



Load Tests Through Jmeter & Case Study

Presented By
Rupesh Garg & Vamsi Krishna
Pothugunta,
Wipro Technologies.



Agenda

- 1 Introduction**
- 2 Define Load Testing**
- 3 Preparation of test plan for web application**
- 4 Recording user actions with Jmeter**
- 5 Customizing the test plan**
- 6 Summary**
- 7 Case Study**

Introduction

- *The success of open-source software has been remarkable, forcing even the largest commercial software vendors to acknowledge its influence.*
- *Another major factor is the current economic environment.*
- *Open source looks attractive from a cost perspective, although IT departments are finding out that open source does not always mean the overall cost of a project will go down.*
- *While software licensing costs are reduced by taking up open-source software which will let the total project budget go down, same time the expectations around delivery time, quality, and support do not change.*

Introduction

- *Current market conditions have caused software companies to cut back on spending. This has indirectly undermined one of the main arguments used by the commercial software development firms against open source: quality of support.*
- *In this document we would like to discuss about the tool “Jmeter” which is from that open source software family. Apache Jmeter is open source performance testing tool, a 100% pure Java desktop application designed to load test functional behavior and measure performance.*
- *It was originally designed for testing Web Applications but later it has expanded to test other functions as well. Jmeter is having almost all the basic features that a commercial tool is having.*

-
- *Now in the current version (2.8) of Jmeter, it supports testing the following applications*
 - *Advanced Web Test Plan*
 - *JDBC*
 - *FTP*
 - *JMS Point-to-Point*
 - *JMS Topic*
 - *LDAP*
 - *LDAP Extended*
 - *Web Services (SOAP).*
 - *TCP*

Definition of Load Testing

Load Tests are end to end performance tests under anticipated production load. The objective of this test is to determine the response times for various time critical transactions and business processes and ensure that they are within Service Level Agreements (SLAs). Load tests also measures the capability of an application to function correctly under load, by measuring transaction pass/fail/error rates.

There are various kind of Load tests, varies depending on the pattern in which they are executed against the application.



Definition of Load Testing cntd.,

Here comes various phases in performance testing.

- Requirements phase:
 - *In this phase initial details were gathered regarding application & scope of testing activities.*
 - *Test Goals & Objectives*
 - *Test Entry & Exit criteria*
 - *Defining or taking already existing SLA's*
 - *Analyzing software & Hardware requirements*



- Definition of Load Testing cntd.,

Planning Phase:

- *Preparing Test plan & Strategies.*
- *Choosing Testing & monitoring tools.*
- *Finalizing the critical scenarios, work load design, Test scripts & design of scenarios.*
- *Test readiness review.*

Environment Setup

- *Setting up test bed as per the requirements.*
- *Involves installation of tools, access to environment etc.,*

