



**Silk Performer 2008
Training Courseware**

By

Softsmith Infotech
www.softsmithinfotech.com

Table of Contents

TABLE OF CONTENTS	2
INTRODUCTION	3
Need For Load Testing.....	3
Silk Performer Features.....	3
Load Test Process Steps.....	3
LOAD TEST PLANNING.....	4
Functional Testing Vs Load Testing.....	4
Load Testing Checklist.....	4
Load Test Guidelines.....	5
PREPARING THE WORK BENCH.....	5
PROFILES	8
CREATION OF SCRIPTS	9
PARAMETERIZATION OF SCRIPTS.....	12
WORKLOADS	23
ANALYSIS.....	28
ACTUAL BASE LINE REPORT.....	32

Introduction

Need For Load Testing

Any multi-user application needs to face the concurrent access some day or the other. Before deploying the application and then exposing the application for multiple users it is better we test it and then do the deployment. This process is load testing.

Minimal Infrastructure - We cannot gather hundreds or thousands of people to carry out concurrent user tests and this will not be possible for large number of users for longer time

Reliable - **Tests** perform precisely the same operations each time they **are** run, thereby eliminating human error.

Repeatable - We can test how the application reacts after repeated execution of the same operations, for longer durations for many days

Programmable - We can program sophisticated tests that bring out hidden information.

Comprehensive - We can build a suite of tests that covers every feature in our application.

Reusable - We can reuse tests on different versions of an application, even if the user interface changes.

Silk Performer Features

The following are the key features of SP (Silk Performer).

Use Add-ins to support multiple environments. These add-ins are the ones that enable SP to recognize different protocols.

- Record vuser scripts and debug scripts (Scripts under Projects)
- Configure and Run scenarios (Work Load under Projects)
- Schedule and run scripts (Work Load under Projects)
- Run tests in a distributed manner (Agents under Projects)
- Analyze graphs (Explore Time Series, Monitor Server under Results)

Load Test Process Steps

- Plan
- Create scripts
- Create scenarios
- Run & monitor scenarios
- Analyze results

Load Test Planning

- Identify most frequently used transactions
- Identify potential number of users
- Identify potential number of concurrent users
- Apply 10:1 or 5:1 ratio for logged-in Vs concurrent users
- Identify the production platform size and configuration
- Identify the data to be used for testing
- Identify the different real-time usage combinations of test scenarios
- Identify the load test run duration
- Identify what kind of information is transmitted between server and client
- Plan load testing only after functional stability of the product is achieved
- Discuss with other stakeholders like network admin, database admin, server admin and others on what information is required for them
- Chalk out the software configurations/settings for web server, app server and database server

Functional Testing Vs Load Testing

- If preconditions are met and steps are followed, function test results are defined. Load test results are always unpredictable
- Functional test results do not change more than 5% when moved from one configuration to the other. Load Test results may even nose-dive!
- Functional test happens on a daily basis; but load test is not that frequent
- Load test results depend on database volume as well and they change when number of users change

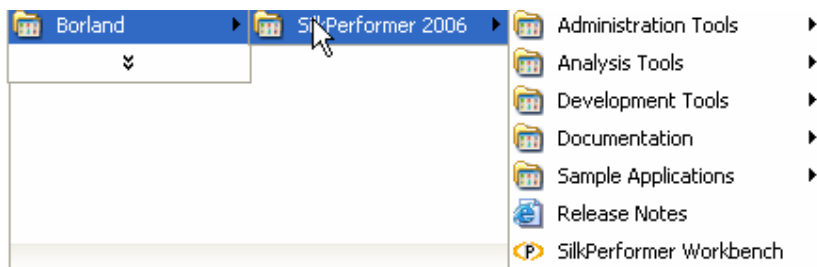
Load Testing Checklist

- Do we have the near-production hardware configuration? If not what is the delta between test hardware and production hardware?
- Is the tool capable of recording the requests based on the protocols used by the application (e.g. HTTPS) and able to replay the same?
- Is the product functionally cleared before load testing?
- Can we get numbers on the user counts from customer, based on past records?
- Is the data pool containing unique data?
- Is the trace log enabled for database and web servers?
- Are the requests distributed equally to different boxes? Is there a load balancer?
- Is there a facility in the tool to mimic different line speeds?
- Is there a facility in the tool to mimic different browser versions?
- Is there a facility in the tool to selectively log messages?
- Is there a facility in the tool to export the data in xls format?
- Is there a facility in the tool to auto-synchronize concurrent requests?
- If the application uses queues, the queue size must be monitored during test runs.
- Do the tests need runs with and without proxy servers?
- Do the tests need runs with and without firewalls?

Load Test Guidelines

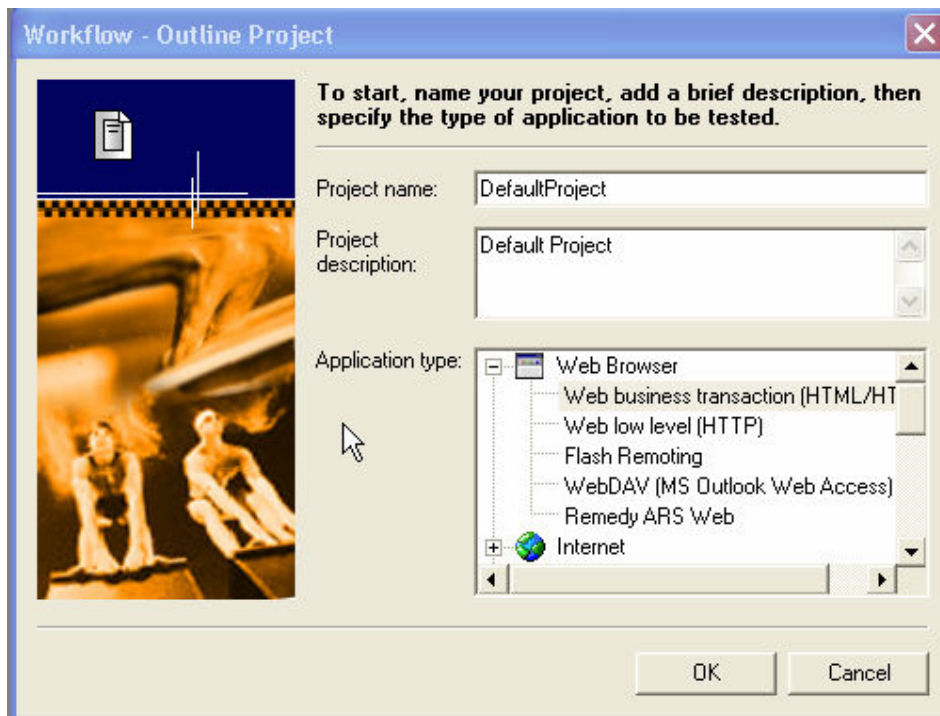
- Number of users Vs response time must not be linear
- Stress test needs to be done for shorter durations and not for longer durations
- To the extent possible, let the data pool contain more unique data than what is needed
- The load generating client machines must not be operated at capacities beyond 80% for CPU and memory
- Avoid enabling detailed log information in the tool which will take more disk IO in the client machines
- Script must be parameterized for accessing the same application with different configurable URLs. So if the application is moved from one box to the other, the script can be reused
- Wherever needed, use rendezvous points to synchronize the requests before any form submission actions in the script. This ensures the simultaneous hits at the time of form submission
- If there is a possibility, disable downloading image files as image files are not downloaded every time in real time usage.
- Check the consistency of response time over a period of elapsed time and compare it with different test runs
- All successful requests must have been submitted and the log files must match. If the requests trigger data base operations, the same must have been recorded in database.
- The queues size must be minimal at any given point of time.
- Most of the time the database and the business logic layer need to be doubted first before the web server is doubted.
- Refer to Microsoft web URLs:
<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnpag/html/SCAG-CH08.asp> for finer details.

Preparing the Work Bench

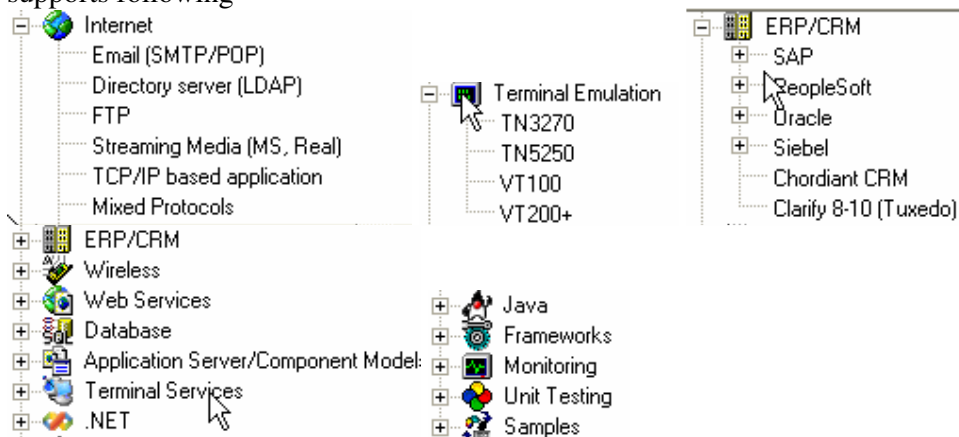


Go to SilkPerformer Workbench, create a project in the silk performer

Each project can have multiple scripts, each script being one business scenario. Each Script can have multiple transaction, Base transactions are Tinit, TMain, TShutdown, In between Tinit and TShutdown, We can have as many transactions as possible.



