

<Organisation Name>

Performance Test Solution

<Solution Name>

[Test Case Summary](#V_TestCaseSummary)

|  |
| --- |
| Document Date: |
|  | 17 January 2020 |
| Document Status: |
|  | WIP |
| File Name: |
|  |  |

Contents

[Contents 2](#_Toc30146641)

[List of Figures 2](#_Toc30146642)

[List of Tables 2](#_Toc30146643)

[Version History 3](#_Toc30146644)

[References and related documents 3](#_Toc30146645)

[Key Contacts 3](#_Toc30146646)

[1 Introduction 4](#_Toc30146647)

[1.1 Document Purpose 4](#_Toc30146648)

[1.2 Application Background 4](#_Toc30146649)

[1.3 Target Audience 4](#_Toc30146650)

[1.4 Releases 4](#_Toc30146651)

[2 Test Case Summary 5](#_Toc30146652)

[3 Test Cases 6](#_Toc30146653)

[4 Test Data 7](#_Toc30146654)

[4.1 Virtual Table Server (VTS) 7](#_Toc30146655)

[4.2 Text Data Files 7](#_Toc30146656)

[4.3 LoadTest Database 8](#_Toc30146657)

[4.4 Test Accounts 10](#_Toc30146658)

[5 Operations 11](#_Toc30146659)

[5.1 Deployments 11](#_Toc30146660)

[6 Test Infrastructure 12](#_Toc30146661)

[6.1 Test Environment 12](#_Toc30146662)

[6.2 Microfocus LoadRunner Environment 13](#_Toc30146663)

[7 Trouble Shooting 14](#_Toc30146664)

[8 Scripting Notes 15](#_Toc30146665)

[9 Common Rules 18](#_Toc30146666)

[Parameter rules 18](#_Toc30146667)

[Parameter rules (continued) 19](#_Toc30146668)

[Parameter rules (continued) 20](#_Toc30146669)

[Other rules 21](#_Toc30146670)

List of Figures

[Figure 1 LoadTest Database Script / Status / Table Cross-Reference 9](#_Toc30146635)

[Figure 2 Schematic diagram of the infrastructure under test 12](#_Toc30146636)

[Figure 3 Schematic diagram of the *Microfocus LoadRunner* environment 13](#_Toc30146637)

[Figure 4 Recording Options to pick up JSON requests 15](#_Toc30146638)

[Figure 5 Auto Log Cache Setting 16](#_Toc30146639)

[Figure 6 OAM Command Line setting in Performance Center 16](#_Toc30146640)

List of Tables

[Table 1 Version History 3](#_Toc30146616)

[Table 2 References and Related Documents 3](#_Toc30146617)

[Table 3 Key Contacts 3](#_Toc30146618)

[Table 4 Test Case Summary - IR10 RSVT 5](#_Toc30146619)

[Table 3 Think Time Guidelines 15](#_Toc30146620)

[Table 6 Common Text Data File parameter rules 18](#_Toc30146621)

[Table 7 Common LoadTest Database parameter rules 18](#_Toc30146622)

[Table 8 Common VTS parameter rules 18](#_Toc30146623)

[Table 9 Common Boundary Correlation rules 19](#_Toc30146624)

[Table 10 Common Regex Correlation rules 19](#_Toc30146625)

[Table 11 Common Custom parameter rules 20](#_Toc30146626)

[Table 12 Common Date/Time parameter rules 20](#_Toc30146627)

[Table 13 Common Random Number parameter rules 20](#_Toc30146628)

[Table 14 Common Additional Attributes 20](#_Toc30146629)

[Table 15: DoxRunner Operation Configuration 21](#_Toc30146630)

[Table 16 Common Header rules 21](#_Toc30146631)

[Table 17 Common SAP GUI rules 21](#_Toc30146632)

[Table 18: Response Time NFR References 21](#_Toc30146633)

[Table 19: Glossary of Terms 23](#_Toc30146634)

