Paper 474-2013

A Case Study of Tuning an Enterprise Business Intelligence Application in a Multi-OS Environment

Fred Forst - SAS Institute, Cary, NC

ABSTRACT

A corporation is planning to expand its use of SAS® Web Report Studio (WRS) by increasing its user base. Two questions emerge: 1) What happens to SAS Web Report Studio response time as users are added? 2) What tuning modifications can be used to improve performance?

This case study uses a SAS Web Report Studio and utilizes a WRS simulation technique and 'grows' the workload and examines WRS response times. The SAS EBI environment utilizes IBM zEnterprise® BladeCenter® Extension (zBX) zBX blade technology for the SAS Mid-Tier using IBM WebSphere® on Linux. Logs from WRS, WebSphere, NMON, and the server-tier are parsed and stored in a SAS data set for all analyses. Many stress test runs are executed, modifying various tuning parameters. The analysis shows the effects of tuning and offers insight on tuning best practices.

INTRODUCTION

Acronyms Used Throughout This Paper

- HP ALM – Hewlett Packard Application Lifecycle Management
- WRS – SAS Web Report Studio
- FDS – SAS Federated Data Server
- WAS – Websphere Application Server
- EBI – SAS Enterprise Business Intelligence
- nmon – Nigel's Monitor

As more and more users are added to an EBI workload, performance eventually declines. What can be done to minimize the worsening WRS response times other than investing significant capital by adding more capacity (i.e. more processors, more memory etc)? This case study uses tools to generate and grow a WRS workload and measure response times at each iteration. As response times deteriorate, various tuning techniques are incorporated and the results are measured and stored in a 'master' SAS data set for analysis and reporting. Each user goes through these distinct steps when generating reports (more about the hardware and software environment later):

1. Logon to WRS
2. Request a report
3. Think time[1] of 1 minute
4. Repeat steps 2 and 3 10 times.
5. Logoff WRS
6. Pause 1 second
7. Repeat steps 1-6 for three hours

[1] The time a person would naturally pause (to think) between selecting reports to display.

These various parameters (that is, length of think time, number of reports to generate, duration of test, etc.) are all easily modifiable in the HP ALM (Application Lifecycle Management) Performance Center product, which is generating the WRS workload. HP ALM, in effect, is simulating a human executing the above steps.

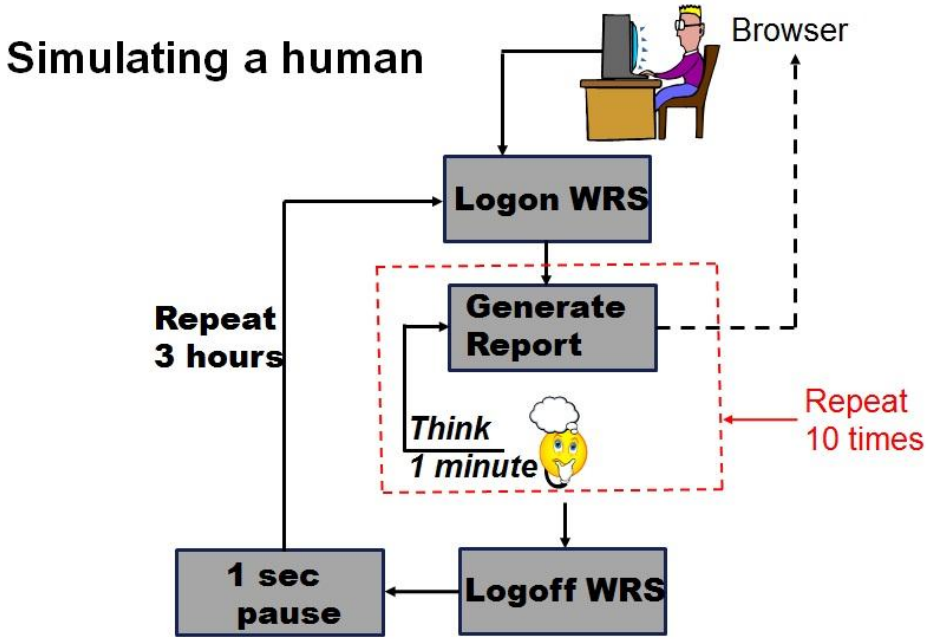


Figure 1 – Simulating a human

There are three tiers in the EBI design.

1. The client tier – In this scenario, this is a user with a web browser. This simulates that by using HP ALMPerformance Center to generate the workload.
2. The middle tier – The middle tier contains the web applications (WRS) and infrastructure and the web application server (IBM WebSphere Application Server).
3. The server tier – The server tier contains the metadata server, which is the heart of everything, OLAP server(s), and the OLAP cube data.

Characteristics of each tier:

- Middle tier:
    1. Hardware is an IBM 2458-002 with 6 HX5 Blades. The SAS mid-tier application(s) occupy a single blade
    2. 12 processors, 2.1Ghz, and 64Gb memory
    3. OS is Red Hat Linux version 6.2
    4. WebSphere version 7
- Server tier:
    1. IBM zEnterprise z196 model 710, 5.2Ghz CP
    2. Two dedicated CPs (later increased to 3)
    3. 48Gb memory
    4. z/OS version 1.13
    5. SAS version 9.3
    6. IBM System Storage DS8800 subsystem

Various logs are parsed and merged after each simulated run and the resulting data is stored in a SAS dataset for analysis. The logs are:

- NMON output from the mid-tier
- Tivoli Performance Viewer (TPV) log from the mid-tier
- Metadata server log from the server tier
- OLAP server log(s) from the server tier
- SAS Web Report Studio log from the mid-tier

Below is a high level view of the 'right out of the box' environment. This is the starting configuration from which the tuning begins.