Application type needs to be selected based web or Sap or Citrix etc. Silk performer supports following



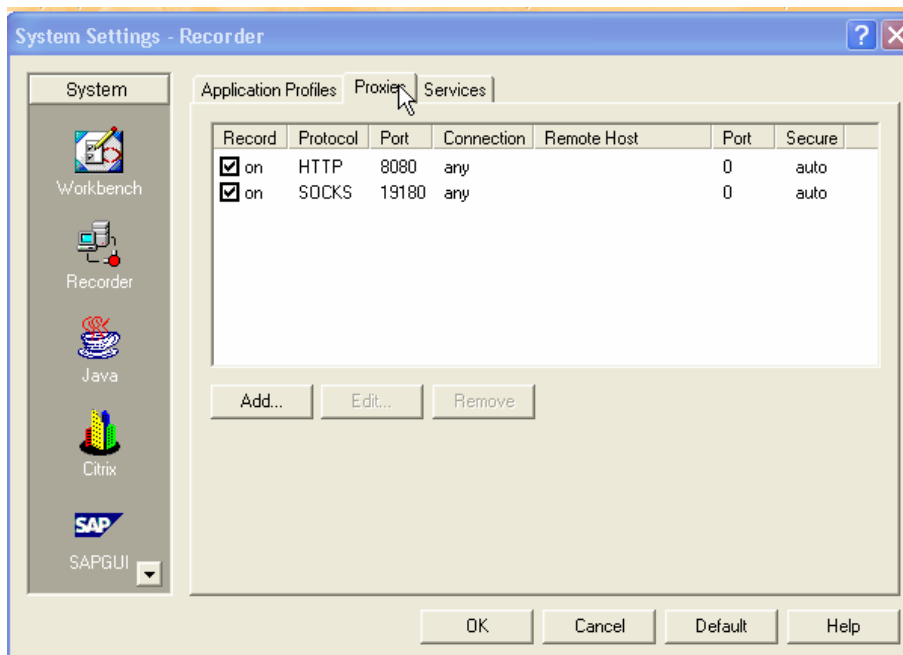
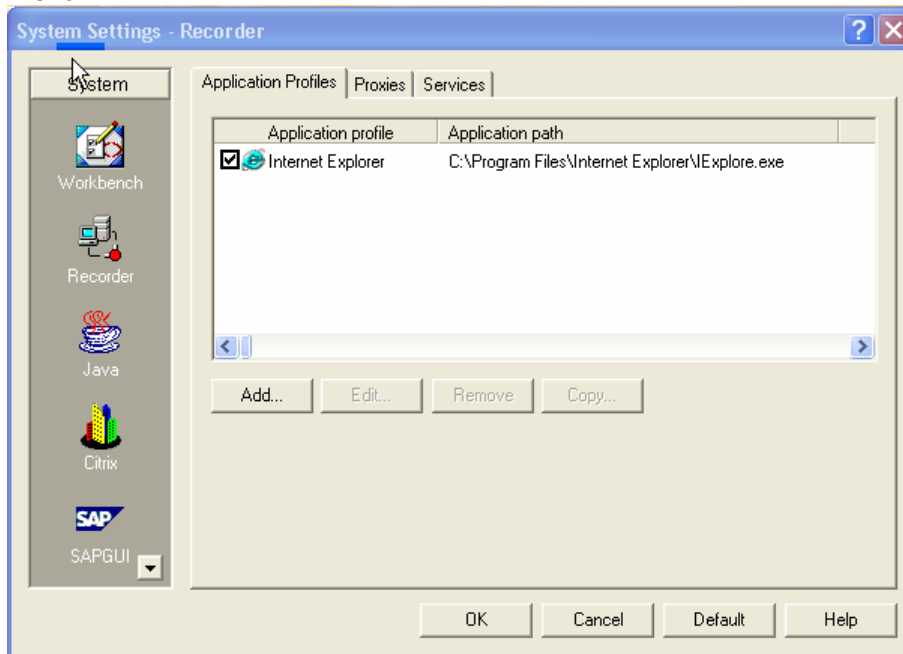
Each type is nothing but different protocols which can be recognized by the performance tool.

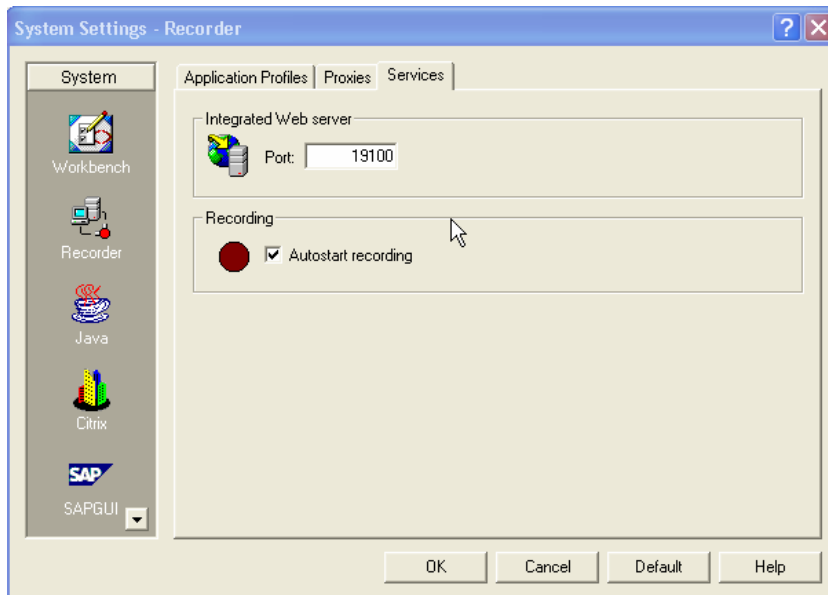
Each project should be associated to one of these protocols. Each project is associated with

1. Profiles
2. Scripts
3. Include files
4. Data files
5. Agents
6. Work loads

After the creation of projects

System setting should be set as below, This setting are under Settings-System from menu



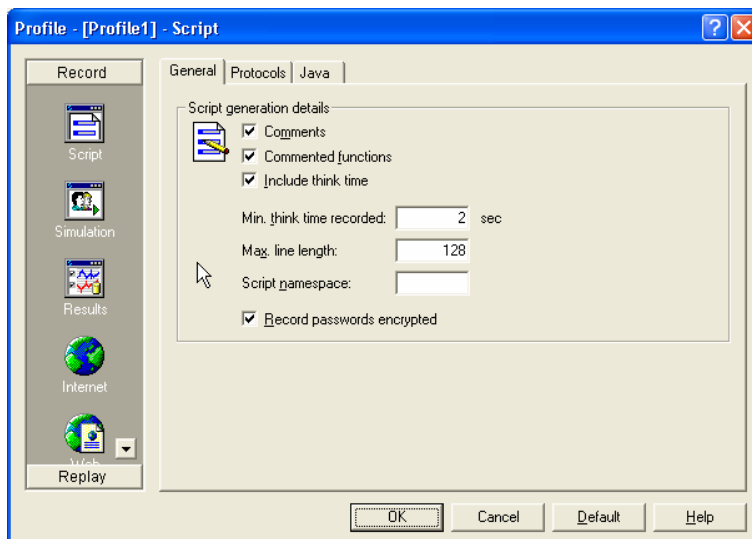


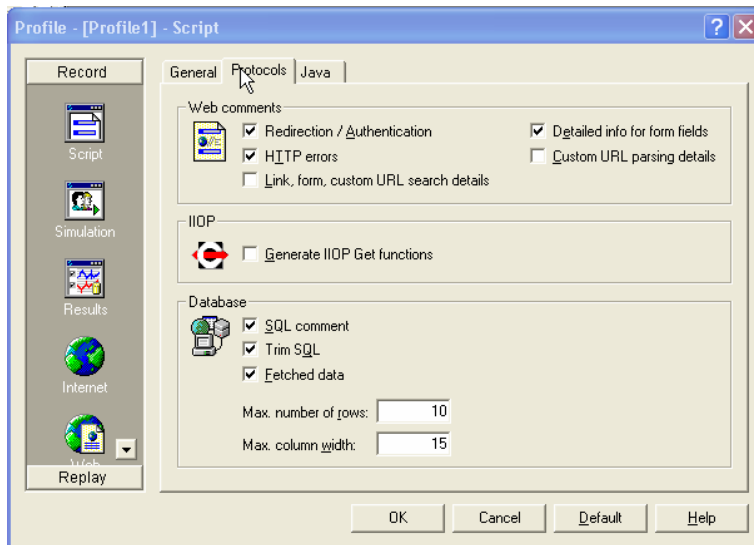
Profiles

Profiles are nothing but the record and Replay setting, This can be configured. Profile has following setting for both Record and Replay

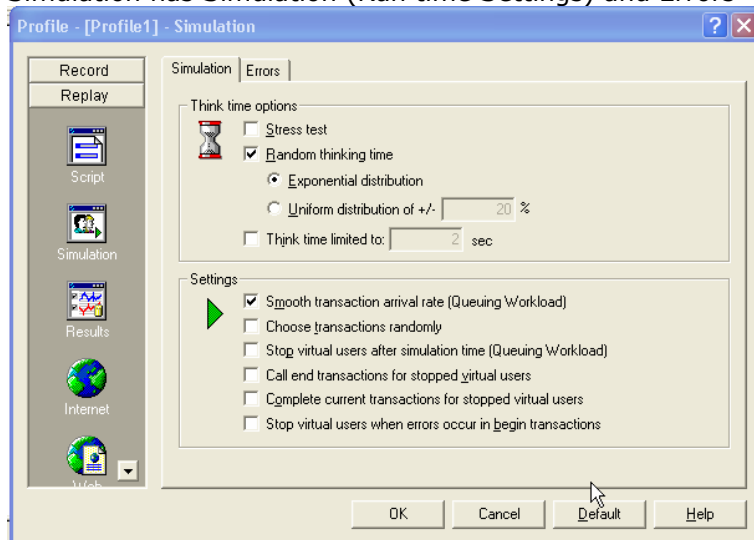
1. Scripts
2. Simulation
3. Results
4. Internet
5. Web
6. Terminal Client etc.

Scripts has General, Protocol, Java





Simulation has Simulation (Run time Settings) and Errors



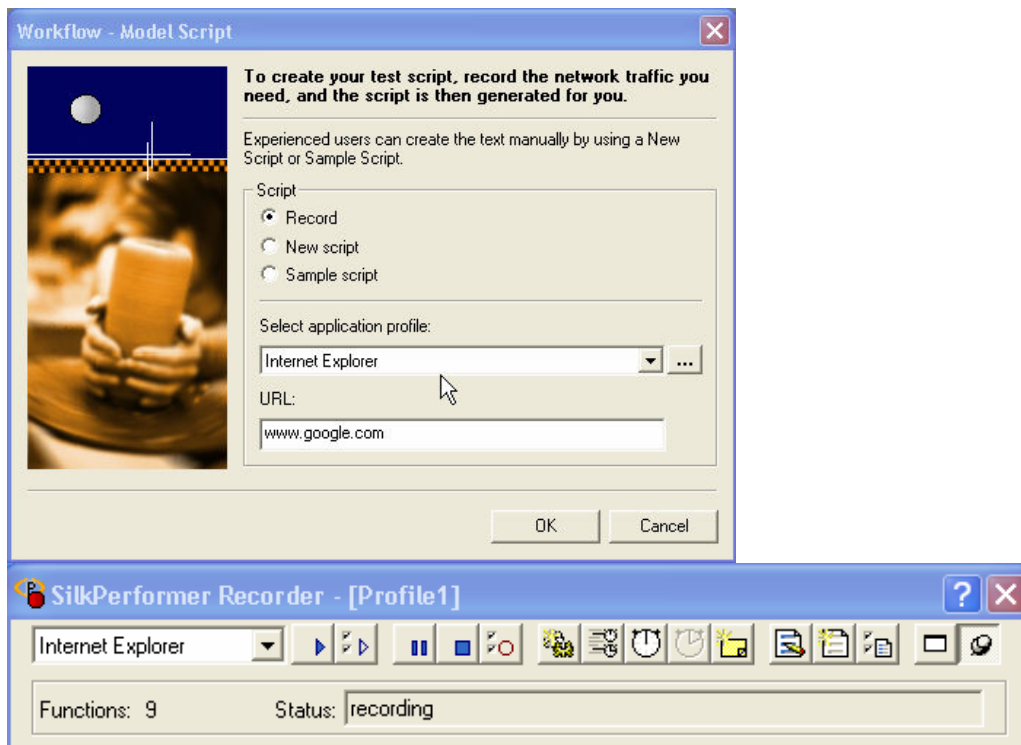
Results has

1. Time Series
2. Monitoring
3. True Log
4. Logging
5. Internet Logging
6. ARM
7. Hook Logging

The others are based on the protocol we use for the application to be load tested.

