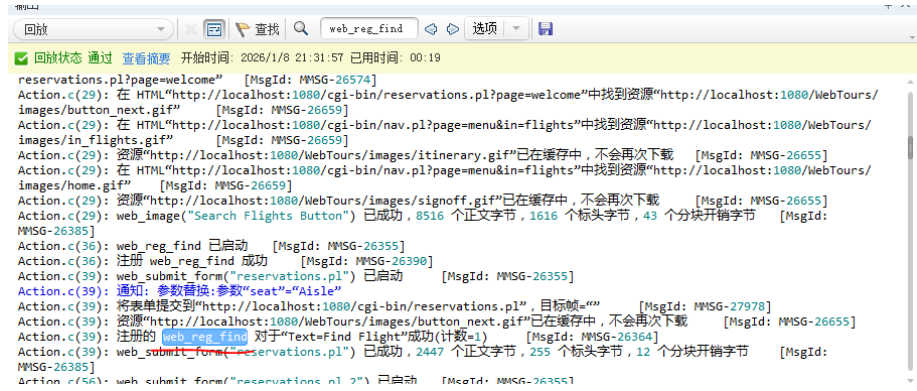
[MsgId: MMSG-26385]
Action.c(36): web_reg_find 已启动 [MsgId: MMSG-26355]
Action.c(36): 注册 web_reg_find 成功 [MsgId: MMSG-26390]
Action.c(39): web_submit_form("reservations.pl") 已启动 [MsgId: MMSG-26355]
Action.c(39): 通知: 参数替换: 参数"seat"="None" [MsgId: MMSG-27978]
Action.c(39): 将表单提交到"http://localhost:1080/cgi-bin/reservations.pl", 目标帧="" [MsgId: MMSG-27978]
Action.c(39): 资源"http://localhost:1080/WebTours/images/button_next.gif"已在缓存中, 不会再次下载 [MsgId: MMSG-26655]
Action.c(39): 注册的 web_reg_find 对于"Text=Find Flight"成功(计数=1) [MsgId: MMSG-26364]
Action.c(39): web_submit_form("reservations.pl") 已成功, 2446 个正文字节, 255 个标头字节, 12 个分块开销字节 [MsgId: MMSG-26385]
Action.c(56): web_submit_form("reservations.pl_2") 已启动 [MsgId: MMSG-26355]
Action.c(56): 将表单提交到"http://localhost:1080/cgi-bin/reservations.pl", 目标帧="" [MsgId: MMSG-27978]
Action.c(56): 资源"http://localhost:1080/WebTours/images/button_next.gif"已在缓存中, 不会再次下载 [MsgId: MMSG-26655]
Action.c(56): web_submit_form("reservations.pl_2") 已成功, 2698 个正文字节, 255 个标头字节, 12 个分块开销字节 [MsgId: MMSG-26385]