Version History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Author | Date | Comment |
| V1.0 | <you> | 17/01/2020 | Continuing updates |

Table 1 Version History

References and related documents

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Title | Version | Date | Link/location |
| [Ref1] |  | v1.2 |  | Table 14 |
| [Ref2] |  | V1.1 | 6/08/2013 | [**SharePoint**](#Ref2) |
| [Ref3] |  |  |  | [**Site**](#Ref3) |
| [Ref4] |  | N/A | 12/2/2015 | [**File**](#Ref4) |

Table 2 References and Related Documents

Key Contacts

The contacts listed in Table 2 may be out of date.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Role | Phone | Mobile | Email |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Table 3 Key Contacts

1. Introduction
	1. Document Purpose

This document represents notes made by the <*Organisation*> performance test team during the scripting, data creation, and testing of the *<Solution Name>*. It documents what actually was implemented regardless of the various *Performance Test Plans* and/or *Designs*.

Its purpose is to:

* Provide a central repository for all knowledge that will make the future testing of the *<Solution Name>* easier;
* Provide a central configuration for automated scripting processes;
* Provide hints and reasons behind decisions made that may affect future projects.

That is, it is meant to span multiple projects and releases of *<Solution Name>*.

Disclaimer:

Official Performance Test deliverables always have priority, so this document may not necessarily be up to date.

* 1. Application Background

<describe as appropriate>.

* 1. Target Audience

The target audience for this document is:

* Performance Test Manager;
* Performance Test Lead;
* Performance Test Analyst(s);
* <add as appropriate>
	1. Releases

This section provides a running history of the various projects and releases so that new members of the performance test team can gain an understanding of the *<Solution Name>'*s background.

Release 1

Briefly describe what was contained in the first release, or provide a link to appropriate documents.

Release 2

Briefly describe what was contained in the second release, or provide a link to appropriate documents.

1. Test Case Summary

Table 4 lists the *<Solution Name>'s* scripts used for the current release (*<Release x>*). Each entry in this table should correspond to a test case that is defined fully in one section in Appendix B.

This table is critical for *DoxRunner's* Test Case and Script operations.

The Test Case ID should contain a link to the full test case which should be referenced by a bookmark structured P\_<TestCaseID>.

| Test Case ID | Test Case Name |
| --- | --- |
|  |  |

Table Test Case Summary - IR10 RSVT

1. Test Cases

<The first test case goes here>

1. Test Data

Test data for native *Microfocus LoadRunner* consists of either Virtual Table Server (VTS) or flat text files.

Test data can also be in the form of a mySQL database table (the term *LoadTest* database is used to describe this database in this document).

Test Accounts are a special form of test data, with their own rules, so any of the above three mechanisms can be used.

* 1. Virtual Table Server (VTS)

If VTS is used, describe how and in what context.

Refer to [*VTS* parameter rules](#V_VTS) for a guideline on how *DocRunner* defines *VTS* parameter rules.

* 1. Text Data Files

Flat files normally exist within a script's folder and is prepared by the scripter in preparation for each test.

These files may be able to be re-used if the data in them is not "consumed".

For data that is consumed, then the test analyst must either remove the consumed records or prepare another file with unconsumed data in order to prepare for the next test. A mySQL LoadTest database is the preferred method for data that is consumed during a test.

It is recommended that all flat text files be placed in their own folder, called an *Included Files* folder. This folder is configured in the [Configuration](#V_Configuration) section.

Refer to [Text Data File parameter rules](#V_TextDataFile) for a guideline on how *DocRunner* defines Text Data File parameter rules.

* + 1. File a

Describe the purpose of this file and what each column represents.

* + 1. File b

Describe the purpose of this file and what each column represents.

* 1. LoadTest Database

The LoadTest database is a generic term for a set of mySQL databases - normally one database per release.