Creation of Scripts

Scripts is generally created using the record command (Model Record)



After recording the generated script look like below

```

// Recorded 10/11/2007 by SilkPerformer Recorder v7.4.0.2776

benchmark SilkPerformerRecorder
use "WebAPI.bdh"

dcluser
user
VUser
transactions
TInit          : begin;
TMain          : 1;

var

dclrand

dcltrans
transaction TInit
begin
WebSetBrowser(WEB_BROWSER_MSIE6);
WebModifyHTTPHeader('Accept-Language', "en-us");
//WebSetUserBehavior(WEB_USERBEHAVIOR_FIRST_TIME);
//WebSetDocumentCache(true, WEB_CACHE_CHECK_SESSION);
end TInit;

transaction TMain
var

```

```

transaction TMain
var
begin
  // Redirecting -> (redirection) http://www.google.co.in/
  WebCookieSet(
    "PREF=ID=a3e228063361b206:TM=1181814966:LM=1190729188:GM=1:S=7cAOi8Di-fipb3xT; domain=1 Oct 2017 06:07:30 GMT", "http://www.google.com/");
  WebPageParseUrl("src", ".src=", "\", WEB_FLAG_IGNORE_WHITE_SPACE);
  WebCookieSet(
    "PREF=ID=1a05d94a441ab9c1:TM=1181814966:LM=1181814966:S=taAgw-Kx505KiMRL; domain=.goc"
    "ct 2017 06:07:34 GMT", "http://www.google.co.in/");
  WebPageUrl("http://www.google.com/", "Google");

  ThinkTime(5.5);
  WebPageLink("src", "gen_204", 3);

  WebPageBack();

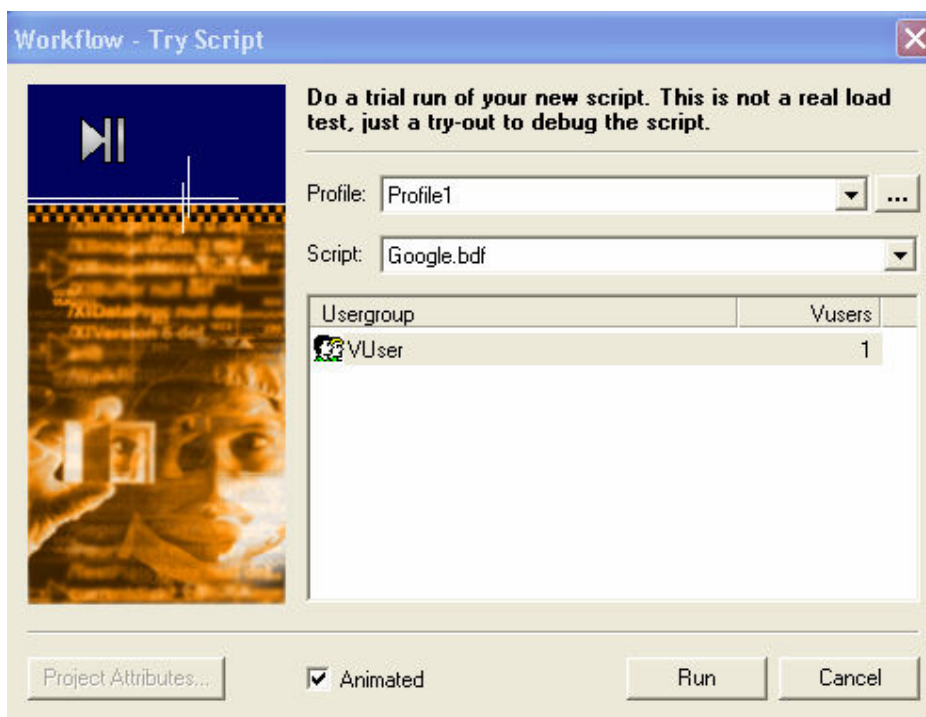
  ThinkTime(5.5);
  WebPageSubmit("f", FORM001, "books - Google Search"); // Form 1
end TMain;

iform
FORM001:
  "hl" := "" <USE_HTML_VAL> // hidden, unchanged, value: "en"
  "q" := "books", // changed
  "meta" := "" <USE_HTML_VAL> // unchanged, value: ""

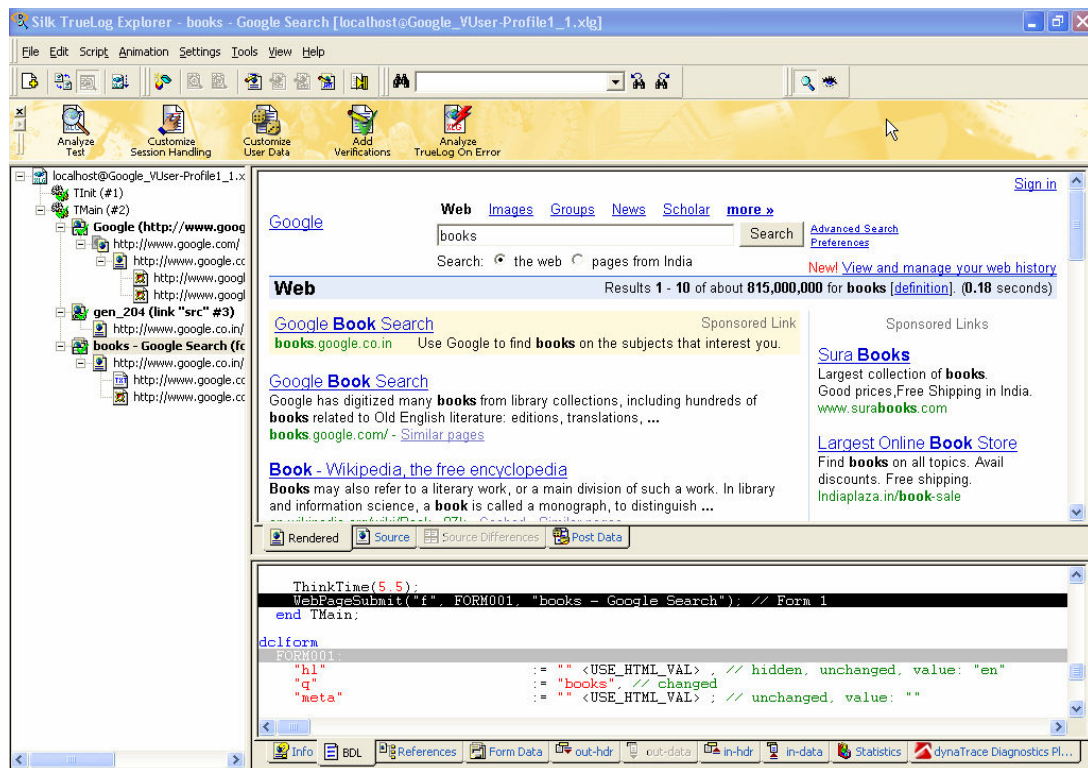
```



Model Script – Records the script to the silk performer
 Try Script – To ensure the recorded script is working properly.



In Try Script, The entire script will run as single user and it ensures the correctness of the script to parameter data, Dynamic data used by the script, proper proxy setting.



User	Agent	Line	Time	Type	Text	Info
localhost\Google_VUser-Profile1_1.x	localhost	27	00:00:00	Transaction	Exec: 0.03; Busy: 0.03; Wait: 0.00	Transaction/TInit/Trans.(busy) ok[s]
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:00	Function	WebPageUrl	"http://www.google.com"
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:01	Timer	1.59	Page Timer/Google/Page time[s]
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:01	Timer	1.25	Page Timer/Google/Document download time[s]
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:01	Timer	0.47	Page Timer/Google/Server busy time[s]
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:01	Timer	6.97	Page Timer/Google/Page data[kB]
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:01	Timer	3.39	Page Timer/Google/Embedded objects data[kB]
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:01	Timer	1.59	Page Timer/#Overall Response Time#/Page time[s]
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:01	Timer	1.25	Page Timer/#Overall Response Time#/Document download time[s]
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:01	Timer	0.47	Page Timer/#Overall Response Time#/Server busy time[s]
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:01	Timer	6.97	Page Timer/#Overall Response Time#/Page data[kB]
localhost\Google_VUser-Profile1_1.x	localhost	40	00:00:01	Timer	3.39	Page Timer/#Overall Response Time#/Embedded objects data[kB]
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:01	Function	WebPageLink	"http://www.google.co.in/gen_204?oi=promos_vis&cad=hppwebie6tb:en-G88"
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:02	Timer	0.34	Page Timer/gen_204/Page time[s]
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:02	Timer	0.34	Page Timer/gen_204/Document download time[s]
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:02	Timer	0.26	Page Timer/gen_204/Server busy time[s]
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:02	Timer	0.47	Page Timer/gen_204/Page data[kB]
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:02	Timer	0.00	Page Timer/gen_204/Embedded objects data[kB]
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:02	Timer	0.34	Page Timer/#Overall Response Time#/Page time[s]
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:02	Timer	0.34	Page Timer/#Overall Response Time#/Document download time[s]
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:02	Timer	0.26	Page Timer/#Overall Response Time#/Server busy time[s]
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:02	Timer	0.47	Page Timer/#Overall Response Time#/Page data[kB]
localhost\Google_VUser-Profile1_1.x	localhost	43	00:00:02	Timer	0.00	Page Timer/#Overall Response Time#/Embedded objects data[kB]

Silk performer shows the run script in the true log window.