Action.c(64): 通知: 事务"find_confirm_flight"以"Pass"状态结束 (持续时间: 0.4756 浪费的时间: 0.0043)。
Action.c(68): web_submit_form("reservations.pl_3") 已启动 [MsgId: MMSG-26355]
Action.c(68): 将表单提交到"http://localhost:1080/cgi-bin/reservations.pl", 目标帧="" [MsgId: MMSG-27978]
Action.c(68): 在 HTML"http://localhost:1080/cgi-bin/reservations.pl"中找到资源"http://localhost:1080/WebTours/images/bookanother.gif" [MsgId: MMSG-26659]
Action.c(68): web_submit_form("reservations.pl_3") 已成功, 3285 个正文字节, 658 个标头字节, 12 个分块开销字节 [MsgId: MMSG-27978]
```

这里记录了每一个迭代中事务“find\_confirm\_flight”的启动处和结束处，以及事务的持续时间如图所示。

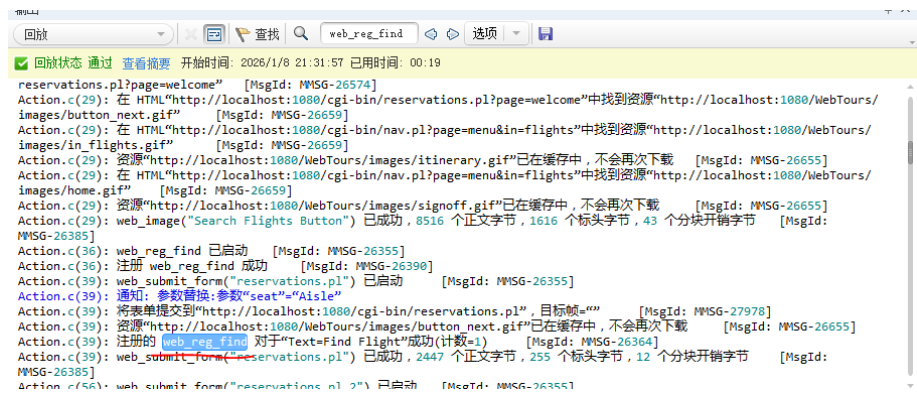
- ii. 查找“参数”，可以找到参数化的相关信息，可以发现每次迭代依序使用不同的参数。

第一次迭代中使用的参数以及第二次迭代开始时通知获取参数的信息如下：

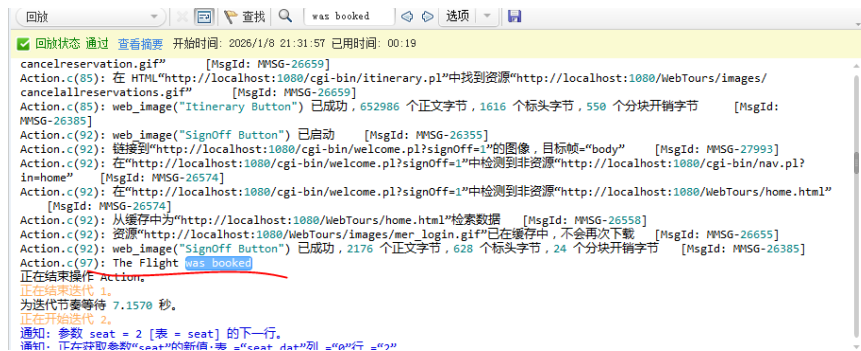
第二个迭代参数为 Window，第三个迭代参数为 None。



- iii. 查找 web\_reg\_find，可以看到文本检查的相关信息。  
第一次迭代中文本检查的启动处、注册成功信息、以及计数信息如下图所示，  
第二和第三次迭代也是类似的



- iv. 查找 “The flight was booked” 可以找到刚才插入的这条信息。



- v. 保存测试。

## 4. 场景创建与负载模拟

用 Controller 创建一个场景，模拟 10 个代理同时登陆，搜索航班，购买 机票，检查行程，登出系统的行为。运行该场景，并观察系统在负载下的 性能。