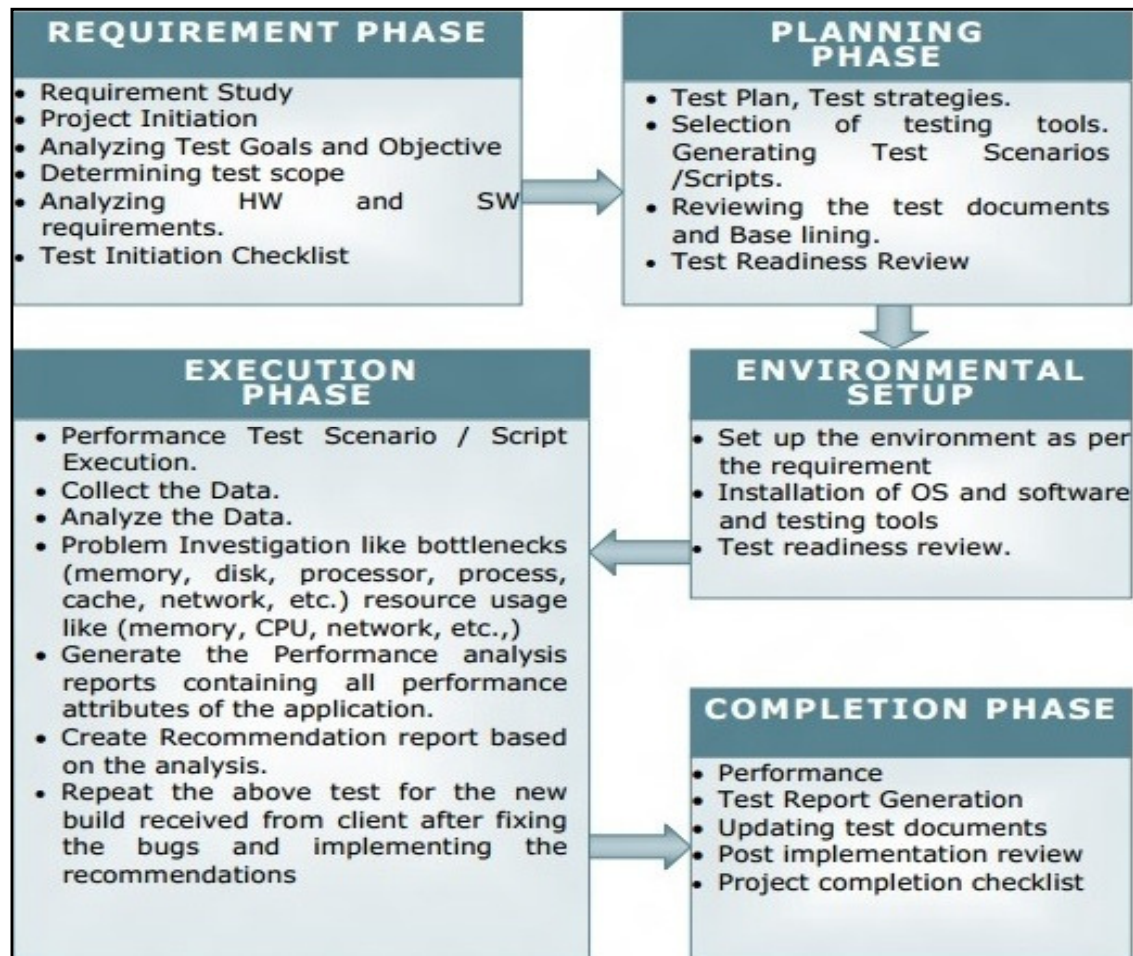
- Definition of Load Testing cntd.,

Execution Phase

- *Setting up the test data required for execution.*
- *Performance test scenario execution.*
- *Analyzing the test results.*
- *Identification of bottlenecks.*
- *Generate performance analysis report with all the performance attributes of the application.*
- *This is iterative phase, depends on how the application is responding.*
- *If the SLAs are met then proceed to next phase.*
- *Else repeat the same tests after the fix provided based on the analysis.*
- Completion phase
 - *This phase involves creation, submission & acceptance of the final performance test report.*

- Definition of Load Testing cntd.,

- Below image shows the pictorial representation of phases in order.

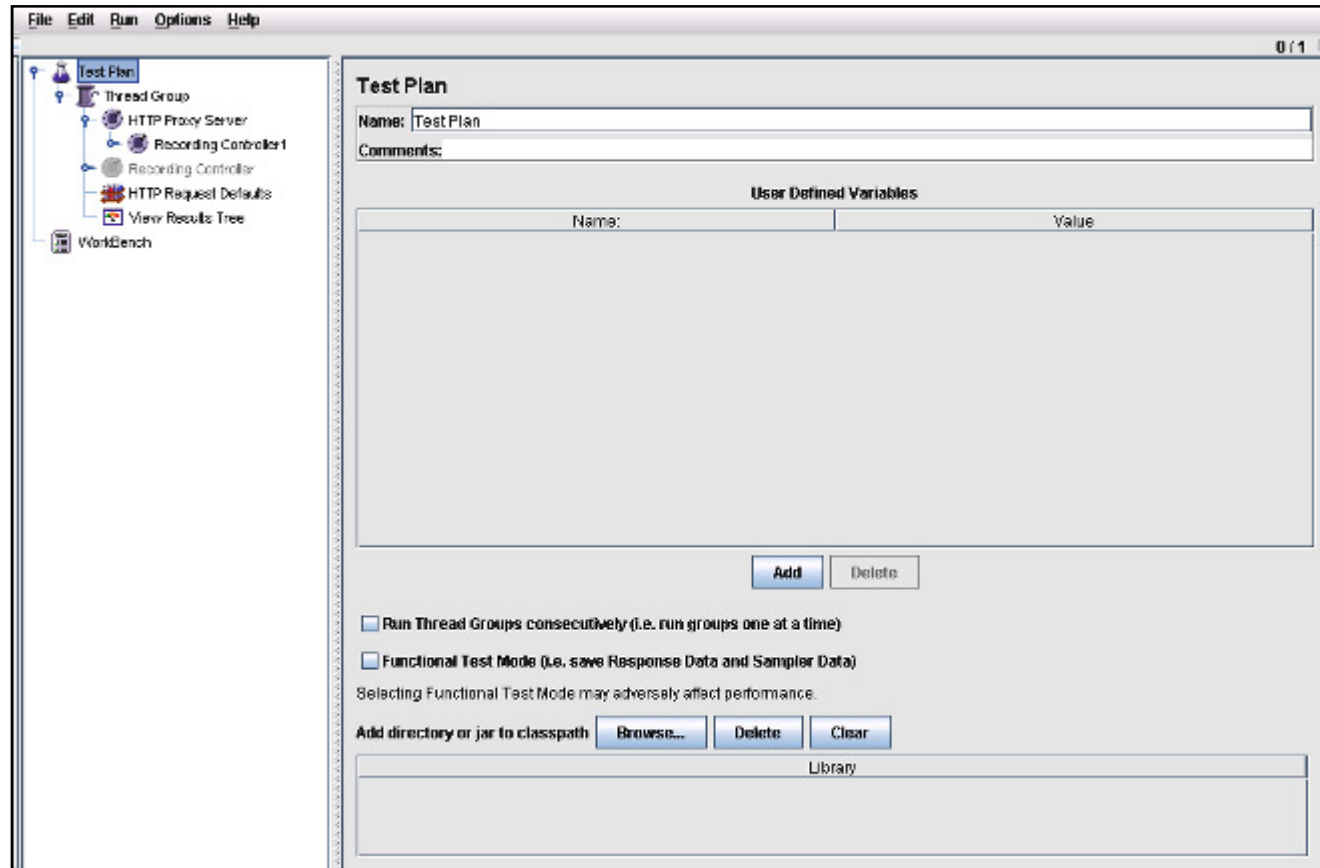


3. Preparation of test plan for a web application:

- *In this document we are providing the detailed information about performance testing of a typical web application. Process will be the same as explained in the previous section.*
- *Various elements are available in Jmeter tool through which you can design your test plan based on the requirement. It holds all the elements of the test designed.*
- *These elements are listed in this document keeping HTTP/HTML protocol in mind.*