Parameterization of Scripts

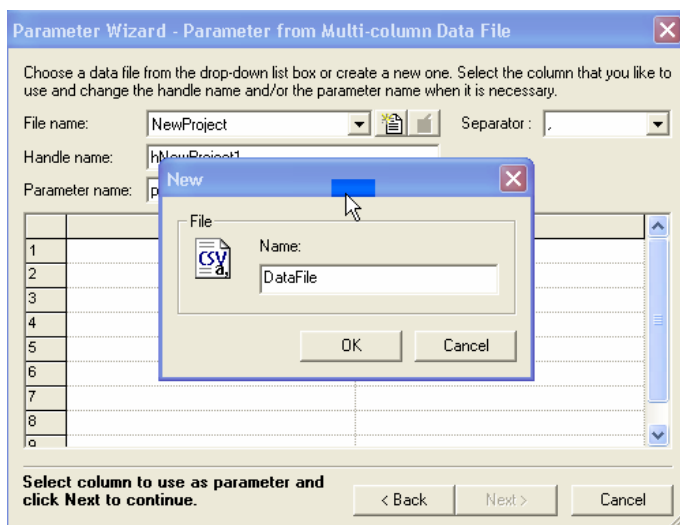
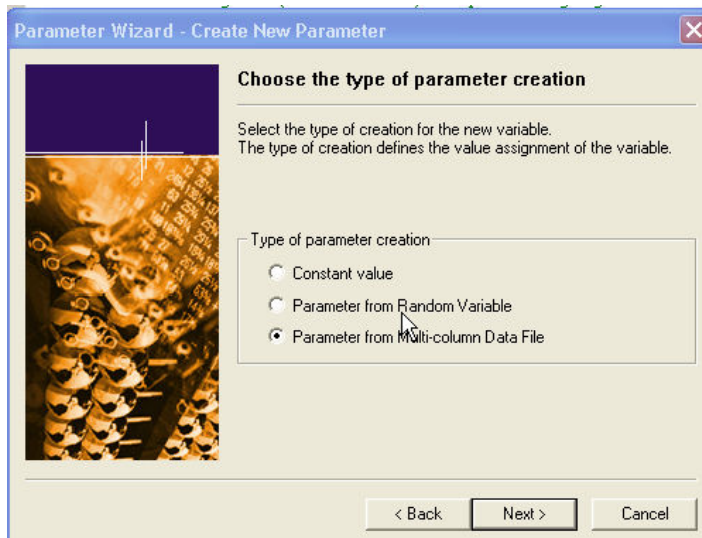
To parameterize the data, select the data that needs to be parameterized

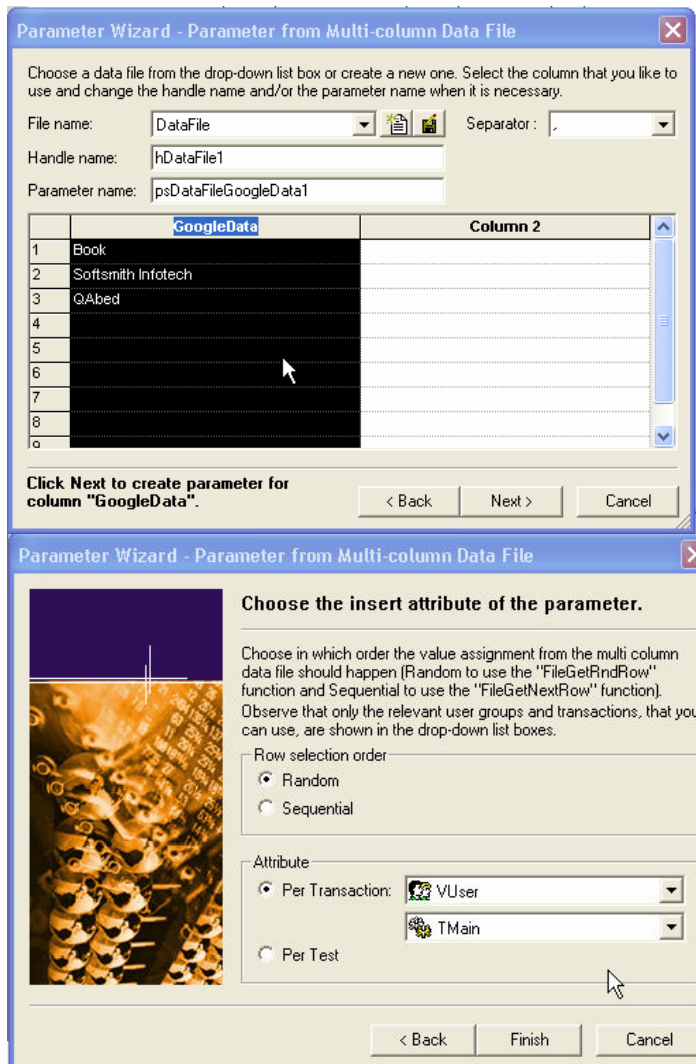
```

delform
FORM001:
  "h1"      := "<USE_HTML_VAL>" // hidden, unchanged, value: "en"
  "q"       := "books", // changed
  "meta"    := "<USE_HTML_VAL>" // unchanged, value: ""

```

Right click and select the customize value





After finishing the parameterization, the script changes with following commands

```
transaction TMain
var
begin
  FileGetRndRow(hDataFile1);
  psDataFileGoogleData1 := FileGetCol(hDataFile1, 1, STRING_COMPLETE);
  // Redirecting -> (redirection) http://www.google.co.in/
  WebCookieSet(
    "PREF=ID=a3e228063361b206:TM=1181814966:LM=1190729188:GM=1:S=7cAOi8Di-fipb3xT; domai
    "1 Oct 2017 06:07:30 GMT", "http://www.google.com/");
  WebPageParseUrl("src", "src=\"", "\"", WEB_FLAG_IGNORE_WHITE_SPACE);
  WebCookieSet(
    "PREF=ID=1a05d94a441ab9c1:TM=1181814966:LM=1181814966:S=taAgv-Kx505KiMRL; domain=.gc
    "ct 2017 06:07:34 GMT", "http://www.google.co.in/");
  WebPageUrl("http://www.google.com/", "Google");

  ThinkTime(5.5);
  WebPageLink("src", "gen_204", 3);

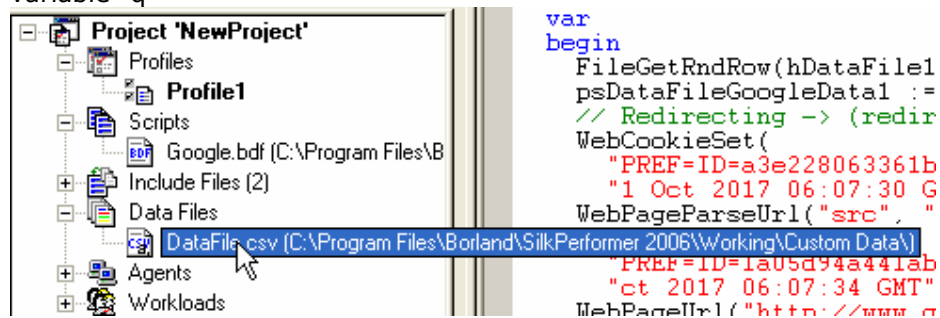
  WebPageBack();

  ThinkTime(5.5);
  WebPageSubmit("f", FORM001, "books - Google Search"); // Form 1
end TMain;

transaction TShutdown
begin
  FileUnload(hDataFile1);
end TShutdown;

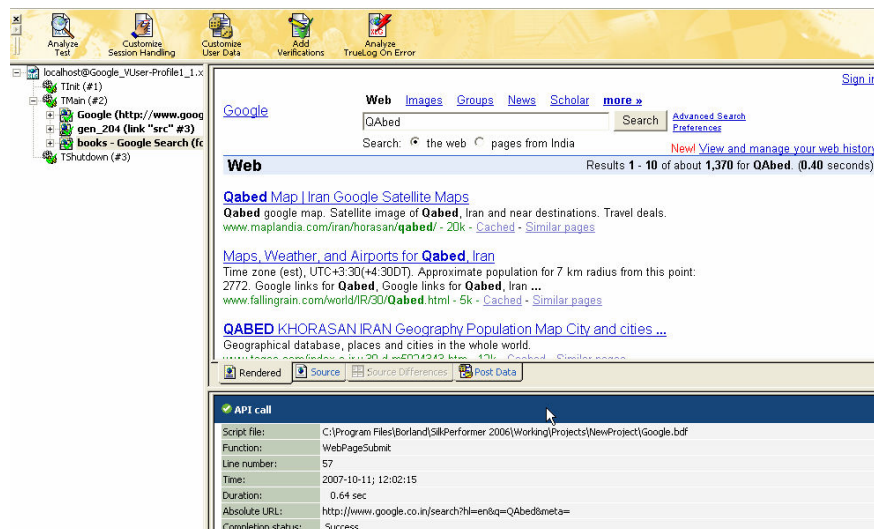
iciform
FORM001:
  "hl" := "" <USE_HTML_VAL> , // hidden, unchanged, value: "en"
  "q" := "books", // changed
  "q" := psDataFileGoogleData1,
  "meta" := "" <USE_HTML_VAL> ; // unchanged, value: ""
```

After the begin in transaction TMain, FileGetRndRow(hDataFile1); got inserted and below under FORM001 Variable "q" got commented and inserted new value for variable "q"



In the DataFiles, CSV file got inserted after the creation of parameter to the script

We can use the same file for other parameter data.



Random run took the Qabed this time of try script. This ensures the right parameterization.

Creation of Scripts with Manual Correlation

Following are the commands that needs to be used for manual correlation,

Manual correlations are used to get the list box items, Hidden Session id etc.

WebParseDataBoundEx(out sResult : string,

in nMaxResultLen : number optional,

in sLeftBoundary : string optional,

in nLeftOccurrence : number optional,

in sRightBoundary : string optional,
 in nOptions : number optional,
 in nDocNum : number optional,
 out nBytesParsed : number optional);

Return value

- none

Parameter	Description
sResult	String variable that receives the string between the specified boundary strings.
nMaxResultLen	Maximum length of the string to return (optional). If this parameter is omitted or set to STRING_COMPLETE all available data is stored in sResult.
sLeftBoundary	Left boundary of the HTML content to compare.
nLeftOccurrence	The sLeftBoundary has to be found nLeftOccurrence times, before the copy process starts and the right boundary is searched (optional). The default value is one. Provide WEB_OCCURENCE_LAST to specify the last occurrence.
sRightBoundary	Right boundary of the HTML content to compare.
nOptions	(optional) <ul style="list-style-type: none"> • WEB_FLAG_CASE_SENSITIVE. If this flag is set the string compare operation is case sensitive. • WEB_FLAG_IGNORE_WHITE_SPACE. If this flag is set all white spaces are ignored . • WEB_FLAG_DONT_FORCE_LOAD. Specify this option to enable caching for subsequent request. Note that nothing is parsed if the specified table is not loaded (cache hit). • WEB_FLAG_SYNCHRON. Parsing operations with this flag set, are done in a sequential way. The second parsing operation is not executed before the first has been completed. • WEB_FLAG_ALL_RESPONSES. If this flag is specified all server responses are scanned (even redirection responses, which are normally not displayed by a browser). • WEB_FLAG_INCLUDE_EMBEDDED. If this flag is specified, even embedded documents can be specified by the nDocNum parameter. Normally the number of a document is defined by the occurrence of the source definition in an HTML document (src=...). If this flag is specified, every embedded object increases this counter, which assigns higher numbers to subsequent frames. • WEB_FLAG_INCLUDE_HEADER. If this flag is specified the response header can also be verified. • WEB_FLAG_HEADER_ONLY. If this flag is specified only

	<p>the response header is verified.</p> <ul style="list-style-type: none"> WEB_FLAG_RULE. Specify this flag to perform the parsing operation in every subsequent web function. After every web function the specified output variables are set to the new value. If you want to use the variables in other transactions or in event handler functions use global variables! Call WebCancelAllRules() to disable all verification and parsing rules. <p>Note: The option WEB_FLAG_RULE should only be used in the INIT transaction or in combination with the WebCancelAllRules function!</p>
nDocNum	Specifies the document to parse (optional). Specify WEB_DOC_ALL if you want to parse all documents. If this parameter is omitted, the first document gets parsed. (see above definition FLAG_INCLUDE_EMBEDDED)
nBytesParsed	Variable receiving the number of bytes actually parsed (optional).