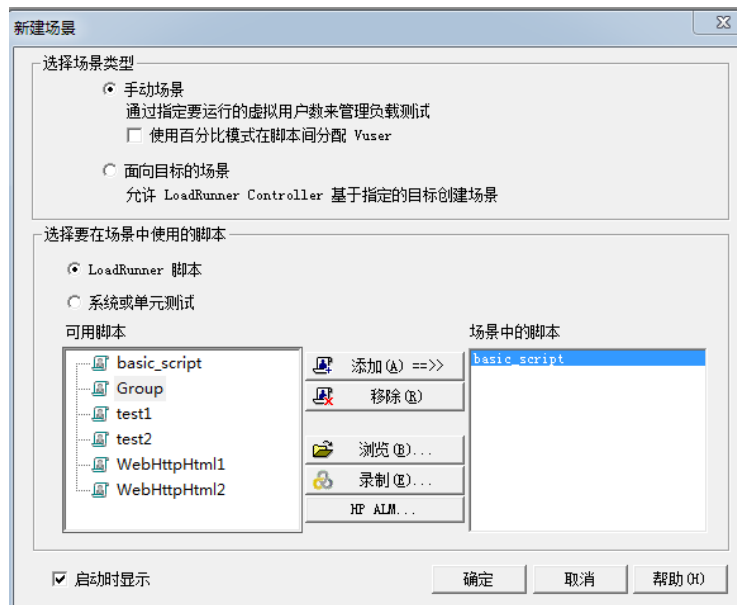
- a) 使用 Controller 创建手工场景，场景使用的 Vuser 可以用步骤 3 中录制的脚本，也可以用<LoadRunner Installation>\Tutorial\basic\_script 标准脚本。场景存放在默认位置，名字可以自取。

- 1) 打开 Controller，此时控制器打开并显示新的场景对话框。

2) 选择场景类型：有两种场景类型

- i. 手动场景可以让我们控制 Vuser 的数量以及他们运行多少次，可以让我们测试自己的应用能够同时运行多少 Vusers。勾选百分比模式可以让用百分比模式在脚本间分配 Vuser。
- ii. 面向目标的场景允许我们基于指定的目标创建场景。例如，我们可以依据某个事物的响应时间或每秒事务数等确定目标，LoadRunner 自动基于我们给定的目标创建脚本。

本次我们采用手工创建场景，清除百分比模式。



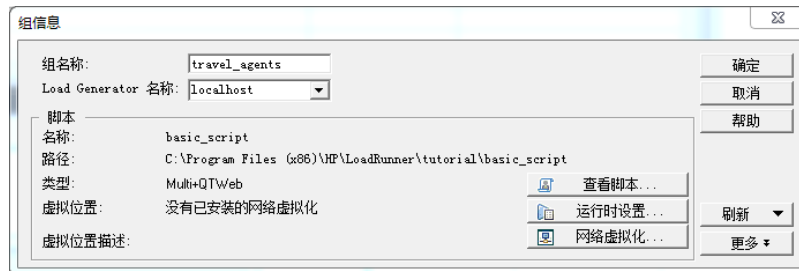
3) 向负载测试增加 Vuser 脚本。

在该教程中，我们只使用一个 Vuser 脚本建模一组用户，该组用户动作一致。如果要更精确的模拟更通用的用户配置，则应创建一些不同的 Vuser 用户组，每个用户组运行几个不同的脚本。

- i. 为达到此目的，我们需要提供一个样例脚本，这里使用教程提供的的样例脚本，该脚本在\Tutorial 文件夹中，脚本名为 basic\_script，将其添加到场景面板的脚本中。
  - ii. 点击 OK。LoadRunner 控制器打开，显示新场景的设计选项卡。（设计选项卡在控制器左下角。）控制器的设计选项卡是设计负载测试主要界面，设计选项
- b) 在该场景下加入 10 个 Vuser 负载，负载产生器设置为 localhost。负载加载的方式为：同时初始化所有 Vuser；每隔 5 秒启动 2 个 Vuser，共启动 10 个；运行 3 分钟后，每隔 5 秒终止 2 个 Vuser 直到所有 Vuser 都终止。（这是检查时的要求，调试时运行时间可自行缩短，已缩短调试需要的时间。）

1) 修改脚本的组名为 travel\_agent

- i. 在场景组面板，查看 basic\_script 的组名。
- ii. 选择 basic\_script 并点击详细信息按钮，弹出组信息对话框，在对话框中可以修改组名为 travel\_agent。点击 OK。