The benefits over flat text files are:

1. Consumed data doesn't need to be removed - a *status* column is used to flag consumed data;
2. Use of the *status* column allows data to move from one state to another without manual intervention;
3. Data can sometimes be changed/added during a test if it is noticed that there will be insufficient records to complete the test.
	* 1. Defining a LoadTest Database

Refer to [LoadTest Database parameter rules](#V_LoadTestDatabase) for a guideline on how *DocRunner* defines LoadTest Database parameter rules.

A mySQL database requires the following to be configured. This must be done in the [Configuration](#V_Configuration) section of this document.

**Database Driver** normally: MySQL ODBC 5.1 Driver

**Database Server** example: server.domain.local

**Database User** typically: root

**Database PWD** the password to the Database User

**Database Name** example: Release10

**Database Port** normally: 3306

* + 1. Creating a new table

The following SQL can be used as the framework for creating a new table.

Columns id, status, and lastupdated should be the first three columns of all tables.

The three example columns EXAMPLE1...EXAMPLE3 show the variable types typically in use.

CREATE TABLE Release1.Addresses

(

id int(10) unsigned NOT NULL AUTO\_INCREMENT,

status int(10) unsigned NOT NULL DEFAULT '0',

lastupdated timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

**EXAMPLE1** VARCHAR(40),

EXAMPLE2 int(11),

EXAMPLE3 VARCHAR(100) NOT NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX **EXAMPLE1**\_KEY (**EXAMPLE1**)

)

ENGINE=InnoDB

AUTO\_INCREMENT=1

DEFAULT CHARSET=latin1;

* + 1. Copying a table from an earlier release

These two mySQL commands are useful when copying a table from a database to another one (eg. for a new release).

CREATE TABLE Release10.table-name LIKE Release9. table-name;

INSERT INTO Release10.table-name SELECT \* FROM Release9.table-name;

* + 1. Data Model

A data model is highly recommended, especially for complex data interaction.

Figure 1 contains an example of a data model.



Figure LoadTest Database Script / Status / Table Cross-Reference

* 1. Test Accounts

The applications under test may not necessarily need test accounts. However, if it does, and most do, then they are regarded as a special form of test data.

Some applications require a login to be unique – that is, the same test account cannot be logged in more than once. This means more care must be placed on what test accounts are assigned to which script.

Some applications have different performance profiles depending on the profile of the test account. This also means more care.

Some business processes require data to be used only by the test account that created it, or at least the same profile as the test account that created it.

All of the above often makes the management of test accounts complex.

When parameterising test account data (usually User ID and Password, but there may be more depending on the application), it is normal to configure the UserID as "unique once", with the remaining attributes configured as "same line as UserID". Hence putting the test accounts in as a flat text file is the simplest way of configuring the script.

The *LoadTest* Database can also be used to hold test accounts, and is very useful if the passwords change often and are different for each test account.

Refer to <https://www.doxrunner.com/test-accounts>

<describe how the test accounts are managed>

1. Operations
	1. Deployments

<explain the deployment process is, and how the Solution becomes ready for testing>.

<also explain any operations that the performance test analyst must execute to ensure a seamless test exercise>.

1. Test Infrastructure
	1. Test Environment

Describe the environment used to test the Solution. Put in enough detail to allow performance engineers to understand it from a performance point of view. Possibly include application(s), databases, servers, network configuration, message paths, etc.

You may need several diagrams and explanations, depending on the complexity.

Also make sure you document any differences between this environment and production. In particular anything that may impact performance.



Figure Schematic diagram of the infrastructure under test

* 1. Microfocus LoadRunner Environment

Describe the *Microfocus LoadRunner* test harness here.



Figure Schematic diagram of the *Microfocus LoadRunner* environment

1. Trouble Shooting

<use this section to document the solutions to any problems and gotchas that may be encountered>

1. Scripting Notes

These notes are examples only.

Browser Cache

Make sure you clear your browser cache before recording.