3. Preparation of test plan for a web application cntd.,

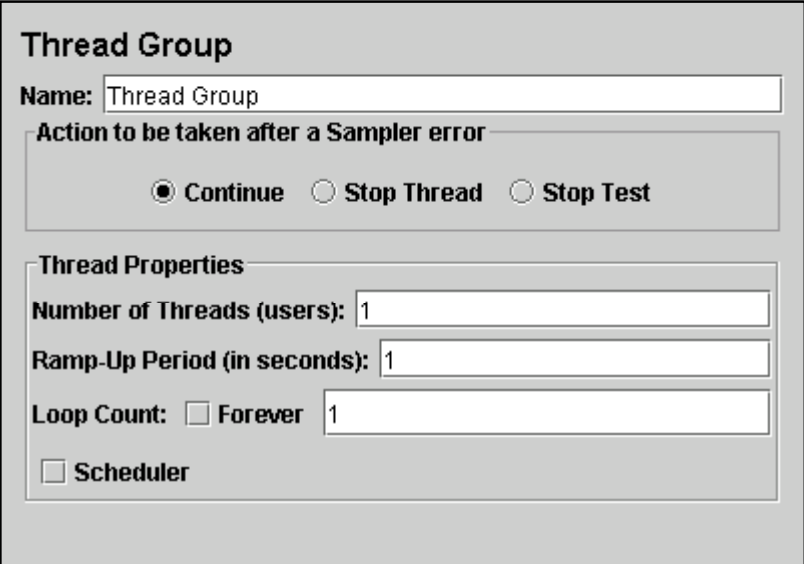
- Test Plan:



3. Preparation of test plan for a web application cntd.,

- **3.1 Thread Group:**

- *This is the basic component which accommodates all the test elements in the test plan.*
- *In thread group provides option to mention the number of users or threads we want to emulate and the rate at which they need to hit the server which we are testing and the number of iterations the users were supposed to run.*



The screenshot shows the 'Thread Group' configuration window. It has a title bar 'Thread Group'. Below it is a 'Name' field with the text 'Thread Group'. Underneath is a section titled 'Action to be taken after a Sampler error' with three radio buttons: 'Continue' (selected), 'Stop Thread', and 'Stop Test'. Below this is a section titled 'Thread Properties' with three input fields: 'Number of Threads (users):' with the value '1', 'Ramp-Up Period (in seconds):' with the value '1', and 'Loop Count:' with a checkbox for 'Forever' (unchecked) and a text field with the value '1'. At the bottom of the 'Thread Properties' section is a checkbox for 'Scheduler' (unchecked).

3. Preparation of test plan for a web application cntd.,

- *Even we can schedule the test. Please check the scheduler box so that it will list the options when to start and stop the test or you can directly give relative start and stop time.*

Thread Group	
Name:	Thread Group
Comments:	
Action to be taken after a Sampler error	
<input checked="" type="radio"/> Continue <input type="radio"/> Stop Thread <input type="radio"/> Stop Test <input type="radio"/> Stop Test Now	
Thread Properties	
Number of Threads (users):	1
Ramp-Up Period (in seconds):	1
Loop Count:	<input type="checkbox"/> Forever <input type="checkbox"/> 1
<input checked="" type="checkbox"/> Scheduler	
Scheduler Configuration	
Start Time	2009/12/08 12:14:52
End Time	2009/12/08 12:14:52
Duration (seconds)	
Startup delay (seconds)	

3. Preparation of test plan for a web application cntd.,

- **3.2 HTTP Request Defaults:**

This element lets you set default values that your HTTP Request controllers use. For example, if you are creating a Test Plan with 25 HTTP Request controllers and all of the requests are being sent to the same server, you could add a single HTTP Request Defaults element with the "Server Name or IP" field filled in. Then, when you add the 25 HTTP Request controllers, leave the "Server Name or IP" field empty. The controllers will inherit this field value from the HTTP Request Defaults element.

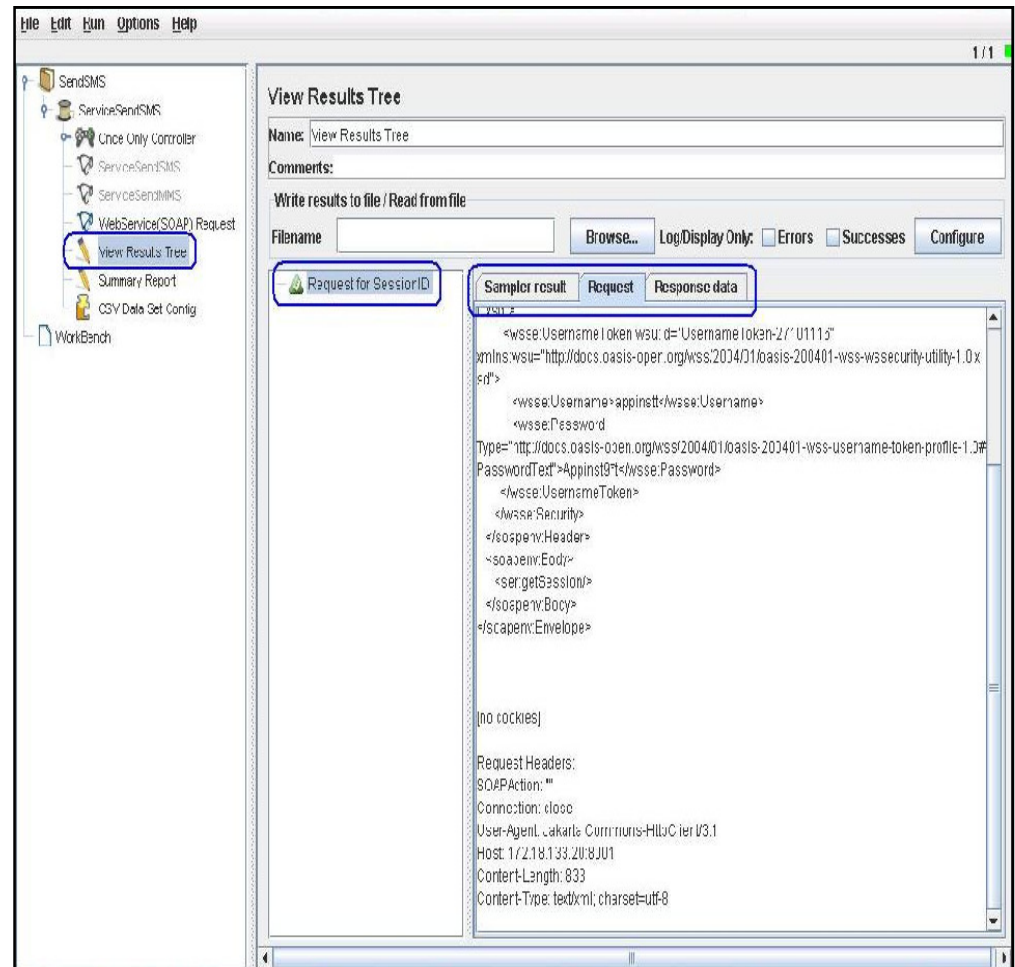
The screenshot shows the 'HTTP Request Defaults' configuration window in JMeter. The left sidebar shows a tree view with 'Test Plan' at the root, followed by 'Thread Group', 'HTTP Proxy Server', 'Recording Controller', 'HTTP Request Defaults' (selected), and 'View Results Tree'. The main panel is titled 'HTTP Request Defaults' and contains the following fields and sections:

- Name:** HTTP Request Defaults
- Comments:** (empty text area)
- Web Server:**
 - Server Name or IP:** 172.18.126.144
 - Port Number:** (empty)
 - Timeouts (milliseconds):** (empty)
 - Connect:** (empty)
 - Response:** (empty)
- HTTP Request:**
 - Protocol (default http):** (empty)
 - Content encoding:** (empty)
 - Path:** /selfcare/Landing.jsp?
- Send Parameters With the Request:**

Name	Value	Encode?	Include Equ...
MSISDN	9716498955	<input type="checkbox"/>	<input checked="" type="checkbox"/>
- Buttons:** Add, Delete
- Footer:** ☐ Retrieve All Embedded Resources from HTML Files

3. Preparation of test plan for a web application cntd.,

- **3.3 Listeners to View & Store the Test Results:**
- *Listener is the component which is used to see the response times and the content, please do add the specific listener based on your requirement adding needless components to the test plan is not recommended.*
- *Listener used to debug the test script you recorded is view results tree. Where in which, you can see the request and response. Even it is handy while doing any correlation.*



4. Recording user actions with Jmeter:

- *First to follow is add a thread group to test plan, add a non test element HTTP Proxy server to work bench.*
- *The purpose of HTTP Proxy server is to capture the request sent to server you are accessing. Give the port number as 8080 and select the target controller in which you want to collect the requests.*
- *Now move the HTTP Proxy server to your test plan.*
- *Next step is to add HTTP request default to your test plan.*
- *In HTTP request defaults, for “server name or Ip” give the Ip address or domain name and give the port number in the port field.*
- *Give the path in the field path. Please observe in the below given snapshot.*

Recording user actions with Jmeter cntd.,

File Edit Run Options Help

0 / 1

Test Plan

- Thread Group
 - HTTP Proxy Server
 - Recording Controller
 - HTTP Request Defaults
 - View Results Tree
- WorkBench

HTTP Request Defaults

Name: HTTP Request Defaults

Comments:

Web Server

Server Name or IP: 172.18.126.144 Port Number:

Timeouts (milliseconds)

Connect: Response:

HTTP Request

Protocol (default http): Content encoding:

Path: /selfcare/Landing.jsp?

Send Parameters With the Request:

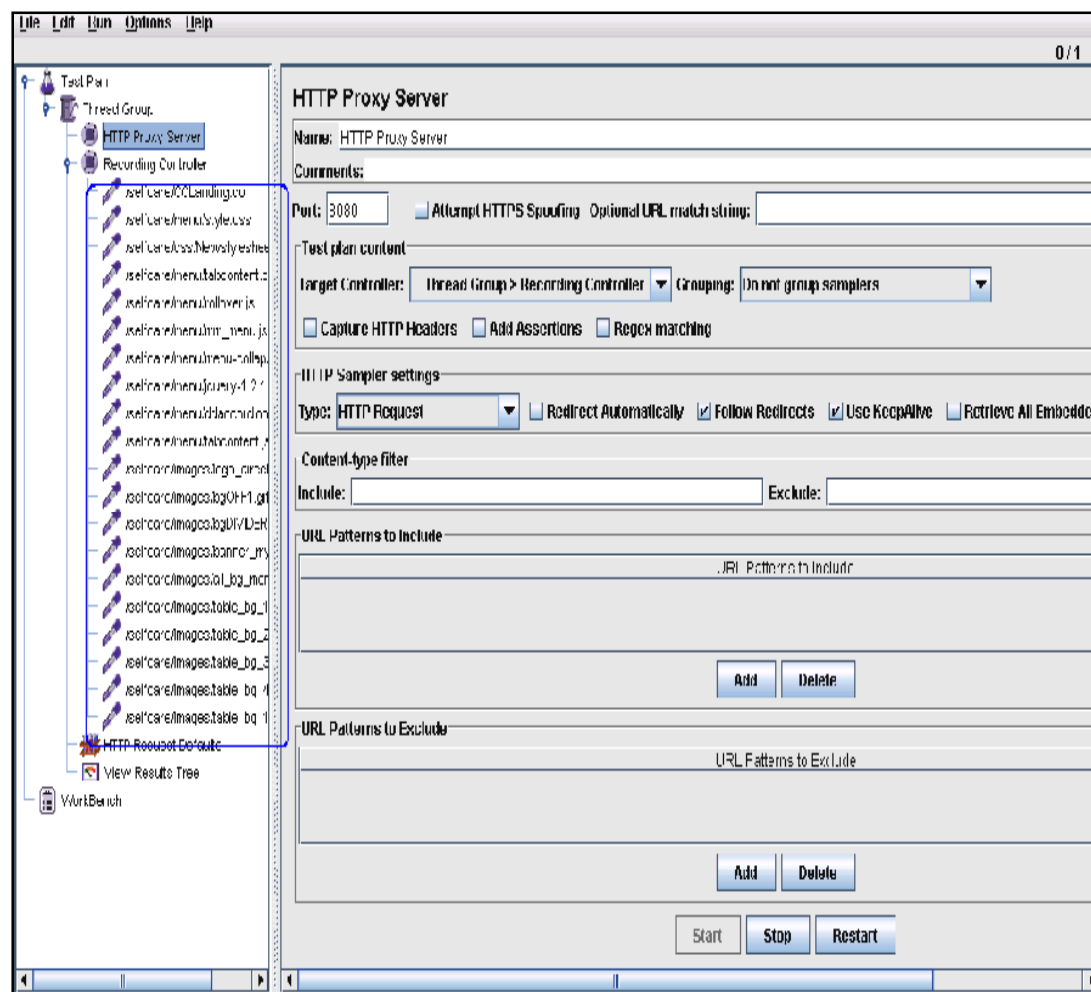
Name:	Value	Encode?	Include Equ...
MSISDN	9716498955	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Add Delete