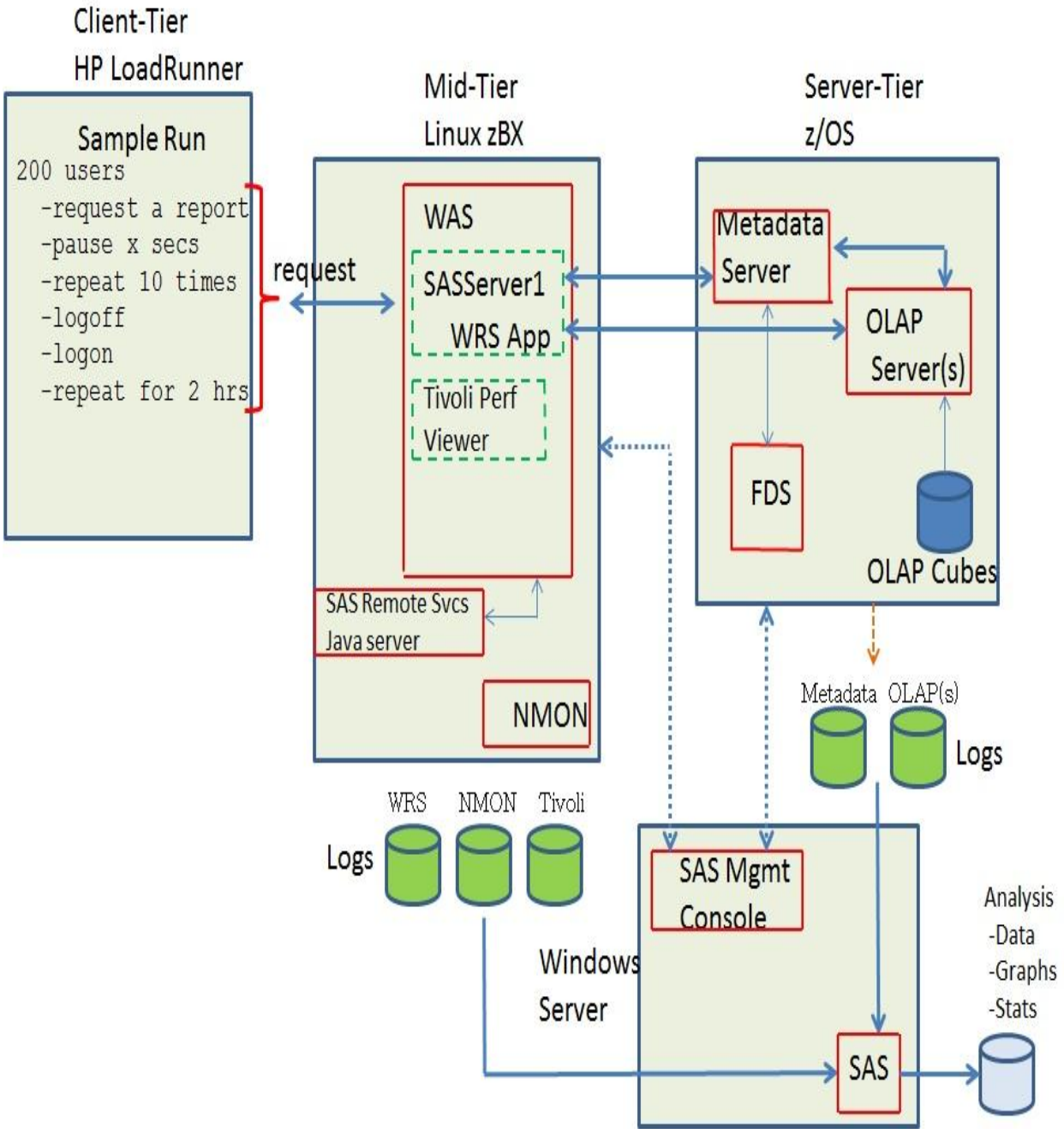


Figure 2. High Level View of the Configuration

The windows server is used for the analysis of the test runs. Other than a convenient place to run the SAS Management Console, is not part of the WRS application being benchmarked.

BACKGROUND

The executions start with the installation default settings. The user base begins with 25 users and is incremented by 25.

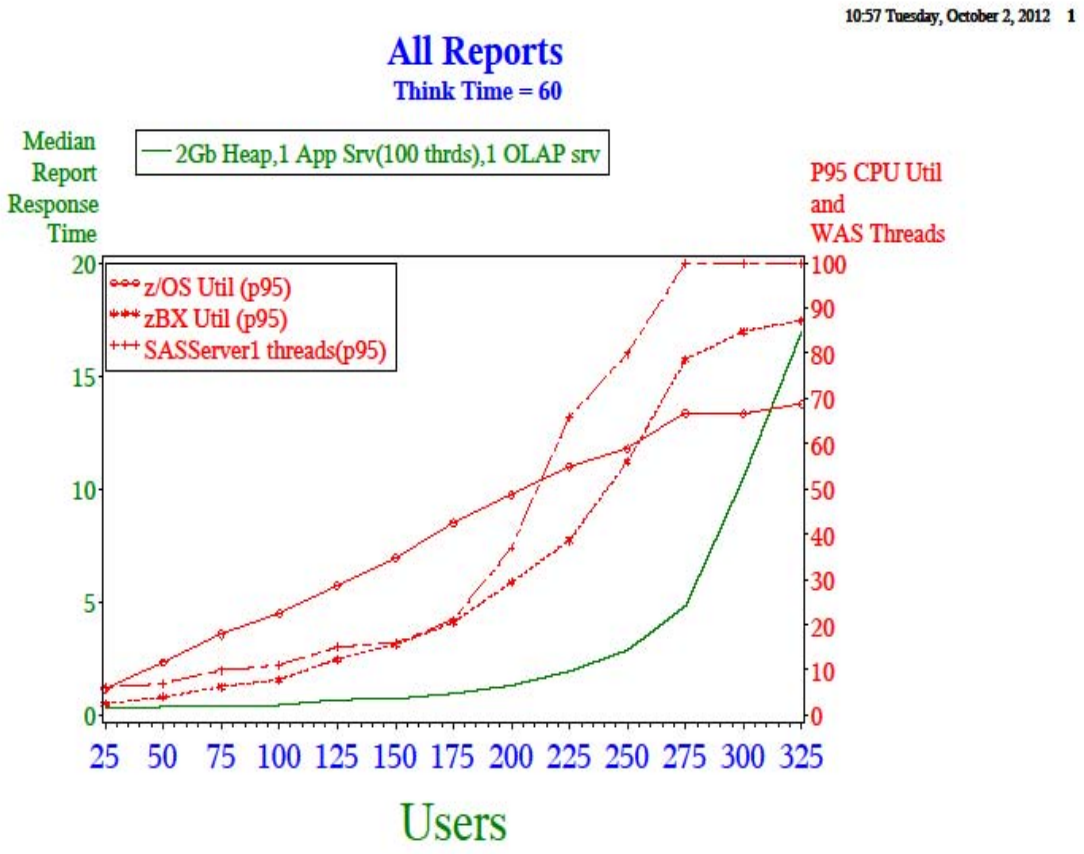


Figure 3. Out of the Box Results of Up to 325 Users

It is apparent that when the number of users reaches 250-275 the median report response time (solid green line) quickly worsens. Also plotted in Figure 3 are useful resource utilizations (in red) from both the server tier (CPU utilization – solid line marked by a diamond) and the mid-tier (CPU utilization – dotted line marked by a star) and the number of active webcontainer threads (dotted line marked by plus). The CPU utilizations are both below 90% busy, so that is not likely the bottleneck. Note that the active web container threads is hitting its maximum configured value (100) around the same time performance increased, at 275 users. Increasing this to a maximum of 200 active threads will surely solve the problem (we hope). This change is made from the WebSphere admin console and we run the same suite of tests again. The following graph shows the result.