Test Data

The *LoadTest* database is used for all test data except Test Accounts. Refer to the rules in [*LoadTest Database* parameter rules](#V_LoadTestDatabase) for more details.

URLs

The URLs are summarized here.

Executing VUGen from your PC

For testing with *VUGen* on your PC, make sure the script can log in to the proxy server with your LAN credentials. Place this code prior to the very first step (use your own password and encrypt it) in *login.c*.

web\_set\_user("goodzip\\richardvolzke", lr\_decrypt("515b6b50707Cfb54214837a7ffd319c8e1"), "proxy1-1.goodzip.net.au:80");

web\_set\_user("goodzip\\richardvolzke", lr\_decrypt("515b6bdd6dcbe2ca600d05ef0636758827"), "proxy1-2.goodzip.net.au:80");

web\_set\_user("goodzip\\richardvolzke", lr\_decrypt("515b6b58ddcbe2ca000405ef0638758127"), "proxy2-1.goodzip.net.au:80");

web\_set\_user("goodzip\\richardvolzke", lr\_decrypt("515b6b58ddcbe2ca000d05ef0836758127"), "proxy2-2.goodzip.net.au:80");

Think Time

The Think Time setting in the Runtime Settings for all scripts is set to random between 75% and 125%.

The Transaction Timers table for all scripts includes a Think Time column. The values put there are determined by the guidelines set out in Table 3 so that the user's operation is more closely simulated:

| Guideline | Think Time (seconds) |
| --- | --- |
| Choosing a control from among one or two | 3 |
| Choosing a control from among three to10 | 5 |
| Choosing a control from among more than 10 | 5 - 10+ |
| Logging on | 10 |
| Logging off | 2 |
| Selecting from a dropdown list box | 5 each |
| The user needs to enter text before moving to the next step - short | 5 each |
| The user needs to enter text before moving to the next step - long | 10 each |

Table Think Time Guidelines

Recording Options

If many scripts use JSON, the *VUGen* Recording Options should be set as follows:

*HTTP Properties/Advanced/Non-Resources - Add the following three Non-Resource Content Types:*



Figure Recording Options to pick up JSON requests

Logging

It's advisable to increase the Log Cache as per Figure 5.



Figure Auto Log Cache Setting

Additional Attributes

In the scenarios in Performance Center, use the Command Line facility to override this attribute. Figure 6 shows that the OAM attribute in all scripts in the test will be set to On and, if the {aOAM} parameter is set as shown above, it will be set to "On". In this way a single scenario can cater for both cases without changing any scripts.

Important: When setting this in a scenario, make sure the value of the drop-down box below it is set to **Apply to all scripts in test**.

Refer to the [Additional Attribute parameter rules](#V_AdditionalAttributes) section for the mechanism that *DoxRunner* uses to document the rules.





Figure OAM Command Line setting in Performance Center

Login.c

All of the HTTP/HTML scripts use a single *Login.c* action block that can be located in the [Included Files](#V_Configuration) folder. That is, if the *Login.c* action block is identical across all scripts. It is in a separate action block so that it can be called during *vuser\_init.c* and at the top of *action.c*, should the iteration fail. Refer to the [*Handling iteration failure*](#R_IterationFailure) paragraph below for more details on this.

The *Login.c* action block handles the following:

* steps 10 and 20;
* the differences between OAM and LDAP login sequences;
* up to 10 retries.

To be identical across all scripts, the {gScriptName} parameter must be set to the name of the script in *vuser\_init.c*. This is done via the Global Parameters.

To be able to toggle between OAM and LDAP, an Additional Attribute called OAM must be set to Yes or No. This attribute becomes parameter {aOM} which is set in *vuser\_init.c*.

Logout.c

All of the HTTP/HTML scripts use a single *Logout.c* action block. It is not identical across all scripts (although with a small amount of effort it could be). It is in a separate action block so that it can be called during *vuser\_end.c* and at the top of *action.c*, should the iteration fail. Refer to the next paragraph for more details on this.