☐ Retrieve All Embedded Resources from HTML Files

Recording user actions with Jmeter cntd.,

- Things rounded in blue line are test plan, server or IP, port number in HTTP request defaults component.
- Now open internet explorer window. Go to internet options → tools → LAN settings give server as local host or the machine on which you are doing recording, port as 8080 and click ok.
- Go to HTTP Proxy server and click on START button. Now, open internet explorer and give the URL in the browser and click go. You can observe the request would be captured under selected target controller.



Recording user actions with Jmeter cntd.,

- *Things rounded in blue line are the captured requests under selected target controller.*
- *Now add a listener based on your requirement to see the response times and response content. Listener is the component which is used to see the response times and the content, add the specific listener based on your requirement.*

5. Customizing the test plan:

5.1 Parameterization or Data driven Testing:

- *When testing any applications, you may want to check how the application performs the same operations with multiple sets of data. For example, suppose you want to check how a Web site responds to ten separate sets of data. You could record ten separate tests, each with its own set of data. Alternatively, you can create Data Table parameters so that your test runs ten times, each time using a different set of data. This way of setting parameters is called parameterization.*

Customizing the test plan cntd.,

- *Every load test should at least require parameterization. Parameterization includes passing different types of data into the application to emulate the real world users performing/entering different values.*
- *The above can be implemented in Jmeter with the help of CSV Data set configuration element.*
- *First we have to identify the value that needs to be parameterized.*
- *Then replace the value with a variable in the respective HTTP Request.*
- *As shown in the below snapshot. Select the http request in which the value need to be parameterized. Syntax to be followed for naming the variable is*
- *Ex: \${variablename}.*

Customizing the test plan cntd.,

The screenshot shows a software interface for configuring a test plan. On the left is a tree view with the following items: Test Plan 555, 555 Activation Request, 555 Request, 555 Activation Request (highlighted), Constant Timer, CSV Data Set Config, View Results Tree, Aggregate Report, and WorkBench. The main area is titled 'HTTP Request' and contains the following fields and sections:

- Name:** 555 Activation Request
- Comments:** (empty)
- Web Server:**
 - Server Name or IP:** 10.105.98.23
 - Port Number:** 8001
- Timeouts (milliseconds):**
 - Connect:** (empty)
 - Response:** (empty)
- HTTP Request:**
 - Protocol (default http):** http
 - Method:** GET (dropdown menu)
 - Content encoding:** (empty)
 - Path:** /AirceITransformation2/ProxyServices/OrderProxy/3.0/CreatePrepaidSubscriberService/CreatePrepaidSubscriberService?
 - ☒ Redirect Automatically ☒ Follow Redirects ☒ Use KeepAlive ☒ Use multipart/form-data for HTTP POST
 - Send Parameters With the Request:**

Name:	Value	Encode?	Include Equ...
senderMsisdn	\${AVAgency_Aud_Msisdn}	<input type="checkbox"/>	<input checked="" type="checkbox"/>
textMessage	WIT(\$witnessMsisdn) Success \${UniquelD} \${Witn...	<input type="checkbox"/>	<input checked="" type="checkbox"/>
TransId	\${TransId_444}	<input type="checkbox"/>	<input checked="" type="checkbox"/>
SC	\${SC}	<input type="checkbox"/>	<input checked="" type="checkbox"/>