Example for Single value

rRndUniN1 – this is the random value with in a range defined as, this is used to take the occurrence between Left boundary “Value=” and Right boundary “\”

dclrand

rRndUniN1 : RndUniN (27..30);

```
WebParseDataBoundEx(sParseDataVar1, STRING_COMPLETE, ToEncoding("value="), rRndUniN1, ToEncoding("\"),
    WEB_FLAG_IGNORE_WHITE_SPACE | WEB_FLAG_CASE_SENSITIVE, 1);
WebPageUrl("http://192.168.1.107/SPCG/AddActionItem.aspx", "Action Item (#1)",
    SPCG_ADDACTIONITEM_ASPX012);
Print("sParseDataVar1: " + FromEncoding(sParseDataVar1));
Writeln("sParseDataVar1: " + FromEncoding(sParseDataVar1));
```

Syntax for Multiple values in an array

Include file

WebAPI.bdh

Syntax

WebParseDataBoundArray(inout saResult : array of string,

in nMaxCount : number,

in sLeftBoundary : string,

in sRightBoundary : string allownull,

out nFound : number optional,

in nOptions : number optional,

in nSkip : number optional,

in nDocNum : number optional,

in nMaxLen : number optional);

Parameter	Description
saResult	Array of string variable that receives all the strings between the specified boundary strings.
nMaxCount	Maximum number of strings copied into the provided array. This value must be less than or equal to the size of the array.
sLeftBoundary	Left boundary to compare.
sRightBoundary	Right boundary. After the sLeftBoundary has been found, all data is copied into the actual string parameter until the right boundary is found (optional). If this parameter is omitted the nMaxLen parameter must be specified.
nFound	Variable that receives the number of the found strings (optional.
nOptions	(optional) <ul style="list-style-type: none"> • WEB_FLAG_CASE_SENSITIVE. If this flag is set the string compare operation is case sensitive. • WEB_FLAG_IGNORE_WHITE_SPACE. If this flag is set all white spaces are ignored. • WEB_FLAG_DONT_FORCE_LOAD. Specify this option to enable caching for subsequent request. Note that nothing is parsed if the specified table is not loaded (cache hit). • WEB_FLAG_SYNCHRON. Parsing operations with this flag set, are done in a sequential way. The second parsing operation is not executed before the first has been completed. • WEB_FLAG_ALL_RESPONSES. If this flag is specified all server responses are scanned (even redirection responses, which are normally not displayed by a browser). • WEB_FLAG_INCLUDE_EMBEDDED. If this flag is specified, even embedded documents can be specified by the nDocNum parameter. Normally the number of a document is defined by the occurrence of the source definition in an HTML document (src=...). If this flag is specified, every embedded object increases this counter, which assigns higher numbers to subsequent frames. • WEB_FLAG_INCLUDE_HEADER. If this flag is specified the response header can also be verified. • WEB_FLAG_HEADER_ONLY. If this flag is specified only the response header is verified. • WEB_FLAG_RULE. Specify this flag to perform the parsing operation in every subsequent web function. After every web function the specified output variables are set to the new value. If you want to use the variables in other transactions or in event

	<p>handler functions use global variables! Call WebCancelAllRules() to disable all verification and parsing rules.</p> <p>Note: The option WEB_FLAG_RULE should only be used in the INIT transaction or in combination with the WebCancelAllRules function!</p>
nSkip	Specifies the number of parse results, which should not be stored in the array (optional). The first element of the array will be the (nSkip+1) th parse result.
nDocNum	Specifies the document to parse (optional). Specify WEB_DOC_ALL if you want to parse all documents. If this parameter is omitted, the first document gets parsed. (see above definition FLAG_INCLUDE_EMBEDDED)
nMaxLen	Specifies the maximum number of bytes copied to each string (optional). If the sRightBoundary parameter is omitted this parameter must be specified and determines the end of every parsing operation.

Measuring page times

To measure the page times in the log, we can also use following to measure the time

```
Measurestart ("StringName")
MeasureStop("StringName")
MeasureGet
```

```
MeasureStart
Include file
```

Kernel.bdh

Syntax

```
MeasureStart( in sMeasure : string ): boolean;
```

Return value

- **true** if successful
- **false** otherwise

Parameter	Description
sMeasure	Measure name used to identify the measure when calling additional measure functions and when analyzing the results written to both the results repository and individual result files

Example

```
MeasureStart ("<StringName>");
```

MeasureStop

Include file

Kernel.bdh

Syntax

```
MeasureStop( in sMeasure      : string  
  
             in bIgnoreOnError : boolean optional ): number;
```

Return value

Final value of the custom time measure in 1/1000 secs.

Parameter	Description
sMeasure	Measure name identifying the custom time measure to stop
bIgnoreOnError	When enabled, measured time is included in calculations if no error has occurred since the timer started (optional). The default value is false.

Example

```
MeasureStop("<StringName>")
```

MeasureGet

Include file

Kernel.bdh

Syntax

```
MeasureGet( in sName : string,  
  
            in nClass : number,  
  
            in nKind : number,  
  
            out fTime : float,  
  
            in bAll  : boolean optional): boolean;
```

Return value

- **true** if successful
- **false** otherwise

Parameter	Description
sName	Name of the object concerned with the measurement. Must be one of the following:

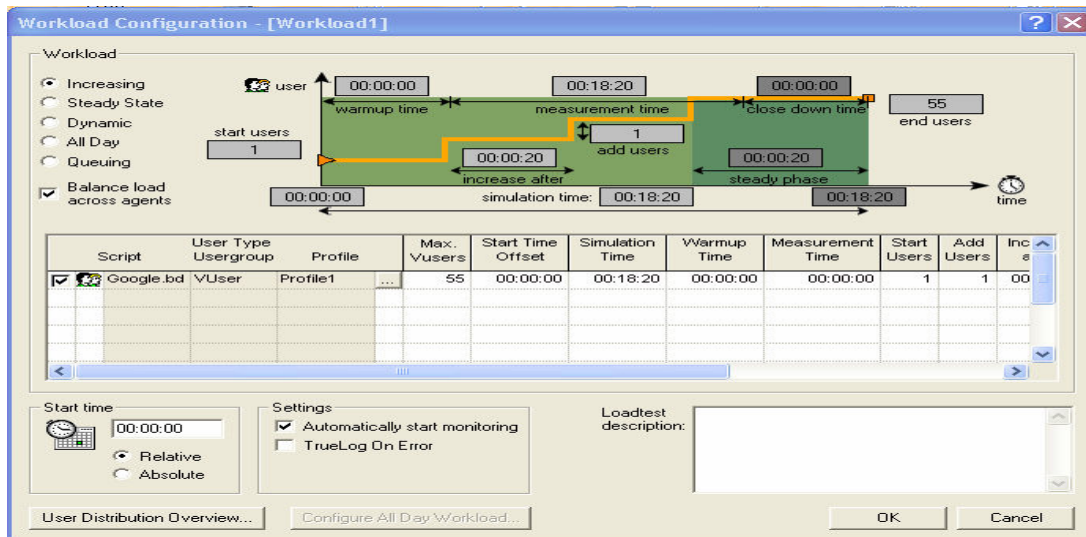
	<ul style="list-style-type: none"> • Custom time measure. Pass to the function exactly the same name as to the MeasureStart and the MeasureStop function • Custom counter. Pass to the function exactly the same name as to the MeasureInc function • Transaction. Specify the transaction name as declared in the load testing script. • SQL command. Make sure to specify the SQL command identifier (defined in the dclsql section of the test script) in capital letters. • Web form. Make sure the specify the Web form identifier (defined in the dclform section of the test script) in capital letters. • CORBA object • TUXEDO service
nClass	<p>Specifies the type of measure to retrieve.</p> <p>To retrieve the value of a custom time measure, pass the MEASURE_TIMER_RESPONSETIME parameter to the function.</p> <p>To retrieve the value of a custom counter, pass the MEASURE_COUNTER_CUSTOMCOUNTER parameter to the function.</p> <p>To retrieve the value of an average counter, pass the MEASURE_COUNTER_AVERAGE parameter to the function.</p> <p>In any other case, pass any of the following parameters to the function, depending on the type of information you are interested in.</p> <ul style="list-style-type: none"> • MEASURE_PAGE_PAGETIME • MEASURE_PAGE_PAGEBYTES • MEASURE_PAGE_EMBEDDEDBYTES • MEASURE_PAGE_DOCDOWNLOAD • MEASURE_PAGE_SERVERBUSY • MEASURE_IOP_ROUNDTRIP • MEASURE_IOP_SERVERBUSY • MEASURE_SQL_SQLPARSE • MEASURE_SQL_SQLEXEC • MEASURE_SQL_SQLEXECDIRECT • MEASURE_TRANS_TRANSOK • MEASURE_TRANS_TRANSERR • MEASURE_TRANS_TRANSCA • MEASURE_FORM_BYTESSENT • MEASURE_FORM_BYTESRECEIVED • MEASURE_FORM_HITSOK • MEASURE_FORM_HITSERR • MEASURE_FORM_ROUNDTRIP • MEASURE_FORM_SERVERBUSY • MEASURE_TUXEDO_BYTESSENT • MEASURE_TUXEDO_BYTESRECEIVED • MEASURE_TUXEDO_RESPONSETIME
nKind	<p>Specifies the type of measure value to retrieve. The following options are</p>

	<p>possible:</p> <ul style="list-style-type: none"> • MEASURE_KIND_SUM. Retrieves the sum of all measure values, for example, the total time for executing all transactions • MEASURE_KIND_COUNT. Retrieves the number how often an action was performed, for example, how often a counter was incremented, or how often a transaction was executed • MEASURE_KIND_AVERAGE. Returns the average of all measure values, for example, the average time required for a CORBA operation call. • MEASURE_KIND_MIN. Returns the minimum of all measure values • MEASURE_KIND_MAX. Returns the maximum of all measure values • MEASURE_KIND_LAST. Returns the last value set for the measure • MEASURE_KIND_SQSUM. Calculates for each measure value the square value, and sums up all square values. • MEASURE_KIND_STDEVIATION. Returns the standard deviation of all measured values
fTime	Variable receiving the measure value
bAll	<p>Specifies the time interval used for measurement value calculation (optional).</p> <ul style="list-style-type: none"> • If this parameter is set to true, the measure value is calculated based on all measurements performed during the simulation. • Otherwise, if this parameter is set to false, the measure value is calculated based only on the measurements performed during the measurement interval (default).