Handling iteration failure

Logging out and logging back in on iteration failure ensures that failed iterations don't continue because the application is at an error page while the script thinks it's OK to continue, yet ensures load is maintained by not failing the VU.

This does not work if the screen where the iteration fails does not have a logout link (eg. The *xxx* screen).

1. Common Rules

Rules defined here are common across more than one test case. If any of these rules are also defined in a test case, the rule in the test case takes precedence.

Parameter rules

Refer to <https://www.doxrunner.com/parameters>

Test data parameter rules

Refer to <https://www.doxrunner.com/test-data>

Text Data File parameter rules

Table 6 lists those *Text Data File* parameter rules that are common across multiple test cases. They will be added to those defined in any test case. In the case of any conflicts, those defined in a test case will take precedence.

The *Configuration* column contains the elements of the rule. Each element must be on a separate line and structured as per the following examples. Each file must have one and only one root parameter.

Example 1 (root parameter):

* File: salary.txt
* Next row: Random
* Update value on: Occurrence

Example 2:

* File: salary.txt
* Same line as: pSalary

| Parameter | Configuration | Description |
| --- | --- | --- |
|  |  |  |

Table Common Text Data File parameter rules

LoadTest Database parameter rules

Table 7 lists those *LoadTest Databas*e parameter rules that are common across multiple test cases. They will be added to those defined in any test case. In the case of any conflicts, those defined in a test case will take precedence.

The *Configuration* column contains the elements of the rule. Each element must be on a separate line and structured as per the following example:

* Table: cart
* Operation: Get and burn

| Parameter | Configuration | Description |
| --- | --- | --- |
|  |  |  |

Table Common LoadTest Database parameter rules

Virtual Table Server (VTS) parameter rules

Table 8 lists those *VTS* parameter rules that are common across multiple test cases. They will be added to those defined in any test case. In the case of any conflicts, those defined in a test case will take precedence.

The *Configuration* column contains the elements of the rule. Each element must be on a separate line and structured as per the following example:

* Function: retrieve\_message
* Column: Stock

| Parameter | Configuration | Description |
| --- | --- | --- |
|  |  |  |

Table Common VTS parameter rules

Parameter rules (continued)

Correlation parameter rules

Refer to <https://www.doxrunner.com/copy-of-testdataoverview>

Boundary correlation rules

Table 9 lists those *Boundary Correlation* rules that are common across multiple test cases. They will be added to those defined in any test case. In the case of any conflicts, those defined in a test case will take precedence.

The *Role* column must only contain *Request* or *Response*.

The *Configuration* column contains the elements of the rule. Each element must be on a separate line and structured as per the following example (note that there is a space after the colon):

* LB: Name=sap-login-XSRF", "Value=
* RB: ", ENDITEM,
* ReplaceOption: Boundary
* Constraint: None
* Direction: Forward
* Scope: All

| Parameter | Role | Configuration | Description |
| --- | --- | --- | --- |
|  |  |  |  |

Table Common Boundary Correlation rules

Regex correlation rules

Table 10 lists those *Regex Correlation* rules that are common across multiple test cases. The *Role* column must only contain *Request* or *Response*.

| Parameter | Role | Configuration | Description |
| --- | --- | --- | --- |
|  |  |  |  |

Table Common Regex Correlation rules

Parameter rules (continued)

Other parameters

Custom parameter rules

Table 11 lists those *Custom Parameter* rules that are common across multiple test cases. They will be added to those defined in any test case. In the case of any conflicts, those defined in a test case will take precedence.

The *Configuration* column contains the elements of the rule. Each element must be on a separate line and structured as per the following example (note that there is a space after the colon):

* Value: preprod.caneview.com
* UpdateValueOn: Once

| Parameter | Configuration | Description |
| --- | --- | --- |
|  |  |  |

Table Common Custom parameter rules

Date/Time parameter rules

Table 12 lists those *Date/Time Parameter* rules that are common across multiple test cases. They will be added to those defined in any test case. In the case of any conflicts, those defined in a test case will take precedence.