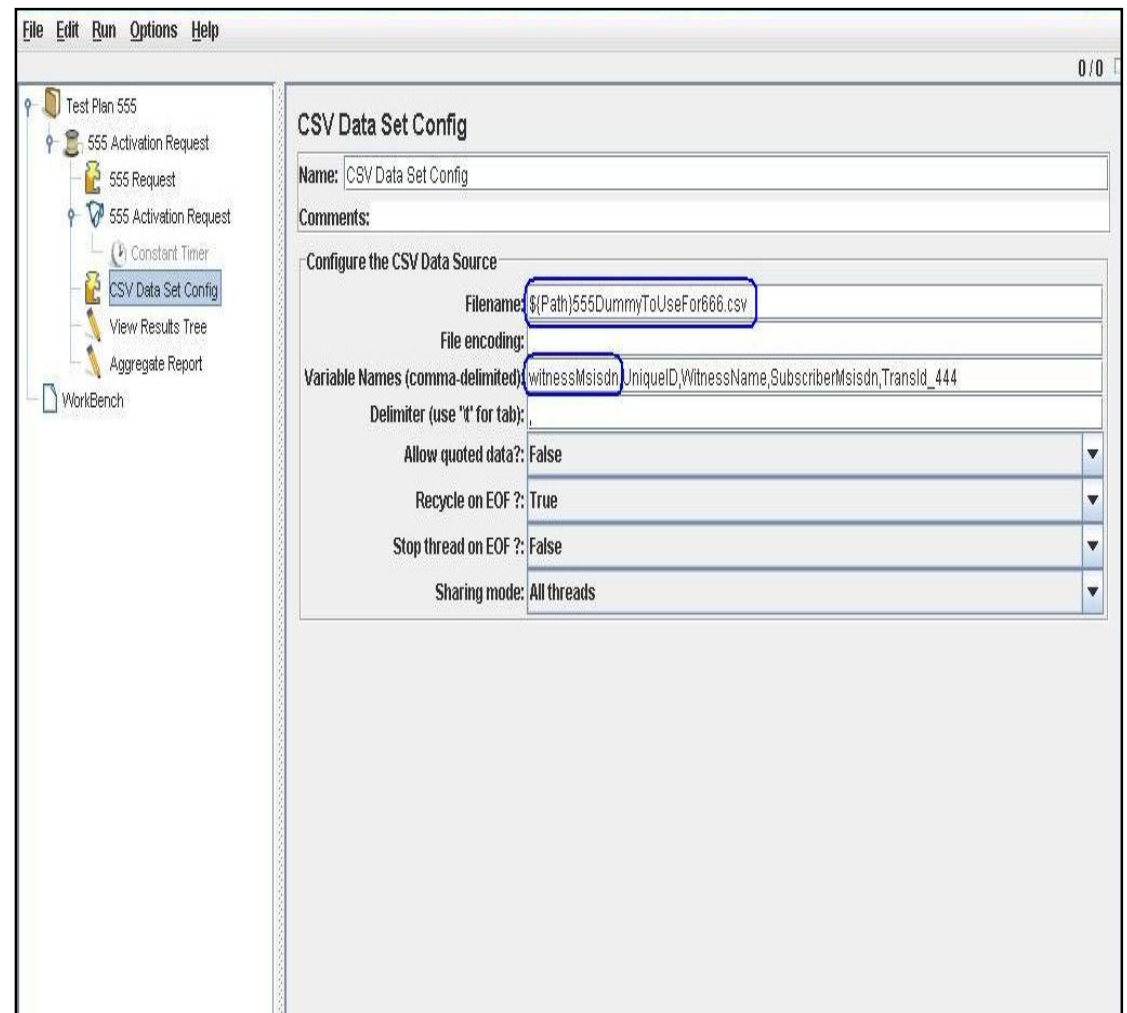
Buttons: Add, Delete
 - Send Files With the Request:**

File Path:	Parameter ...	MIME Type:
------------	---------------	------------

Buttons: Add, Browse..., Delete
 - Optional Tasks:**
 - ☐ Retrieve All Embedded Resources from HTML Files
 - ☐ Use as Monitor
 - ☐ Save response as MD5 hash?
 - Embedded URLs must match:** (empty text box)

Customizing the test plan cntd.,

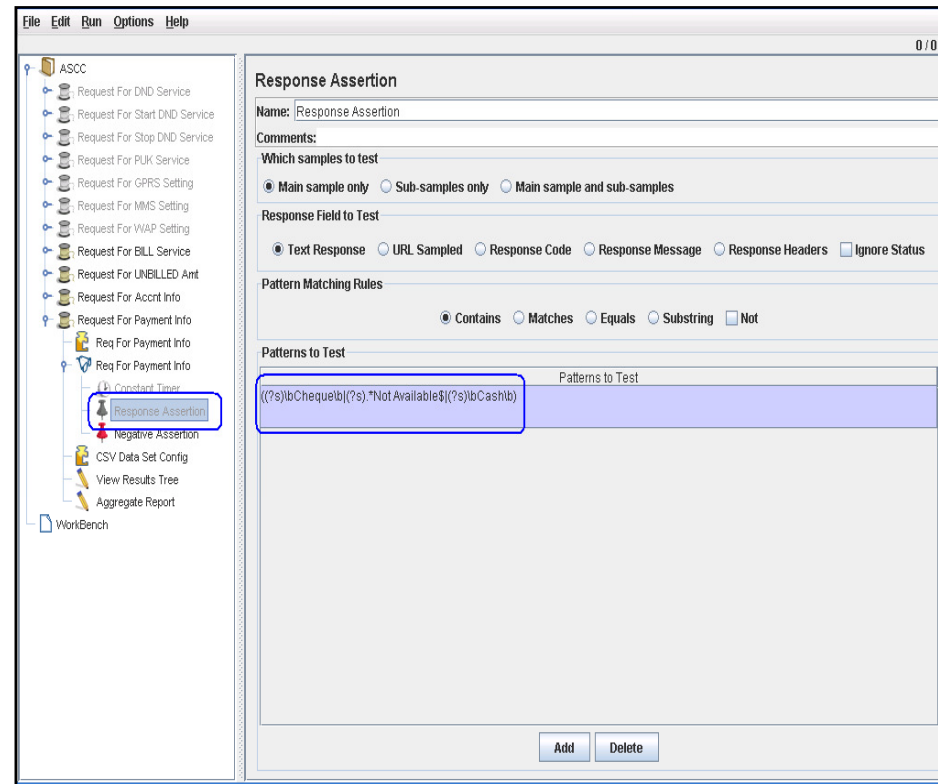
- *After replacing the value with variable add CSV data set config element to the test plan.*
- *Go to CSV data set config element.*
- *Before File name please provide the location of the data file where it is placed.*
- *Provide the variable names which are parameterized before variable names.*
- *Delimiter used in the file should be mentioned before delimiter field.*



Customizing the test plan cntd.,

5.2 .Verification Points or assertions:

- *Some times while doing performance testing you need to validate how the application is giving the responses under load. To validate the responses there are some elements provided in Jmeter those are called as Assertion elements. Different types of assertion elements are available based on the type of response you are validating.*



Customizing the test plan cntd.,

5.3. Correlation:

Correlation is the capturing of dynamic values passed from the server to the client and back. You save this captured value into a parameter, and then use this parameter in the script in place of the original value and when it makes requests of the server, send this new, valid value back to the server.