2) 在场景组下添加 10 个 Vuser 负载，负载产生器设置为 localhost

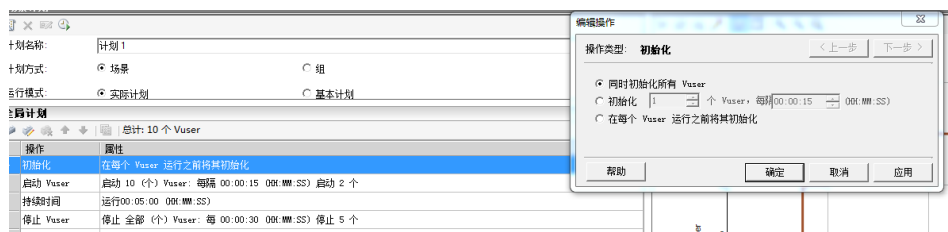
i. 系统已默认设置好：

3) 设置负载加载的方式：

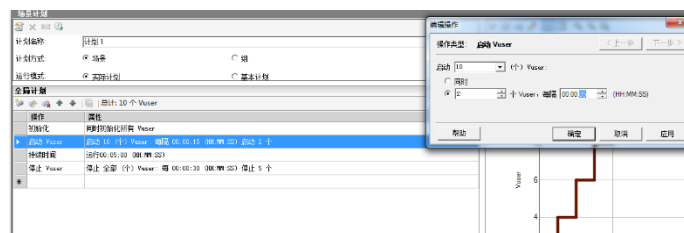
我们可以在全局计划网格中或者在交互计划图表中设置如何让 Vuser 开始运行，结束运行，以及持续时间。在其中一边设置后，另外一边也会相应变化。

在网格中双击需要设置的行，即可打开相应对话框，然后进行相应设置，具体表述和截图如下：

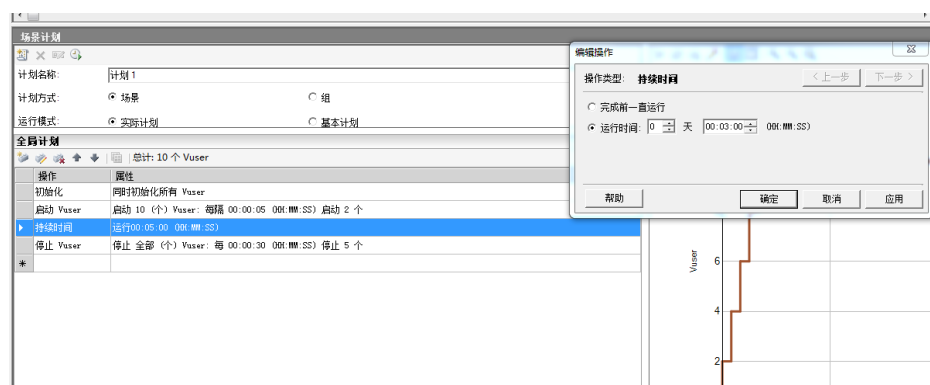
i. 双击“初始化”：设置同时初始化所有 Vuser；



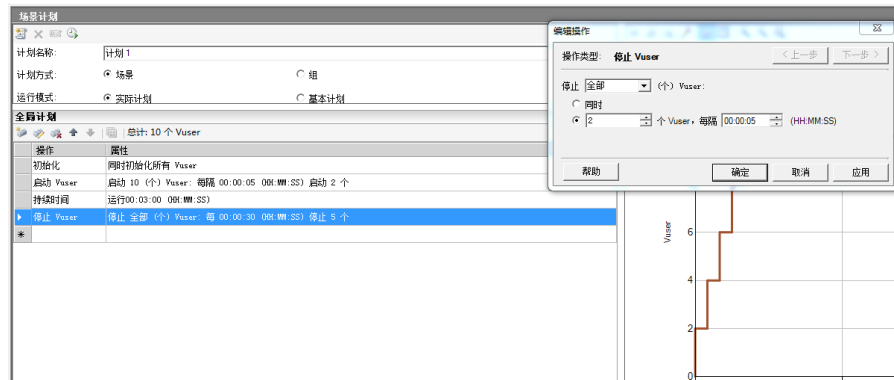
ii. 双击“启动 Vuser”：设置每隔 5 秒启动 2 个 Vuser，共启动 10 个；



iii. 双击“持续时间”：设置运行 3 分钟；



iv. 双击“停止 Vuser”：设置每隔 5 秒终止 2 个 Vuser 直到所有 Vuser 都终止



4) 最终设置的效果如下:

全局计划	
总计: 10 个 Vuser	
操作	属性
初始化	同时初始化所有 Vuser
启动 Vuser	启动 10 (个) Vuser: 每隔 00:00:05 (HH:MM:SS) 启动 2 个
持续时间	运行 00:03:00 (HH:MM:SS)
停止 Vuser	停止 全部 (个) Vuser: 每隔 00:00:05 (HH:MM:SS) 停止 2 个

c) 进行场景运行时设置: 允许思考设置为“使用录制思考时间的随机百分比”, 启用日志, 日志选择选择“始终发送消息”, 扩展日志选择“服务器返回的数据”。

1) 打开运行时设置:

- i. 在控制器上, 点击设计选项卡
- ii. 在场景组面板, 选择 travel\_agents 组。
- iii. 点击场景组工具栏上的运行时设置按钮, 运行时设置对话框打开。

运行时设置可以允许我们模拟不同的用户活动和行为。他们 包括:

1. 运行逻辑: Vuser 重复一套操作的次数。
2. 节奏: 重复操作之前等待的时间。
3. 日志: 测试期间收集信息的级别。第一次运行场景时, 推荐产生日志信息以确保如何发生错误获得调试信息。