The *Configuration* column contains the elements of the rule. Each element must be on a separate line and structured as per the following example (note that there is a space after the colon):

* Format: %Y-%m-%d
* UpdateValueOn: Iteration
* Offset: 1 working day and 02:03:04 in the past

| Parameter | Configuration | Description |
| --- | --- | --- |
|  |  |  |

Table Common Date/Time parameter rules

Random Number rules

Table 13 lists those *Random Number* rules that are common across multiple test cases. They will be added to those defined in any test case. In the case of any conflicts, those defined in a test case will take precedence.

The *Configuration* column contains the elements of the rule. Each element must be on a separate line and structured as per the following example (note that there is a space after the colon):

* Format: %06lu
* UpdateValueOn: Occurrence
* Range: 200 to 500000

| Parameter | Configuration | Description |
| --- | --- | --- |
|  |  |  |

Table Common Random Number parameter rules

Additional Attribute rules

Table 14 lists those *Additional Attributes* that are common across multiple test cases. They will be added to those defined in any test case. In the case of any conflicts, those defined in a test case will take precedence.

The *Configuration* column contains the elements of the rule. Each element must be on a separate line and structured as per the following example (note that there is a space after the colon):

* Name: PreProdURL
* Value: preprod.doxrunner.com

| Parameter | Configuration | Description |
| --- | --- | --- |
|  |  |  |

Table Common Additional Attributes

Other rules

Refer to <https://www.doxrunner.com/document-other-rules>.

Configuration

This section is critical for the operation of *DoxRunner* operations.

Table 15 defines *Configuration Items* that are critical for *DoxRunner* operations to do their job. If the configuration of any test case needs to override any of these, a cut down version can be added to the test case.

| Type | Value | Description |
| --- | --- | --- |
| Organisation | Dox Runner | The name of the organisation (optional). |
| Solution | Cane View | The name of the solution (optional). |
| Project | Upgrade Cane View | The Project ID if any (optional). |
| Release | Release01 | The Release ID if any (optional). |
| Environment | SVT | The name of the environment (optional). |
| Support Folder | C:\Cane View\\_02 Support\ | The folder containing this Solution document (mandatory). |
| Script Folder | C:\Cane View\\_05 Scripts\ | The folder containing the scripts (mandatory). |
| Included Files Folder | C:\Cane View\\_05 Scripts\\_Included Files | The folder containing any included files (mandatory). |
| Timer Name Template | <Test Case ID> <Timer ID> <Timer Name> | Determines the format of Transaction Timer names (mandatory) |
| Database Name | CaneView04 | The name of the database if the *LoadTest* Database is used (optional). |
| Database Server Name | Srv007typ.com.au | The name of the server where the *LoadTest* Database is installed. |
| VTS Server Name | ServerTTT | The name of the VTS server. |
| VTS Server Port | 4444 | The port used to access the VTS server. |
| Correlate Values | Yes | Option for the Process Raw operation to correlate values (optional). |
| Validation Template LB | <title | Used when the Validation cell in the Transaction Timer table is blank. |
| Validation Template RB | </title> | Used when the Validation cell in the Transaction Timer table is blank. |

Table 15: DoxRunner Operation Configuration

Header rules

Table 16 contains Header rules that complement those in *VUGen* and that are common across multiple test cases. If they are missing they will be added to those defined in any test case. In the case of any conflicts, those defined in a test case will take precedence.

| Name | Value | Description |
| --- | --- | --- |
|  |  |  |

Table Common Header rules

SAP GUI rules

Table 17 contains SAPGUI rules that are common across multiple test cases. They will be added to those defined in any test case. In the case of any conflicts, those defined in a test case will take precedence.