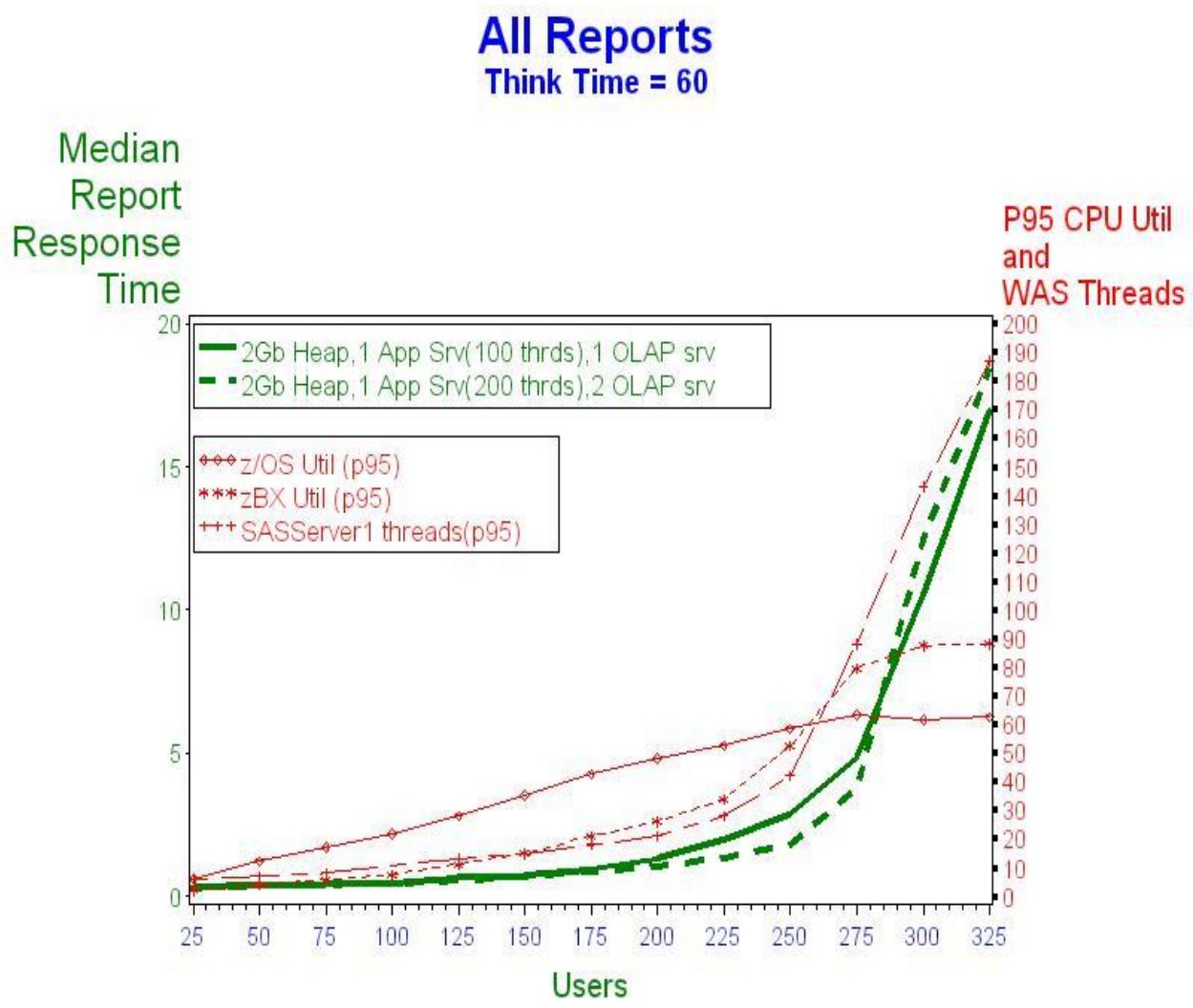


Figure 4. First Tuning Iteration – More OLAP Servers and More WAS Threads

It appears that doubling the number of maximum web container threads as well as increasing the number of OLAP servers to two had little effect on median response time. This new configuration (green dotted line) shows only a modest improvement from 200 to 275 users and there seems to plenty of capacity left on the server tier (red diamond line). The mid-tier cannot send work to the server tier fast enough, as supported by flat z/OS CPU use after 250 users. This makes it appear the bottleneck is still on the mid-tier. The search continues. Both response time curves follow almost exactly with the active thread count line. This suggests that a lack of application server threads might be the problem. With a single application server, all web applications request threads from the same thread pool. With SAS Web Report Studio, when a request is made to generate a report, an http request from SAS Web Report Studio is sent to the application server. This request, in turn, makes a request to the SAS Content Server, which is also running in the same application server. As more and more requests come in from SAS Web Report Studio, the thread pool gets exhausted and there are no available threads for the content server.

If the content server had its own thread pool, then perhaps it does not see the contention for threads. One way to do this is to create a second application server, which means it has its own thread pool, and move only the content server to this new application server. Below is an updated configuration showing this.