Example

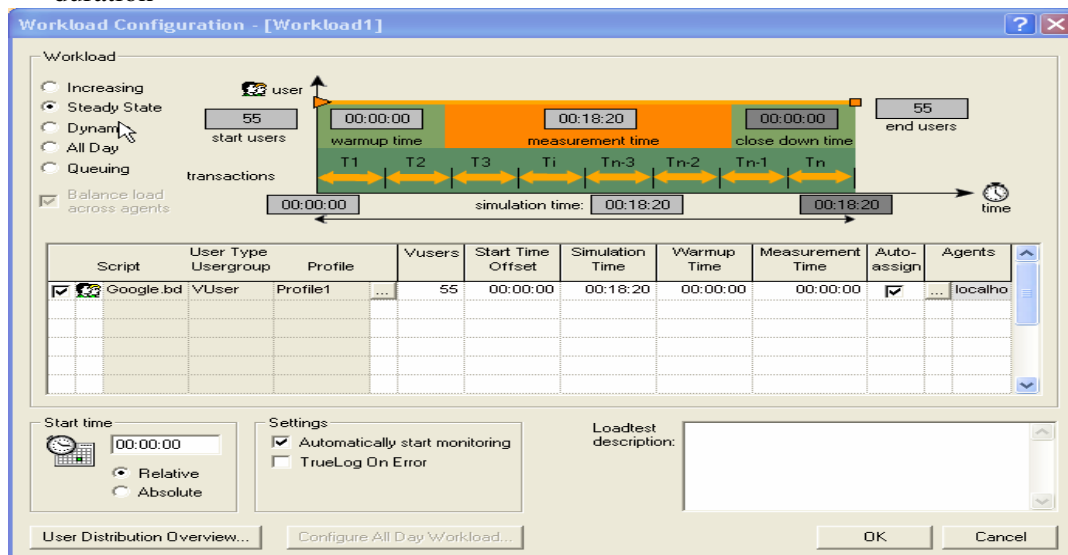
```
MeasureGet("<StringName>", MEASURE_TIMER_RESPONSETIME,
MEASURE_KIND_SUM, fValue);
```

WorkLoads

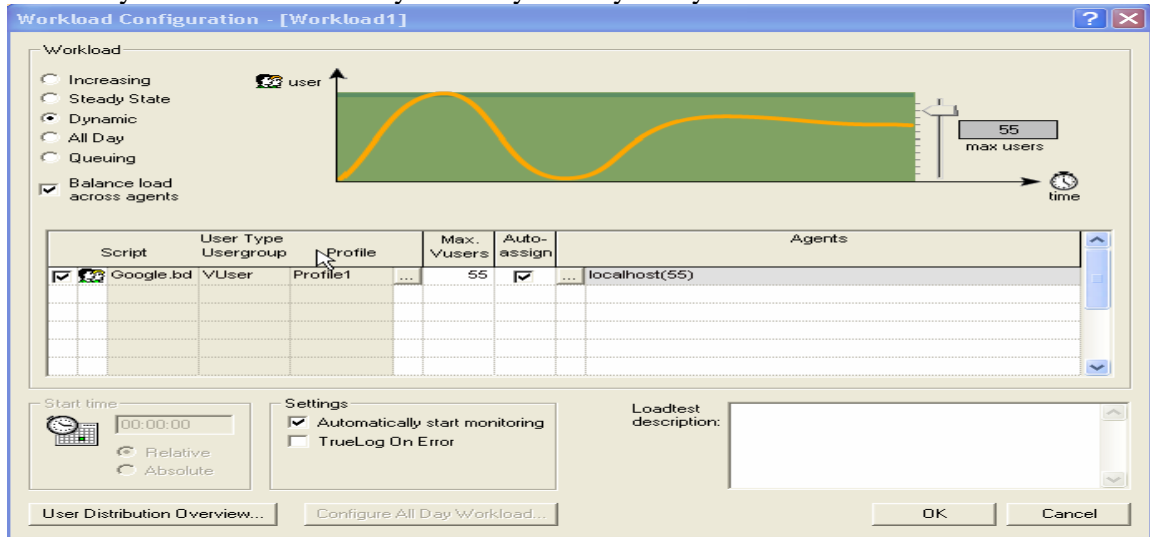


Workload means scheduling the scripts for multiple users. We can schedule the load in following pattern

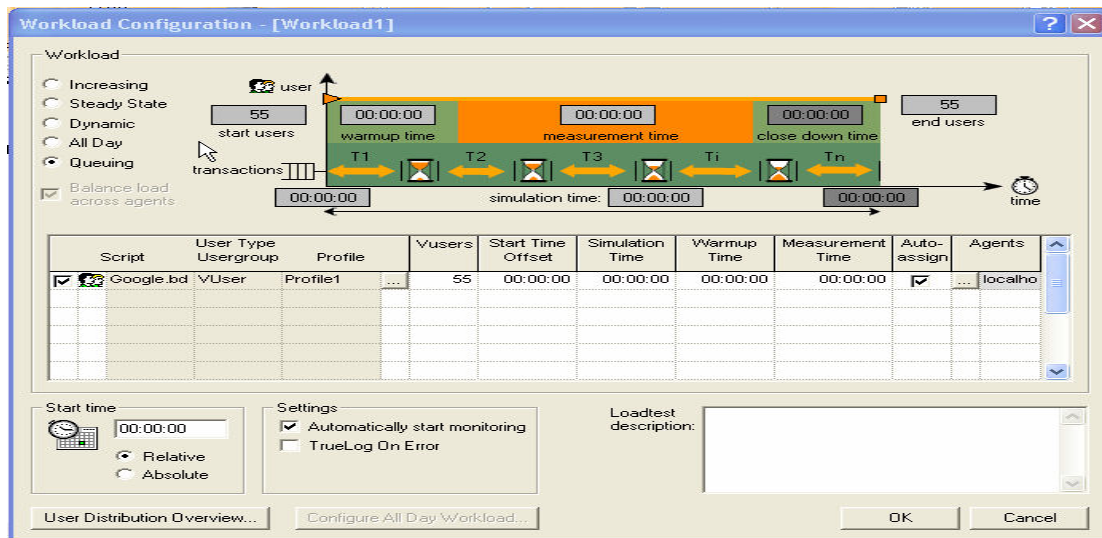
1. Increasing – This is called ramp up where the load to the server is increased by definite time and can run for certain duration called Simulation time.
2. Steady State – Here all the virtual users are loaded simultaneously for definite duration



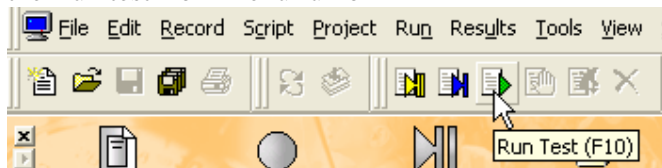
3. Dynamic – The load is dynamically used by the system for definite duration

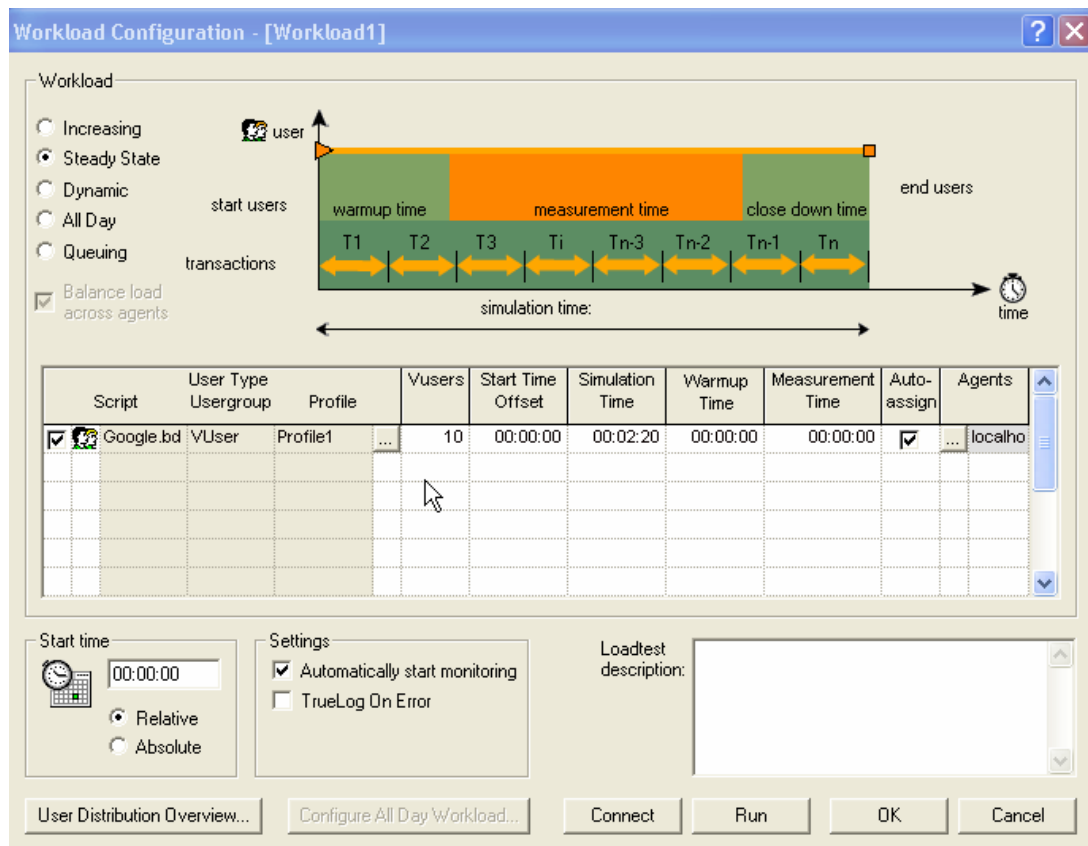


4. Queuing – There will be definite wait time between transactions.



Once we set the work load for which the load testing needs to be done, Run the test using the Run test from menu run or





Starting the run test will give the configuration screen for load, we have changed the vusers to 10 and duration to 2:20 min