| Function Name | Argument Number | Description |
| --- | --- | --- |
|  |  |  |

Table Common SAP GUI rules

Response Time NFR rules

Table 18 contains the non-functional response time rules relevant for all tests. They can be referenced by rules in the Transaction Timers table of any test case via the ID column. They can be more complex than the example here.

| Category | ID | 90% | Comments |
| --- | --- | --- | --- |
| High | H | 10 | Steps that are known to have a high response time and 10 secs is acceptable. |
| Moderate | M | 5 | Steps that are known to have a moderate response time and 5 secs is acceptable. |
| Low | L | 3 | Steps that are known to have a quick response time and 3 secs should be achievable. |

Table : Response Time NFR References

1. Obsolete Test Cases

<Move all obsolete test cases here>

1. Glossary of Terms

Table 19 describes initialisations, acronyms, and terms used in this document that are typically not found elsewhere in the organisation's documentation.

| Term | Description |
| --- | --- |
| 90% Percentile | When measuring response times, typical metrics such as Average and Maximum can be misleading. The 90th percentile (and often the 95th percentile) is used to eliminate outliers and ensure that 90% of typical usage falls within a specified response time.For example, if the response time was 2 seconds at the 90th percentile, then 90% of the response times completed within 2 seconds. The 10% that were longer than 2 seconds can be considered as irrelevant for typical usage. |
| ASM | Application Simulation Model. Includes runtime settings such as VUs, Transaction Rate, Pacing, and Think Time randomisation. |
| AUT | Application(s) Under Test – those applications that have been selected for testing. They are considered to be a part of the Solution. |
| Baseline Testing | The execution of single scripts at load levels equivalent to what is defined in the NFRs to understand how each of the scripts execute when run at load individually. |
| Load Testing | Load testing is a test that subjects an application to varying workloads to measure and evaluate the performance behaviours and abilities of the test application to continue to function properly under these different workloads. The goal of load testing is to determine and ensure that the system functions properly to the maximum expected workload. Additionally, load testing evaluates the performance characteristics, such as response times, transaction rates, and other time-sensitive issues. |
| NFR | Non Functional Requirement. Performance testing is a non-function test. Requirements include transaction volumes and response times. |
| Shakeout Test | Single script, 2 virtual users, low volume scenarios whose main purpose is to confirm that all the scripts are correctly configured (parameters, data, correlation, iterations) and able to execute correctly ensuring that the environment is correctly configured and is stable. |
| SLA | Service Level Agreements – particularly in relation to managing defects |
| SOA | Service Orientated Architecture – the name of the architecture selected to provide inter-application communication |
| Soak Testing | Soak testing is running a system at moderate levels of load for prolonged periods of time. A soak test would normally execute several times more transactions in an entire day (or night) than would be expected in a busy day, to identify any performance problems that appear after a large number of transactions have been executed. Also, due to memory leaks and other defects, it is possible that a system may stop working after a certain number of transactions. It is important to identify such situations in a test environment. |
| Solution | The Solution is equivalent to the SUT. It includes the application and infrastructure that makes up the components under a test. It is best practice to ensure the Solution has the same capacity as production. |
| Stress Testing | A type of testing that tests extreme conditions to determine boundary threshold capabilities of applications or systems. |
| SUT | System Under Test. This includes the application and infrastructure that makes up the components involved in a test. It is best practice to ensure the SUT has the same capacity as production. The Solution is equivalent to the SUT. |
| Test Harness | The *LoadRunner* servers, *LoadRunner* software, a *mySQL* database if one is used, and the network / firewall components that connect the test harness to the SUT. |
| Volume Testing | Volume testing subjects the Solution to large amounts of data to determine whether limits are reached that causes the software to fail. Volume testing also identifies the continuous maximum load or volume the Solution can handle for a given period. The volume specified is usually that forecast for a given point in time (eg. June 2015). |
| VU | Virtual User. A component of *Microfocus LoadRunner* that simulates one user. |

Table : Glossary of Terms