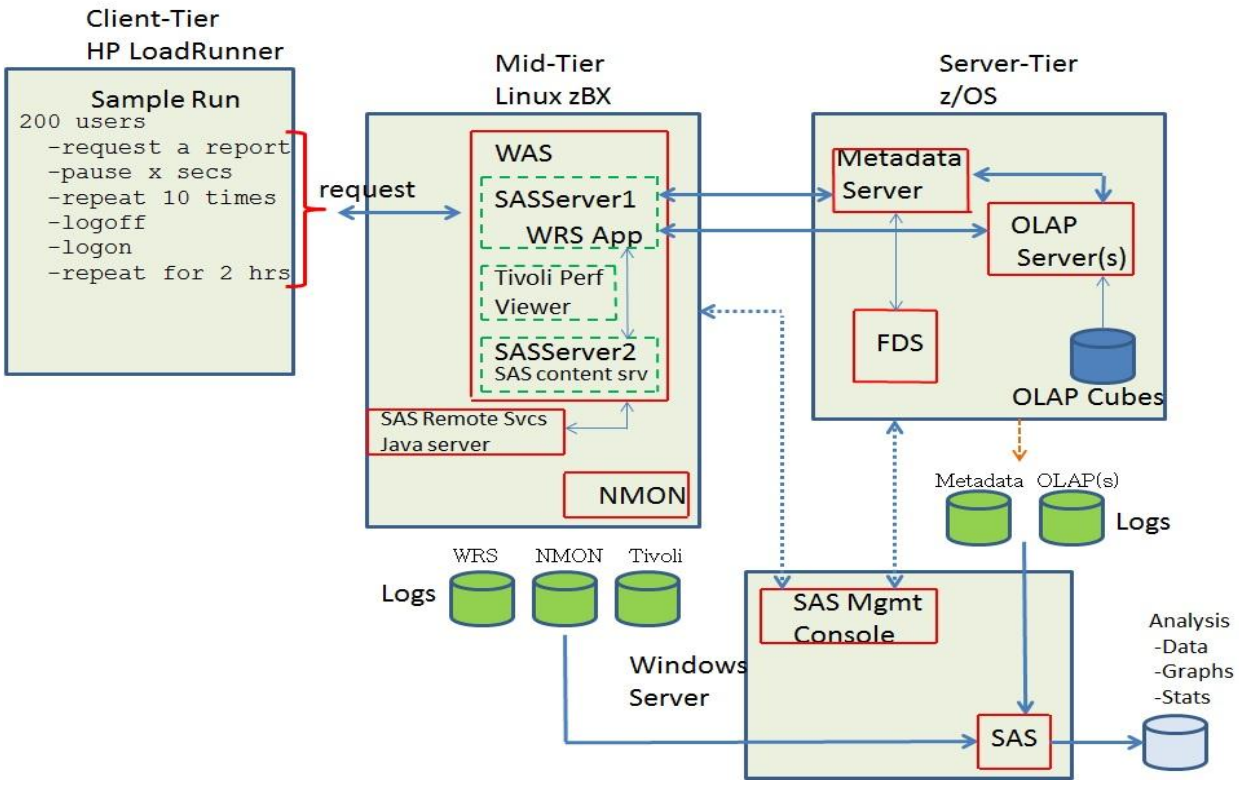


Figure 5. Modified Configuration – SAS Content Server in its Own Application Server

Now continue testing with this new configuration:

- ✓ Two application servers each with 2Gb heap and its own thread pool with a maximum thread count of 25 threads each.
- ✓ Four OLAP servers

Begin testing at around the user level where there were problems before, at 300 users. Figure 6 shows the result of adding 300-450 users in increments of 25 users. There are some important results to observe from this iteration:

1. The green dotted line shows the median response times for 300-450 users. Notice the dramatic improvement in response time.
2. Even as the number of users grows beyond 300 the active thread count has gone from 100 under the old configuration to around 25 under the new configuration while vastly improving response times.
3. The zBX CPU utilization has now dropped to less than 20% from 80%.
4. The z/OS CPU utilization is steadily climbing from 300-450 users as it receives more requests from the mid-tier now that the bottleneck has been eliminated.