5 sec

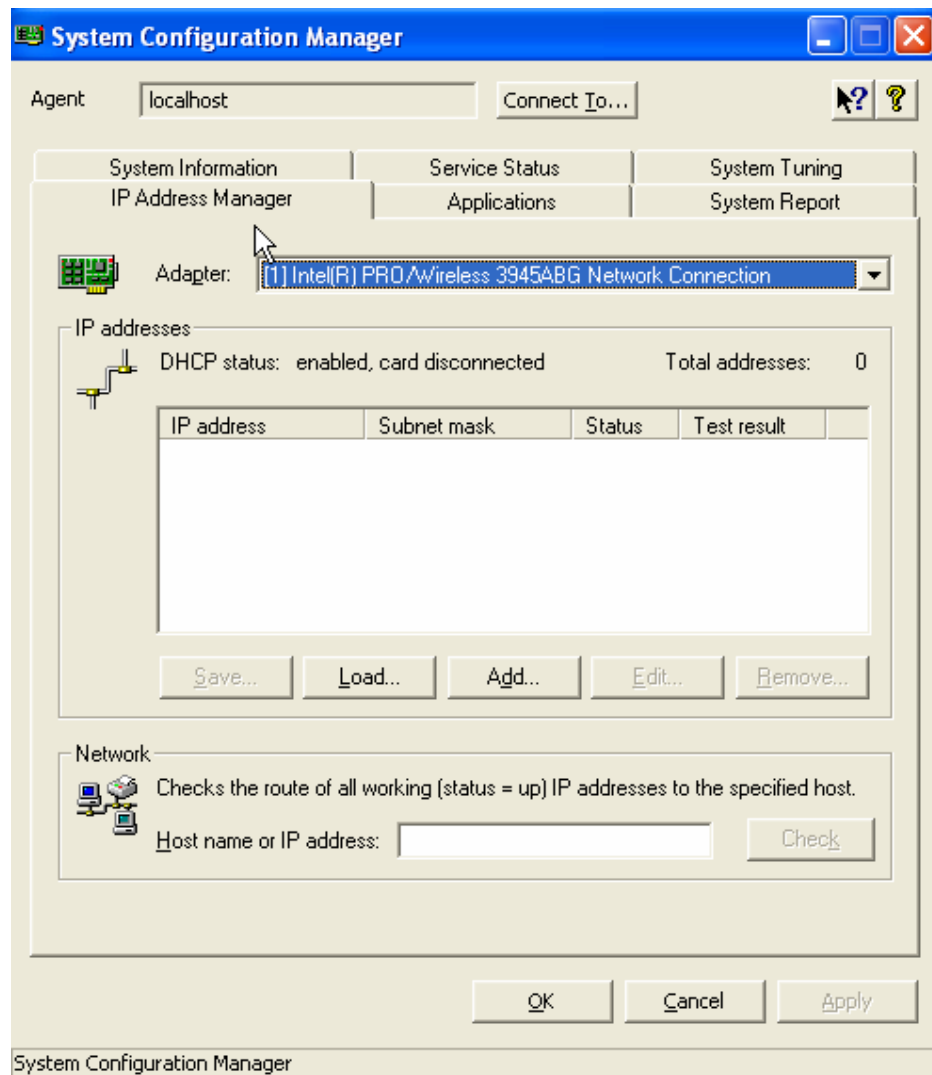
Summary	Status	Users	created	exec.	failed	Cpu	Memory	Resp...	Transactions	Tra. Bu...	Page
All Users	executing	10	10	10	0	11%	121%	100%	114	2.17	
localhost	executing	10	10	10	0	11%	121%	100%	114	2.17	
Google.bdf/VUser-Profile1	executing	10	10	10	0	---	---	---	114	2.17	

User	Agent	Status	Current Transaction	Last Resp.	Avg. Resp.	Transactions	Tra. Bu...	Page
VUser-Pro...	localhost	executing	TMain (9)	17.25	15.02	9	1.71	
VUser-Pro...	localhost	thinktime	TMain (9)	14.65	14.78	9	7.98	
VUser-Pro...	localhost	thinktime	TMain (15)	10.00	8.46	15	1.71	
VUser-Pro...	localhost	thinktime	TMain (12)	4.23	11.05	12	2.28	
VUser-Pro...	localhost	thinktime	TMain (11)	16.39	10.86	11	2.14	
VUser-Pro...	localhost	thinktime	TMain (8)	12.65	16.53	8	2.28	
VUser-Pro...	localhost	thinktime	TMain (10)	8.32	10.68	10	2.25	
VUser-Pro...	localhost	thinktime	TMain (14)	8.20	8.93	14	2.67	
VUser-Pro...	localhost	executing	TMain (10)	23.85	14.10	10	3.29	
VUser-Pro...	localhost	thinktime	TMain (13)	9.00	10.22	13	1.71	

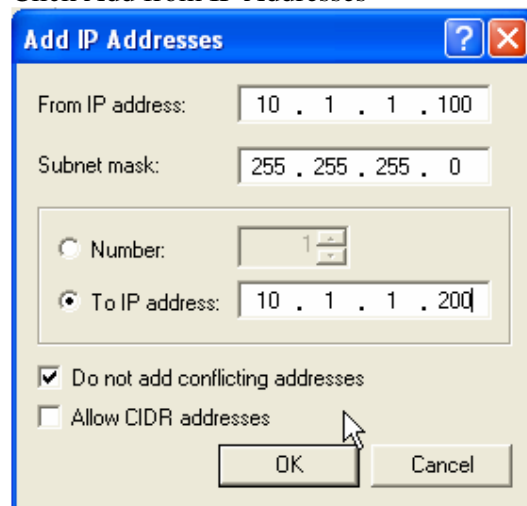
Distributing users to Multiple IP Address

In the above situation, the virtual users are distributed through only one ip address, if we like to distribute the users with multiple IP address then

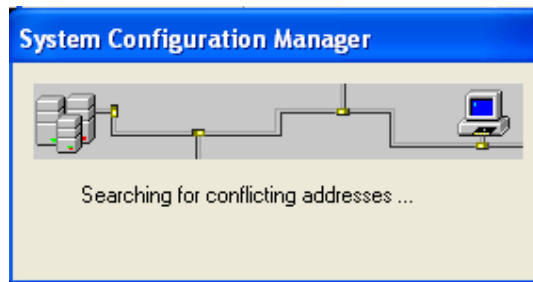
1. Go to SilkPerformer->Tools->SystemConfigurationManager
2. Select IP address Manager



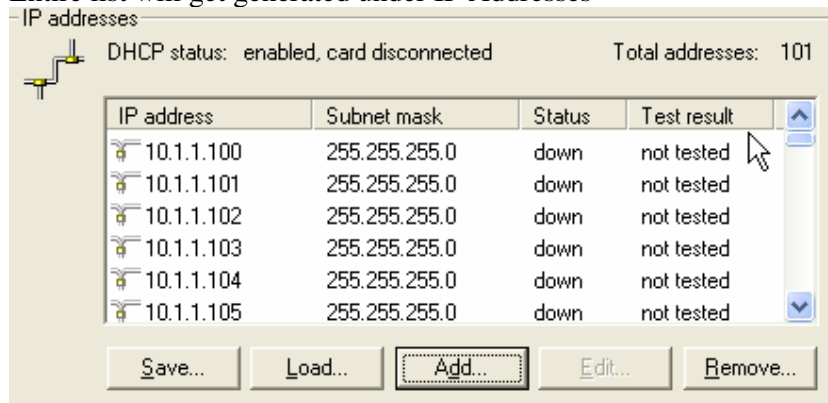
3. System Configuration Manager
4. Click Add from IP Addresses



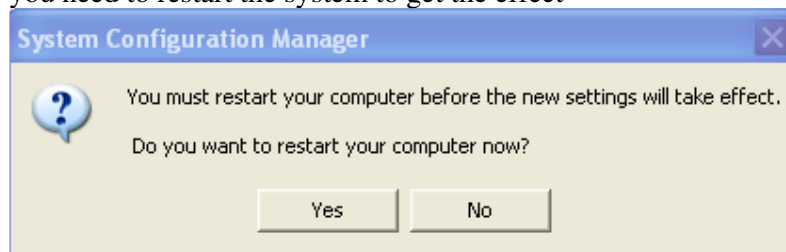
- 5.



- 6.
7. Entire list will get generated under IP Addresses



- 8.
9. you need to restart the system to get the effect



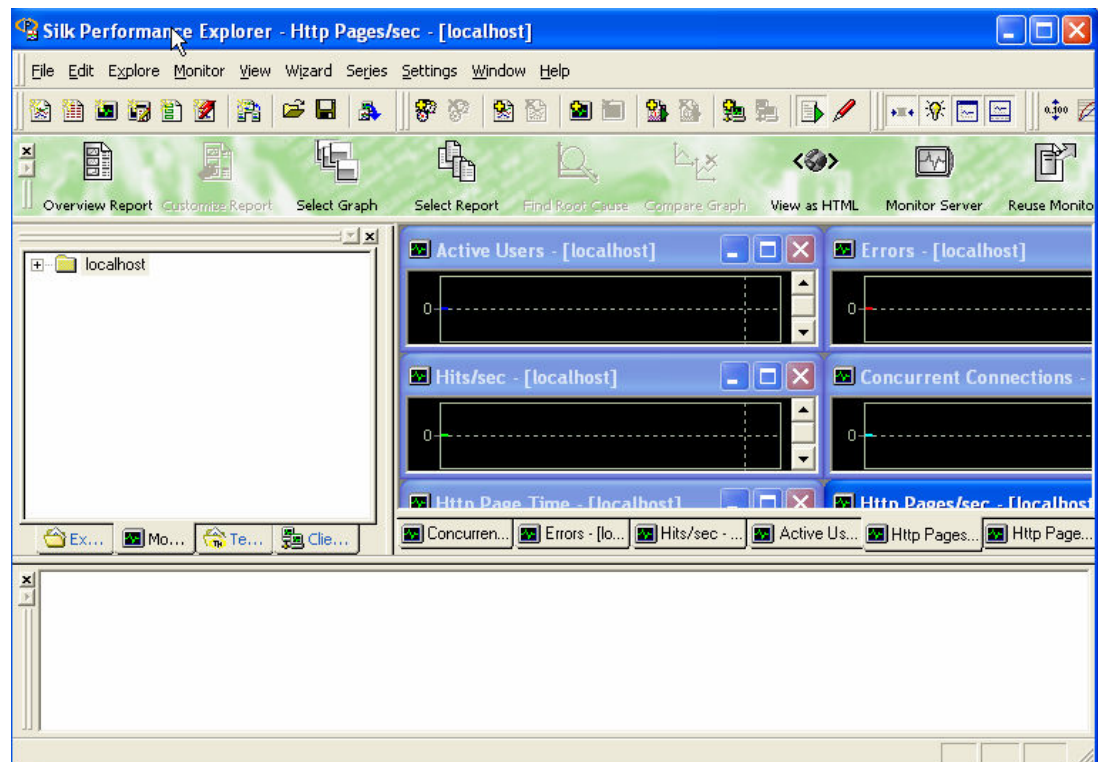
- 10.
11. The same also can be saved as .smf file which can be loaded for different workspace.

Configuring the Servers and its monitors

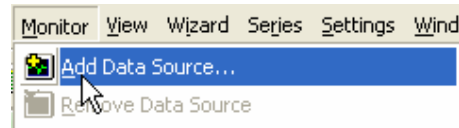
For performance testing, we have to configure the servers to get the information on those servers, The information can be of %cpu usage, Memory available, Concurrent connection etc.

To do the same.