4. 思考时间: 用户在操作步骤之间思考的时间。
5. 速度模拟: 用户使用不同的网络连接例如调制解调器, DSL, 光纤等
6. 浏览器模拟: 用户使用不同浏览器查看应用的性能。
7. 内容检查: 自动检测用户定义的错误。假设你的应用在发生错误时发送一个自定义页面, 这个自定义页面通常都会包含 ASP Error 的字样。你需要搜索服务器返回的所有页面以查看是否含有 ASP Error 字样。你可以使用 内容检查运行时设置设置 LoadRunner 在测试中自动的 搜索这个文本。LoadRunner 搜索该文本, 如果搜索到, 则产生一个错误。在场景运行时, 你能够识别内容检查错误。

2) 允许思考时间

在运行时设置对话框, 点击常规→思考时间, 选择重播思考时间, 选择“使用录制思考时间的随机百分比”, 其他默认。





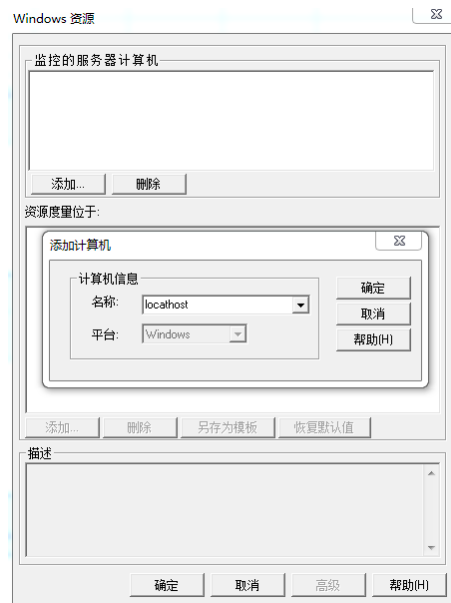
- 3) 允许日志。
  - i. 运行时设置对话框，常规→日志，选择启用日志。
  - ii. 日志选项里，选择“始终发送消息”
  - iii. 点击“扩展日志”，选择“服务器返回的数据”。



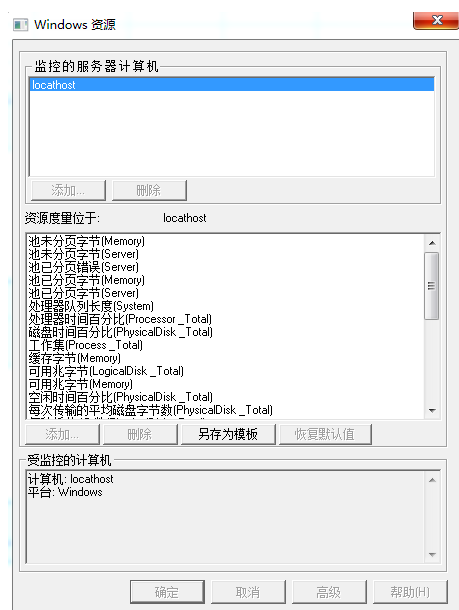
注：初始的调试运行后，可以关闭扩展日志。

- iv. OK. 关闭运行时设置对话框。
- v. 保存场景。
- d) 配置并激活 Windows 资源监视器（localhost 为监视器）
  - 1) 选择 Windows 资源监视器。
    - i. 点击控制器底部的运行选项卡
    - ii. 在 Windows 资源图表上点击右键选择“添加度量”。Windows 资源对话框打开。
  - 2) 选择被监视的服务器。
    - i. 在“监控的服务器计算机”部分点击“添加”，打开增加计算机对话框。

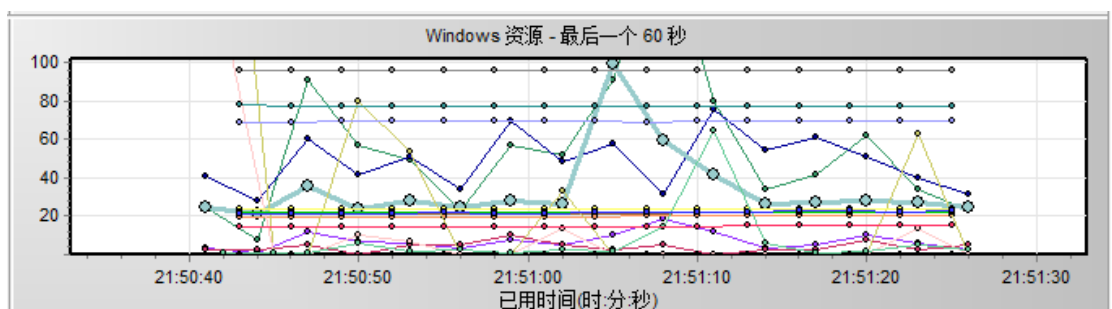
- ii. 名称栏键入 localhost（负载产生器运行的计算机的服务器名或者 IP 地址）
- iii. 平台选择该机器运行的平台。
- iv. 点击 OK。



- v. “资源度量 位于: localhost” 下会列出默认 Wiindows 资源测量。



- 3) 激活监视器：点击 OK 关闭对话框，激活监视器，激活页面如下：