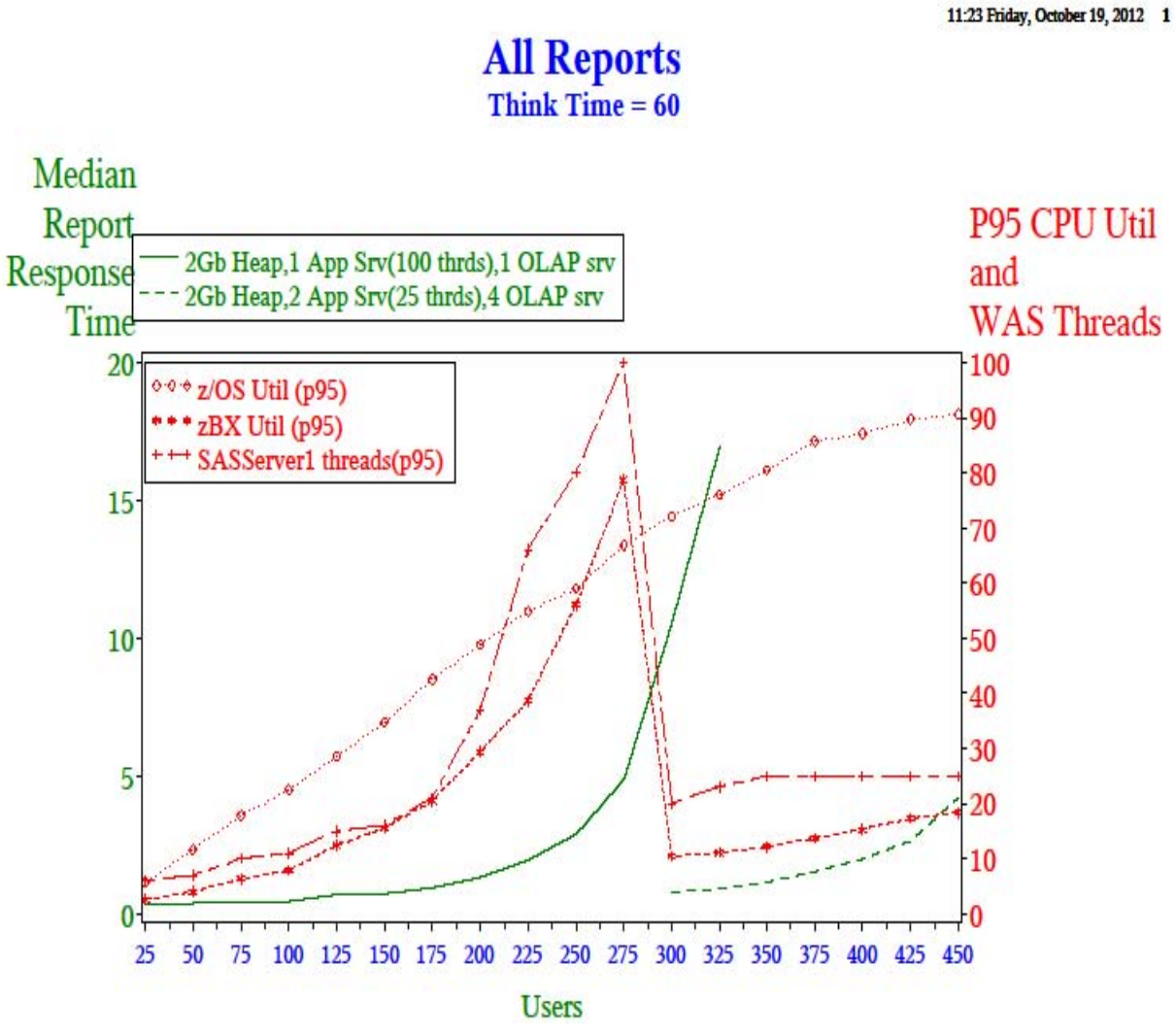


Figure 6. Results of Next Tuning Iteration

Under the new configuration, response time at 450 users is approximately the same as what was experienced at 275 users (around 4 seconds) under the first configuration. With 450 users, there appears to be plenty of CPU capacity remaining on the mid-tier (zBX), but the server tier (z/OS) with only two processors is reaching its maximum.

An explanation of how the z/OS CPU utilization is calculated is in order. The RMF interval is 2 minutes and the RMF cycle is 0.5 seconds. This means that every 0.5 seconds the CPU utilization is obtained and every two minutes the *average* of these 240 data points is calculated. The number plotted on the graph is the 95th percentile of these two minute averages. Since RMF calculates the average, and not the maximum, it is very unlikely this number will reach 100%, especially with short periods of time of no activity due to think time. The application server threads may also be a limiting factor as the 95th percentile is reaching 25, which is the maximum configured. From 300 to 450 users the response time more than doubled (although vastly better than the first configuration). Can this be improved even further? For the next attempt, you will use:

- 4Gb heap
- 2 application servers with 50 threads each

- Four OLAP servers
- 3 z/OS processors

This iteration will begin at 475 users with the outcome below.

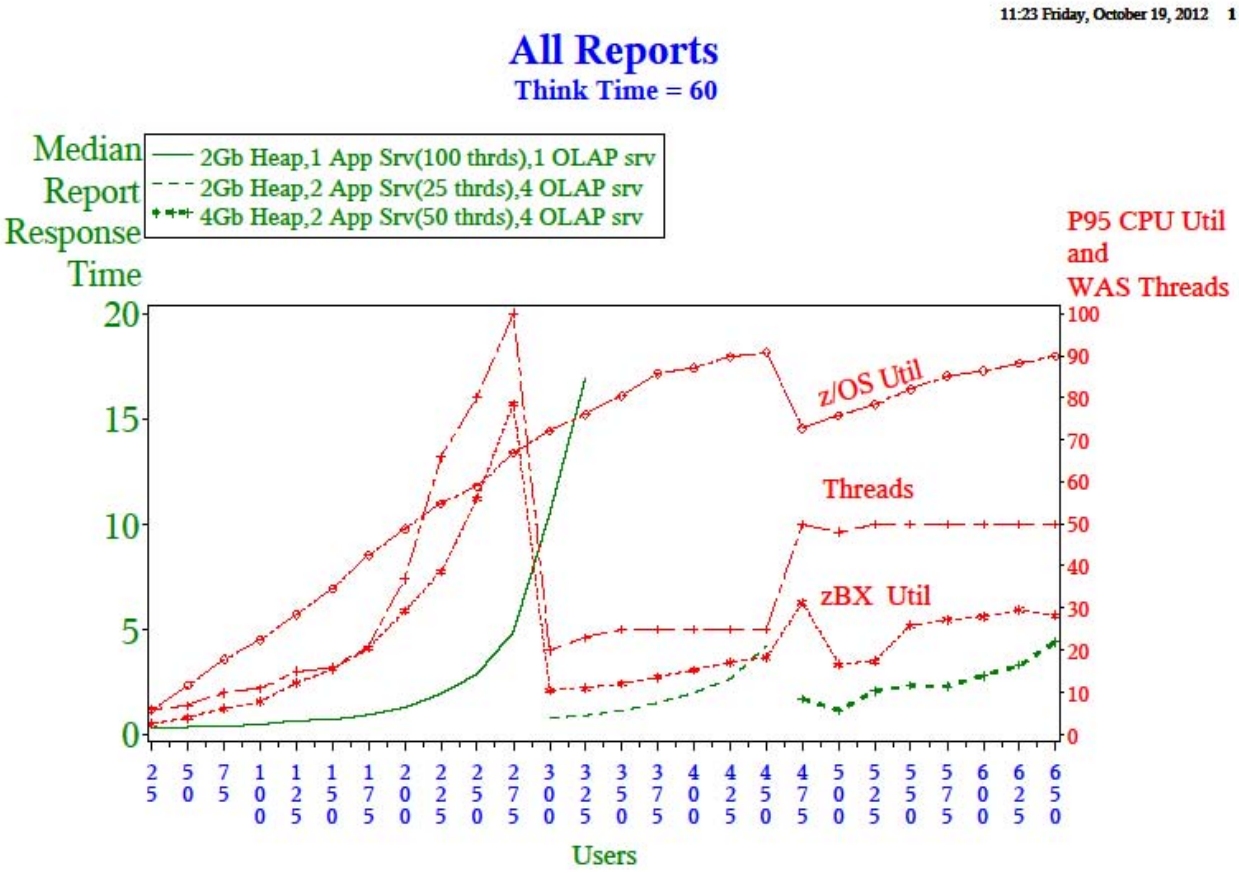


Figure 7. Tuning Iteration #4 – Larger Heap and More Threads

At 475 users the response time is about half of what it was at 450 users under the previous configuration. Also note the drop in z/OS CPU utilization at 475 users, due to adding one processor. When this configuration is continued up to 650 users, it appears the response time at 650 users is about the same as it was at 450 users (around 4-5 seconds).

Although this configuration once again improves median response times, you might wonder if increasing the application server maximum thread count to 75 might 'squeeze' out more improvement on performance. Judging by the active threads red line, it appears it is hitting the 50 thread maximum. It is worth noting that the application server that contains WRS (SASServer1) is the server reaching the thread max. SASServer2, housing the SAS Content Server, has very low thread usage.