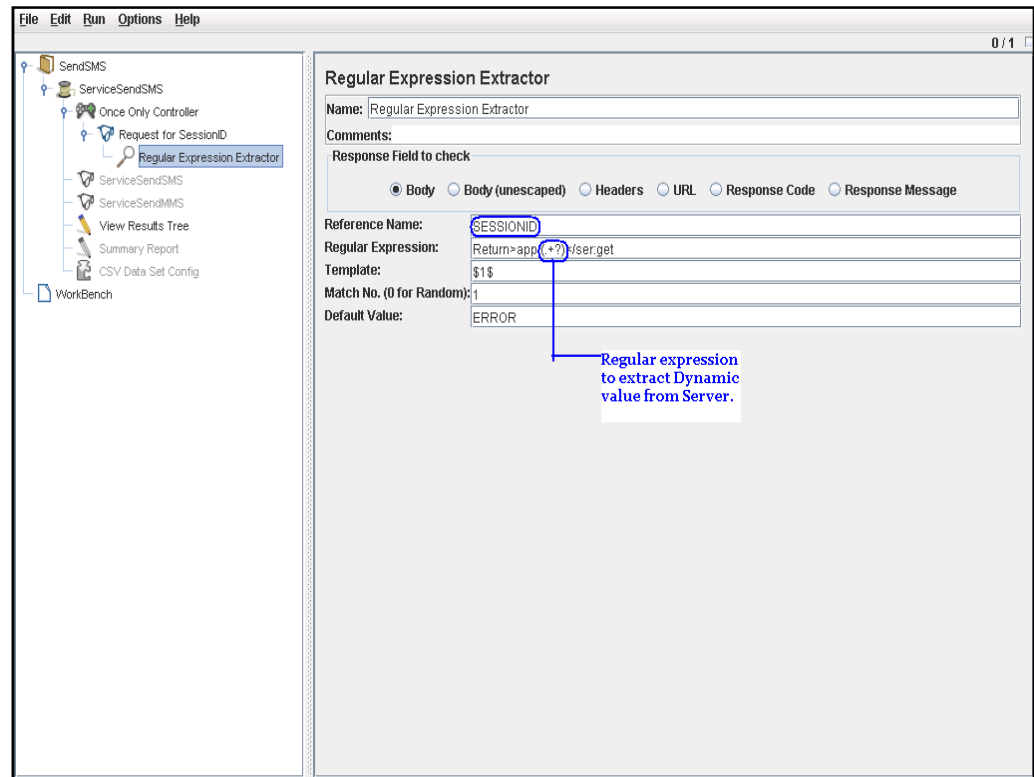
Correlation is where the script is modified so that some of the hard coded values in the script are no longer hard coded. Rather than sending the original value to the server, we may need to send different values.

Correlated data is data which is sent to the Client from the Server, and later sent back to the Server by the Client.

For example, the original recorded script may have included the server sending the client a session identification number, something to identify the client during that particular session. This Session ID was hard coded into the script during recording.

Customizing the test plan cntd.,

During replay, the server will send the client a new Session ID. You need to capture this value, and incorporate it into the script so you can send it back to the server to correctly identify yourself for this new session. If you leave the script unmodified, you will send the old hard coded Session ID to the server. The server will through an exception as the session id is invalid, or unknown, and so will not send the pages that have been requested.



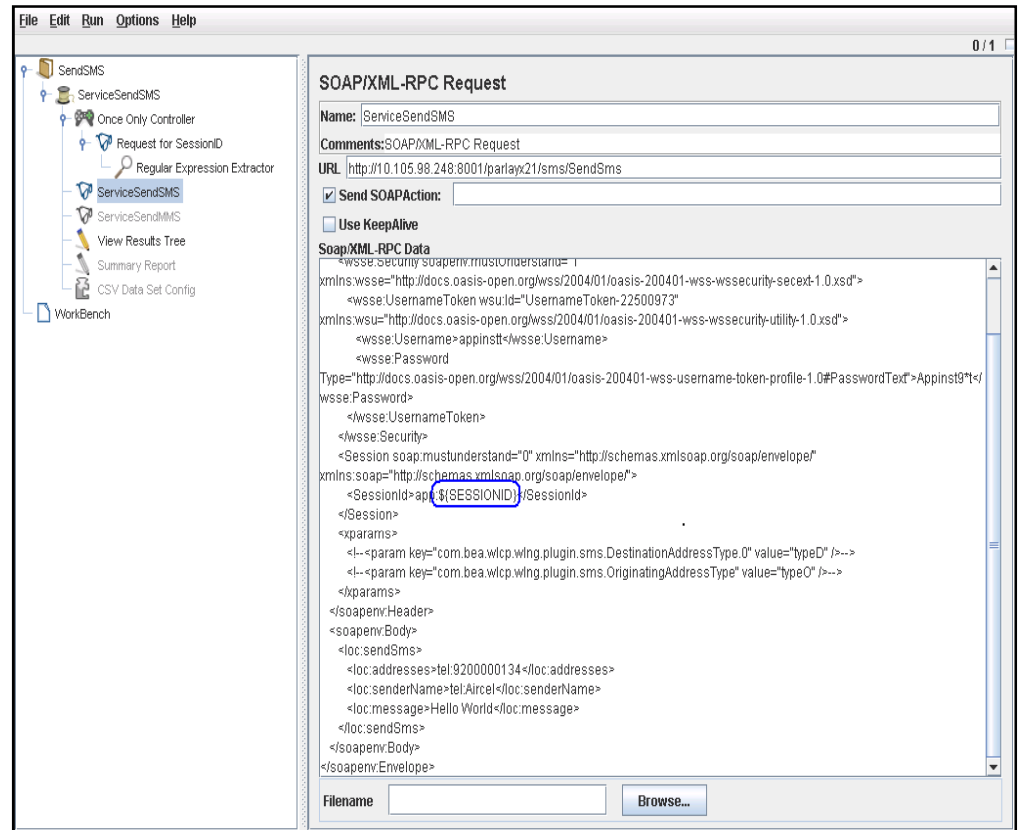
Customizing the test plan cntd.,

Explore the request for which server is sending dynamic responses. For that request add regular expression extractor available in post processors. Mention the variable name into which you want store the extracted value. Give the regular expression based on the type of value that needs to be extracted in between left and right boundaries before regular expression field. Field Match no represents the instance which you want to match. If the desired value is not extracted then the default value will be substituted in the variable in the above Snap shot ERROR is given as the default value.