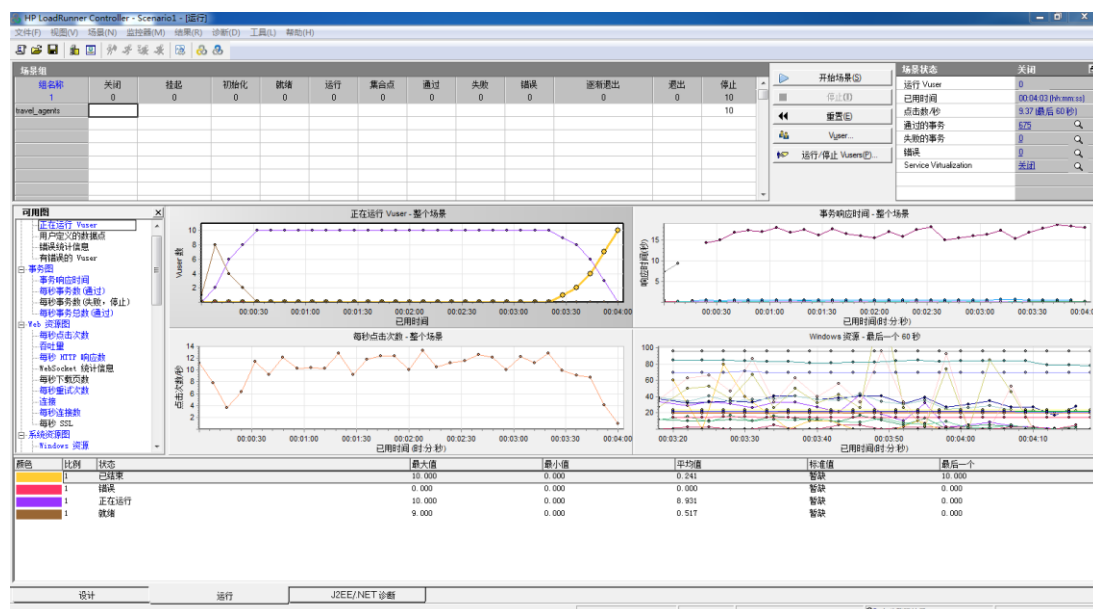




e) 运行并观察场景的运行 {参考 Tutorial/lesson6}

1) 打开控制器运行视图, 点击控制器底部的运行选项卡。

观察到在场景组面板“关闭”一栏有 10 个 Vuser。因为场景还没有运行, 所以其他所有的计数器都还是 0, 图表显示区的所有图 (除了 Windows 资源图) 都是空的: 当我们开始运行场景时, 这些图和计数器将会显示信息。



2) 开启场景。

点击“开始场景”或者选择“场景”→“开始”, 开始运行场景, 运行时长为 3 分钟, 3 分钟后会显示运行结果如下: 如果我们是第一次运行这个教程, 控制器会直接开始场景, 结果 文件会自动存储到负载产生器的 temp 文件夹中。

3) 各个性能图表的含义:

运行选项卡上的图显示面板默认状态下显示下面一些图表:

- 正在运行的 Vuser-整个场景图。显示在一个给定时间里正在运行的 Vuser 数目。
- 事务响应时间-整个场景图。显示每个事务完成所需要的时间。
- 每秒点击次数-整个场景图。显示场景运行时每秒内 Vuser 向 Web 服务器发送的 HTTP 请求数目 (点击次数)。
- Windows 资源图。显示场景运行时 Windows 资源测量值。

## 五. 实验中遇到的问题、难点及解决方案

问题 1: 实验一中“submit from: reservations.pl\_2”不存在, 而是错误地存在 “submit data: reservations.pl\_2”。

解决方法: 重新录制

## 六. 感想和收获

通过本次 LoadRunner 性能测试实验，我们深刻体会到性能测试在软件质量保障中的关键作用。LoadRunner 不仅是一个强大的工具，更需要测试人员对系统架构、网络通信和用户行为有深入理解，才能设计出真实有效的测试场景。实验中，从脚本录制、参数化、事务插入到场景设计的完整流程，让我们认识到性能测试是一项系统工程——脚本优化直接影响测试精度，参数化和关联处理解决了动态数据问题，而场景设计则需综合考虑负载策略和监控指标。当看到 10 个虚拟用户并发时系统各项性能指标的变化，我们直观感受到高并发对服务器资源的压力，这也启示我们未来开发中需重点关注代码效率和资源管理。此次实验不仅提升了工具操作技能，更培养了性能测试的整体思维，为今后从事软件质量工作奠定了基础。