Under this scenario you have:

- 4Gb heap
- 2 application servers with 75 threads each
- Four OLAP servers
- 3 z/OS processors

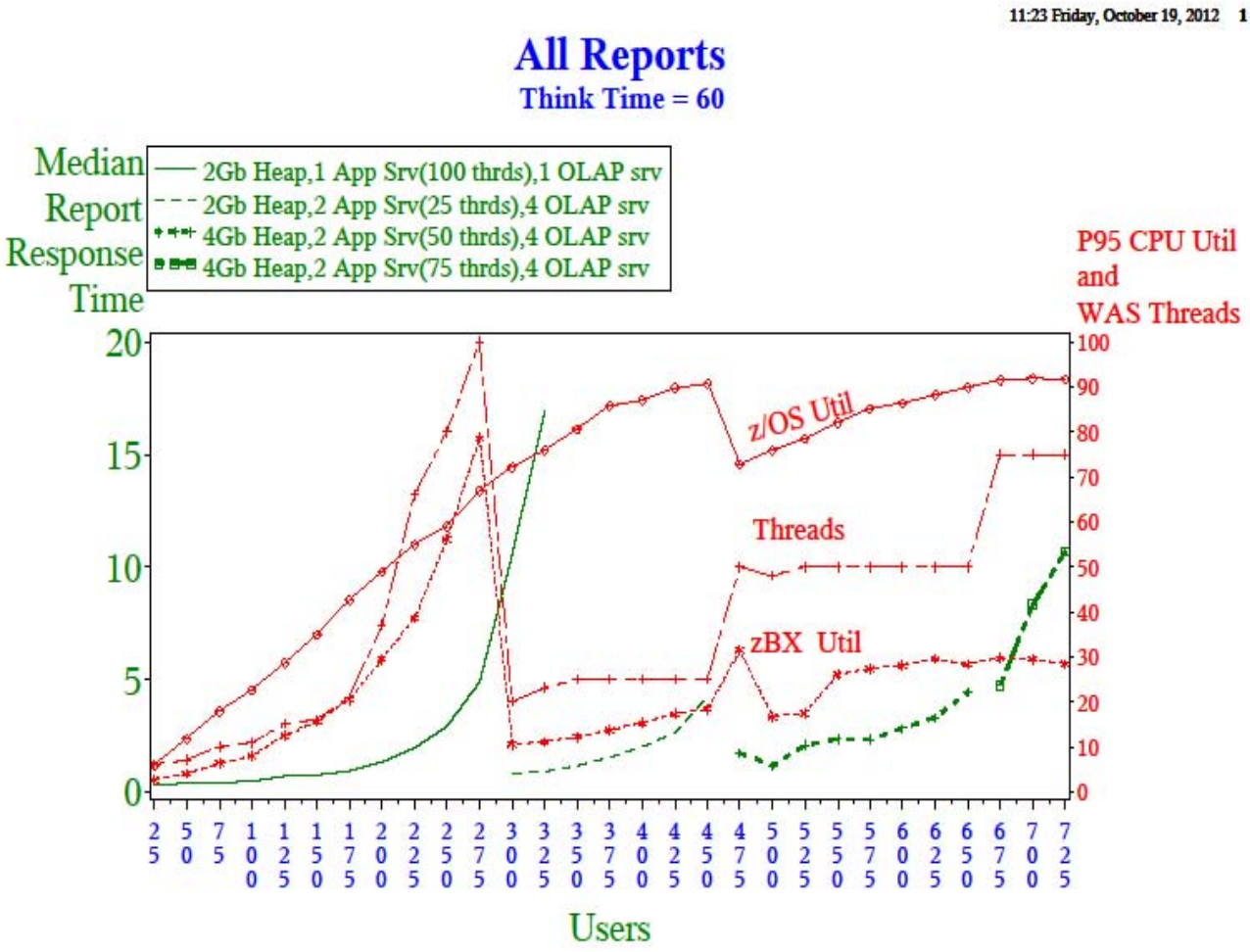


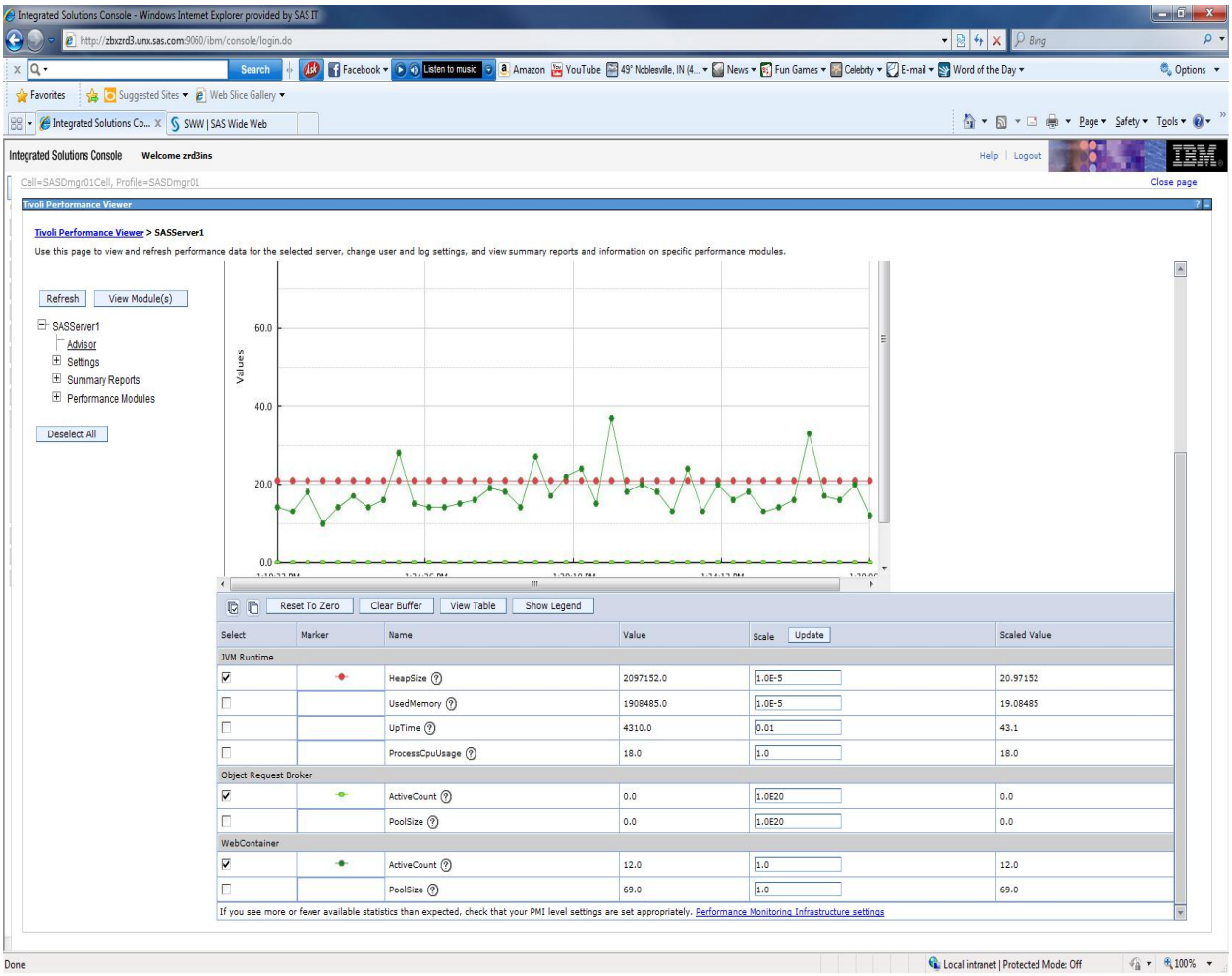
Figure 8. Final Tuning Iteration

As seen by the rightmost green line, this iteration did not improve things and the steep slope means that response time is quickly increasing, possibly due to the server tier reaching capacity.

MONITORING TEST RUNS WITH THE TIVOLI PERFORMANCE VIEWER

The WebSphere administrative console includes a Tivoli Performance Viewer (TPV) that provides a way to monitor the WebSphere resource usage during a test. It displays and graphs, as shown below, including:

- Current heap size
- Current memory size (amount of heap memory being used)
- WebContainer active thread count (which is useful so that you can see if your maximum thread limit is high enough



Display 1. Tivoli Performance Monitor

In this image, the green line represents the current active web container thread count and the red line is the heap size. The green spikes represent increases in WRS requests. The default refresh rate for this graph is 30 seconds. In addition to viewing this timeline, you can output it to an xml file (click the 'Start Logging' button on this page. When you 'Stop Logging', it creates a zip file that contains the xml file. This xml file is one of the various logs scanned to produce the final SAS analysis data set.

SCALABILITY AND THROUGHPUT

Looking at the entire realm of users (25 to 725), did the tuning iterations maintain scalability? If the number of users is increased, does the response time increase in the same proportion? It is obvious that the first configuration did not scale well after about 250 users. Looking at 3 data points:

Users	Median Response Time
25	31 seconds
400 (16x increase)	1/99 (6.4x increase)