The variable name has to be placed in the subsequent request in place of dynamic value, so that server will recognize the current session that extracted in to the variable. Below figure the dynamic value is replaced with the variable name into which we have taken the dynamic response from the server.

Customizing the test plan cntd.,

This concludes the general preparation of a web test plan and customization. There are still more elements available, use the elements based on your requirements. The more work Jmeter doing makes it lesser accurate so prepare the test plan accordingly.



6. Summary:

Overall as an Open source tool, it provides almost all the features that a commercial tool is having currently in the market. This tool will be very handy for small and medium budget projects where budget allocated for the tool licensing is less. Also it is easy to learn and use.

6. Summary:

Pros:

No Tool licensing costs.

Easy to use, especially for the protocols like Web services, FTP, JDBC.

Testers can easily learn & implement.

Using this tool we can do record & replay the tests.

Correlation & Data driven testing is possible through this tool.

Cons:

As it is an open source tool, support for the tool is not available.

Have to rely on the Apache User manual & Support forums.

Compared to commercial tools available in the market, usability of the tool is difficult if you are working on complex web applications where correlations are more.

Customized results will not be available from this tool.

7. Case study:

This is the tool we used in performance testing of the key modules in Aircel Transformation testing assignment through which we reduced the licensing cost involved in procuring test tools. The procedure which we followed involves lot of manual intervention as the application we tested is an integrated application. The application for which we measured the performance is given below.

Tool Jmeter emulates retailer and distributor. From the tool we are sending the request for registration and activation. Process of registration starts by sending 111 requests (Registration) which retailer use to do for requesting to activate services to a new sim. Logging is enabled at various levels to calculate the response times. Same for activation also but here the sender will become distributor. Based on the input parameters it will do the intended action.

Case study:

Here instead of retailer, requests were sent by Jmeter to the server, excluded network elements. Requests were sent by parameterising the new sim number. Once the request reaches esb the transactions we used to measure are

Time at which request reached Esb.

Time at which IVR flag was updated in CRM DB.

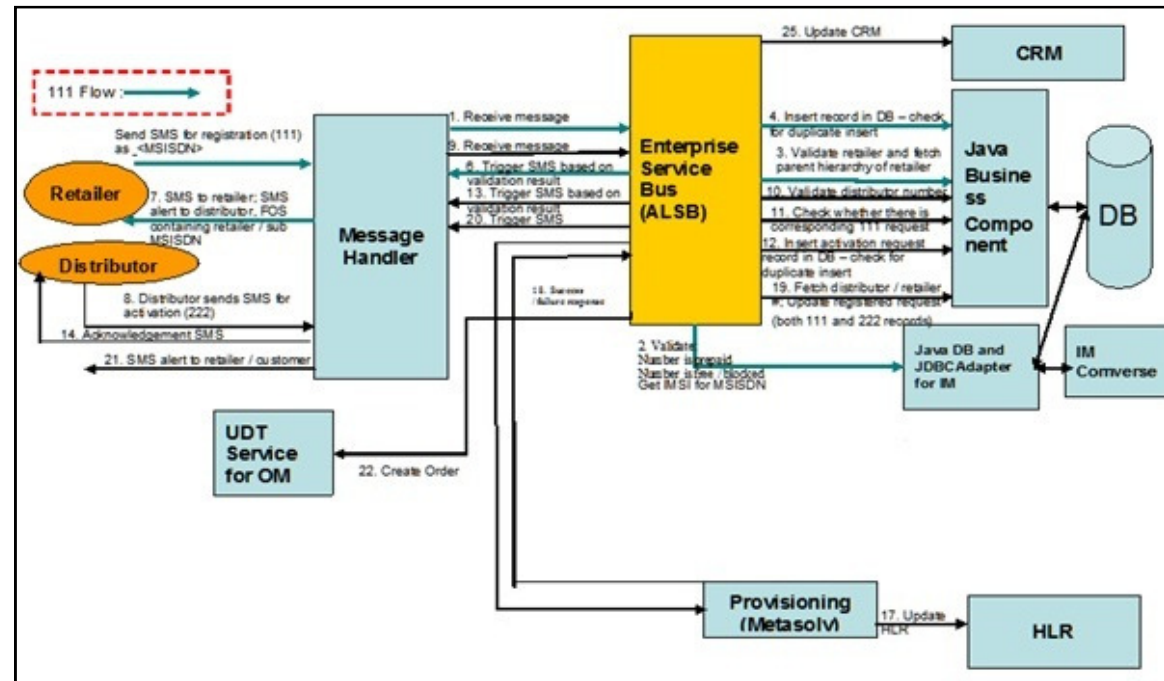
Time took to update registration request in activation DB.

Finally messages will be delivered to retailer, distributor regarding the status of the request.

All the times measured above are manually through the logs from the respective servers.

The test conducted to measure the transaction response time at various ends.

By using Jmeter we were able to simulate the above core telecom application to validate the end to end performance. Though manual intervention is required to collect the data points at various tiers, we have reduced the tool licensing costs by using this without compromising on the quality of the outcome.



8. References:

1. *Apache Jmeter by Emily H. Halili*
2. <http://jakarta.apache.org>



Thank You