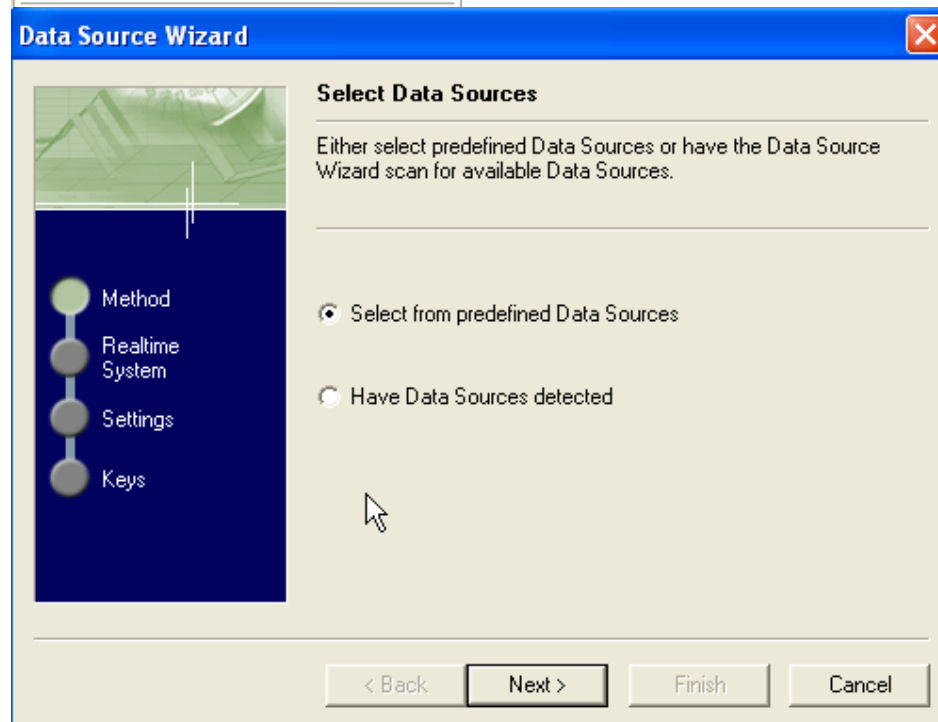
1. Select the From SilkPerformer->Results->MonitorServer



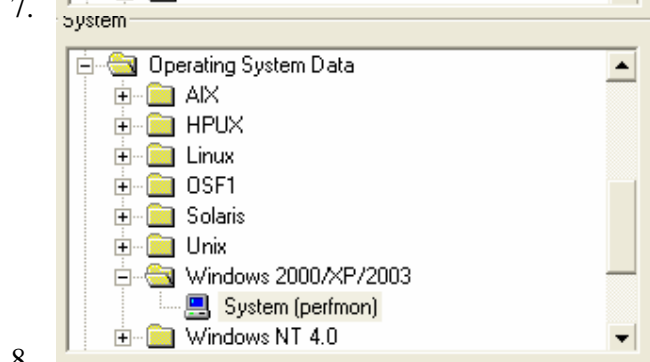
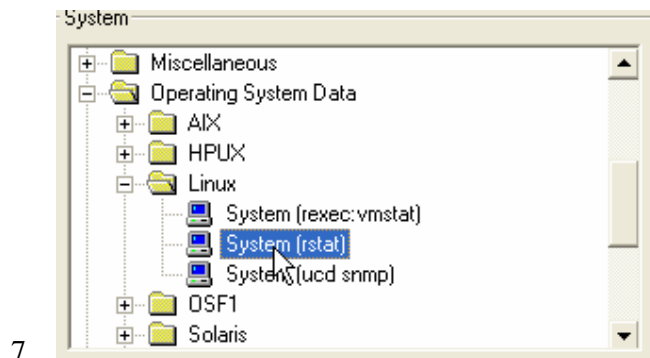
- 2.
3. Select From Monitors->Add data source



- 4.



- 5.
6. Select from the system the server you like to monitor



9. Give the connection Parameter

Connection parameters

Specify the hostname and other parameters needed to connect to the required data source.

Connection parameters

Hostname: 192.168.1.107

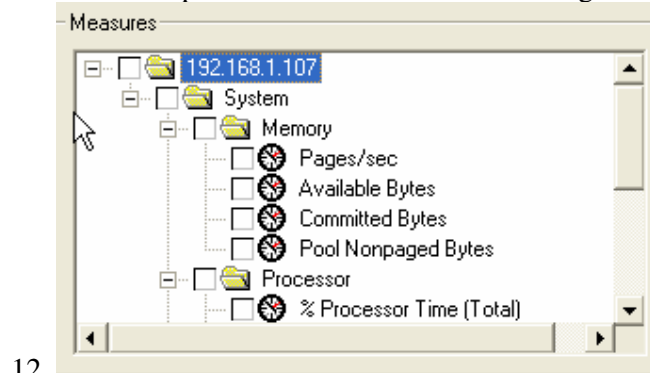
Alias: 192.168.1.107

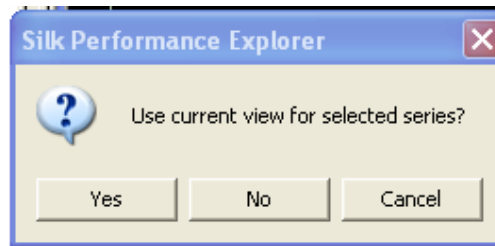
Username: softsmith

Password: xxxxxxxx

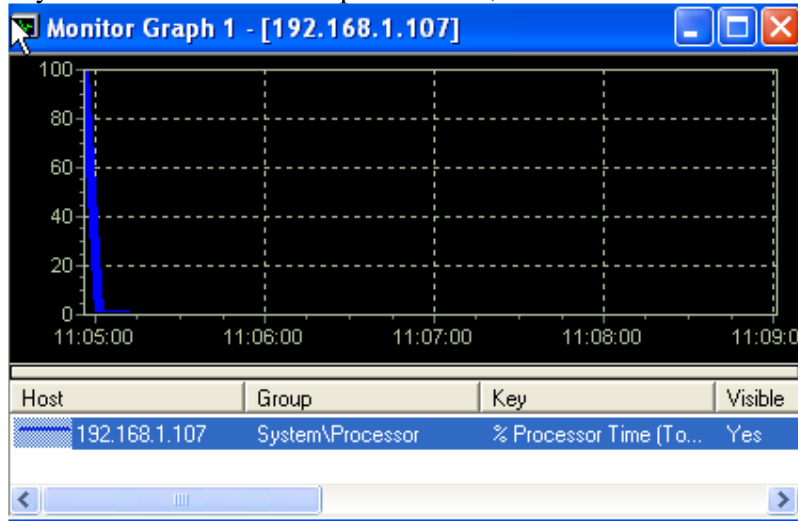
Domain:

10.
11. Select the parameter which we like to configure





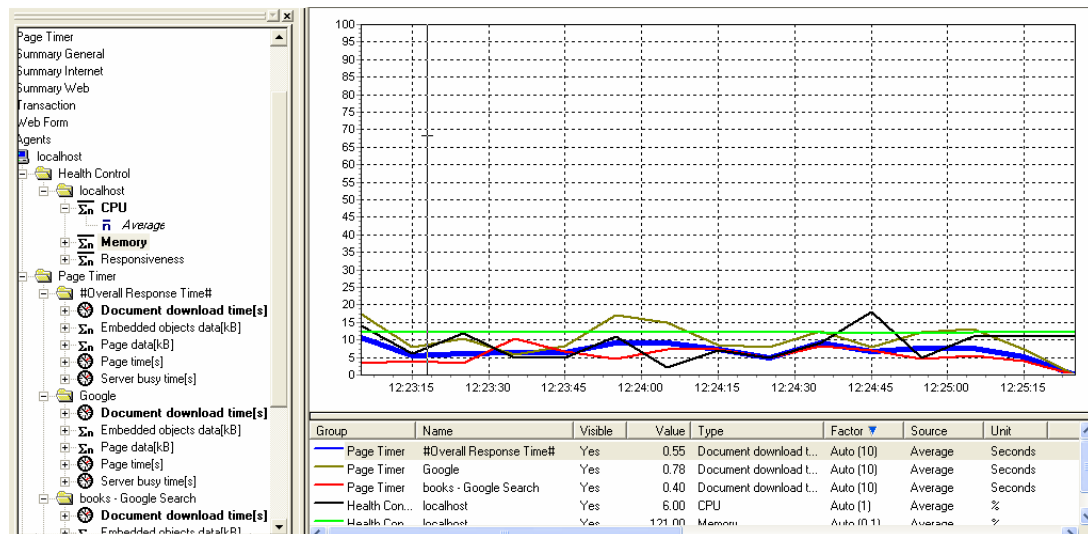
- 13.
14. If you want to make it as separate series, select No



- 15.
16. We can add any no. of parameter to this.

Analysis

Analysis can be done based on the Responses for each page and understand the behavior of each page.



By selecting the appropriate measuring units, we can drill down the bottle neck in the pages.

For errors, there is "Analyze Error" tool which can cumulate all the errors (http errors) and from that we need to provide the solutions.

Actual Base Line Report

From Menu – Results – Actual Base Line report can be generated. This report is mostly used for every final run.

BorlandReport generated at: 2007-10-11 12:45:29

Actual Baseline Report

Version: SilkPerformer 7.4.0.2776

Project: NewProject

Start date/time: 2007-10-11 12:22:55

Workload: Workload1

Workload model: Steady State

Status: No errors occurred





Baseline test result overviewback to top

Select a user type, or click on one of the counters, to jump directly to the according section.

User Type	#Users	Test duration [s]	Session Time [s]	Session Busy Time [s]	Average Page Time [s]	#Transaction OK	#cancelled	#failed	#Errors
Google.bdf/VUser/Profile1	10	00:02:21	11.808	2.670	0.892	122	0	0	--

Accept Baseline

Results for [Google.bdf/VUser/Profile1](#)back to overview

Name	Avg	Min	Max	Count	StdDev	Histogram
#Overall Response Time#						
response time breakdown (server / document / page)						
Page time[s]	0.892	0.281	7.359	349	0.972	
Document download time[s]	0.725	0.281	6.484	349	0.729	
Server busy time[s]	0.384	0.000	1.734	349	0.246	
Page data[kB]	4.338	0.475	6.975	349	2.784	
Embedded objects data[kB]	1.184	0.000	3.386	349	1.614	
books - Google Search						
response time breakdown (server / document / page)						
Page time[s]	0.594	0.281	3.391	112	0.551	
Document download time[s]	0.594	0.281	3.391	112	0.551	
Server busy time[s]	0.258	0.000	1.281	112	0.231	
Page data[kB]	5.437	5.210	5.689	112	0.233	
Embedded objects data[kB]	0.000	0.000	0.000	112	0.000	
gen_204						
response time breakdown (server / document / page)						
Page time[s]	0.452	0.312	1.812	115	0.258	
Document download time[s]	0.452	0.312	1.812	115	0.258	
Server busy time[s]	0.349	0.000	1.734	115	0.254	
Page data[kB]	0.475	0.475	0.475	115	0.000	
Embedded objects data[kB]	0.000	0.000	0.000	115	0.000	
Google						
response time breakdown (server / document / page)						
Page time[s]	1.581	0.781	7.359	122	1.275	
Document download time[s]	1.102	0.562	6.484	122	0.976	