725 (29x increase)	10.68 (34x increase)

At 400 users, which is a 16x increase relative to 25 users, the response time had only a 6.4x increase which indicates good scalability. At 725 users, the response time increased by a factor of 34, which is close to the 29x increase in users. Even at 725 users the scalability is acceptable. If you remove the 60 second think time (60 second pause between reports) and have a zero second think time, you can compute the maximum number of reports per second that is possible.

Users	Reports Per Second
25	15.03 reports/second
50	14.14
75	14.29
100	13.46

The server tier on z/OS with only 3 processors is most likely the limiting factor here.

WEB CONTAINER THREADS

Does the maximum number of web container threads allowed affect the response time? Is more always better? Recall that the current configuration has two application servers: one contains the WRS application and the other contains the SAS Content Server and each has its own thread pool. At 400 and 450 users, run simulation tests for 25 to 175 SASServer1 (the WRS application server) max threads. Below are the results:

Threads	Median Elapsed Time (in seconds)	
	400 users	450 users
25	2.71	5.76
50	2.42	5.21
75	2.02 (26% faster)	4.45
100	2.10	4.26
125	2.53	3.90
150	2.33	3.89
175	2.12	3.87
200	2.15	3.79 (34% faster)

The optimal thread setting for 400 users appears to be 75 and for 450 users it appears to be 200. More threads do not imply faster response time, but rather it is a function of the number of users.

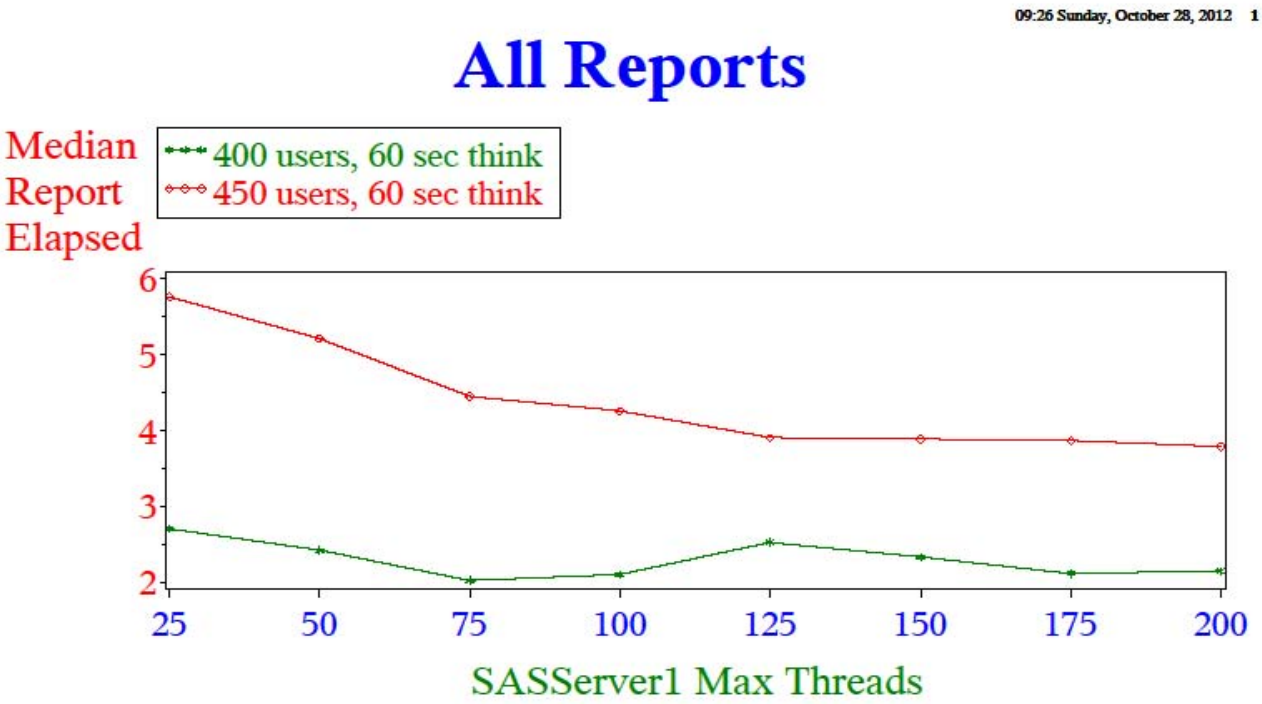


Figure 9. Results of Varying Web Container Thread Settings

DISTRIBUTION OF RESPONSE TIMES

You can create a histogram of response times to get an idea of how the response times are distributed. The image below displays a histogram for 50 users, with 60 second think time. It looks very much like an exponential distribution.

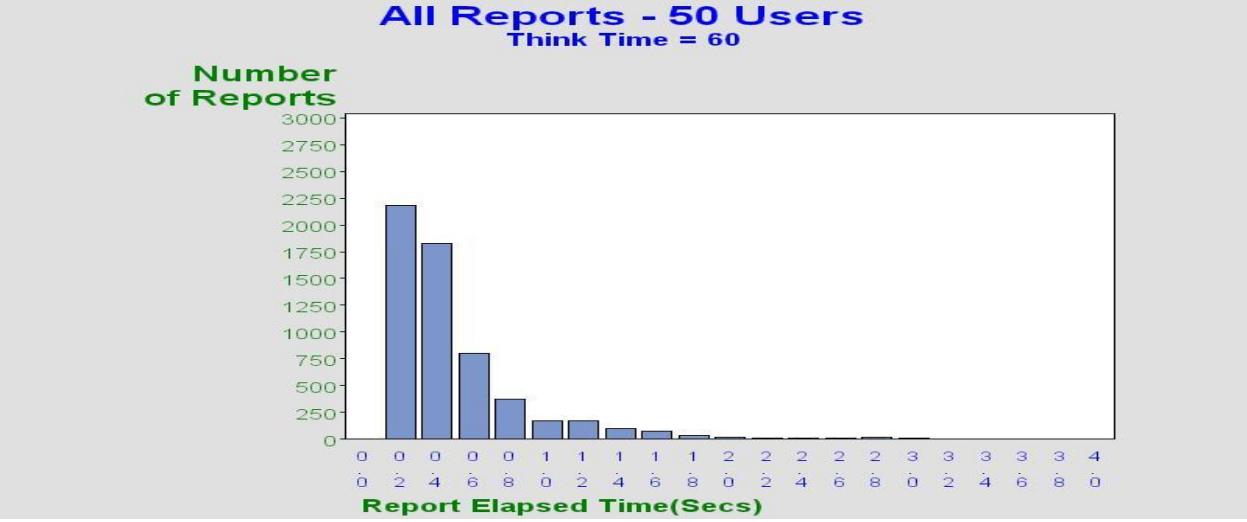


Figure 10. Distribution of Response Times for 50 Users.

If you did the same thing, for 725 users one would expect the same shape of distribution (since it is generating the same SAS Web Report Studio reports) but only the X and Y axis scales would be different. Notice the histogram for 725 users (below). It appears that the distribution of response times has changed with a much heavier load.

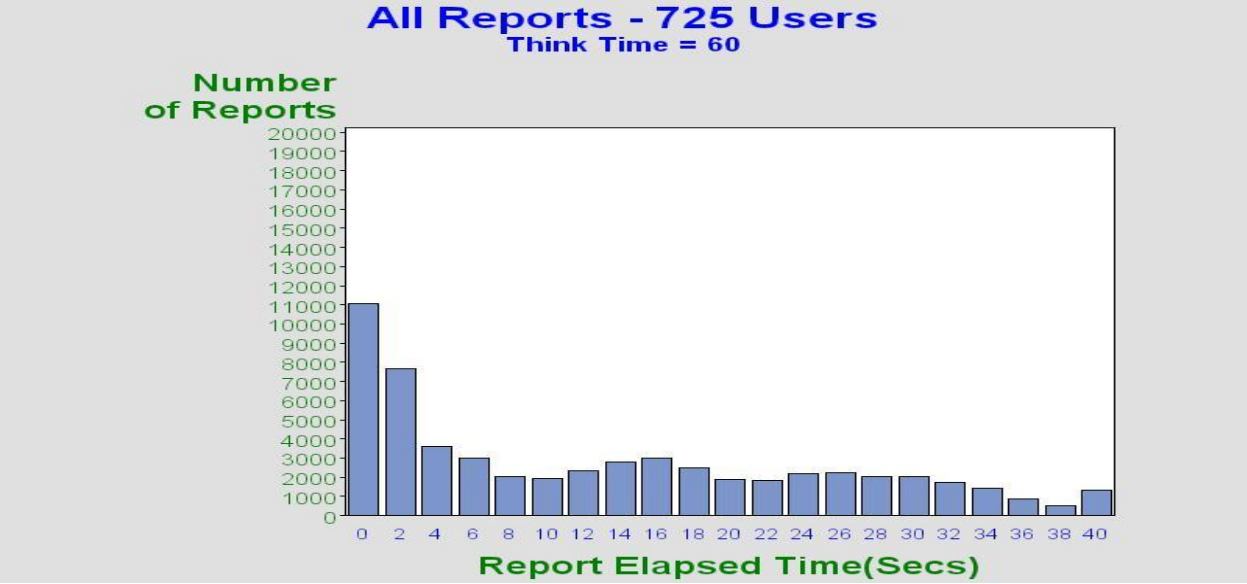


Figure 11. Distribution of Response Times for 725 Users.

EFFECTS OF HEAP SIZE

Does maximum configured heap size have any effect on report response time? Is the effect independent of users? To test this hypothesis, analyze the average report response time for heap sizes of 1.0Gb, 1.5Gb, and 2.0Gb across 25 users and 125 users using Analysis of Variance and the Duncan Multiple Range test. Statistical inference based on hypothesis testing is used to test that the mean response time is the same when comparing different groups (that is, heap size). For example, when comparing just two groups, you can use the t-test to determine if the average response time for a heap size of 1.0 is the same as the average response time for a heap size of 1.5. This is normally written as:

$H_0 : \mu_{1.0} = \mu_{1.5}$ (the null hypothesis)

$H_1 : \mu_{1.0} \neq \mu_{1.5}$ (the alternative hypothesis)

The mean, standard deviation, and n (the number of reports generated) for each of the two groups is used in either accepting or rejecting the null hypothesis. The limitation of this approach is that only two groups can be compared. There are three groups (that is, 1.0Gb, 1.5Gb, and 2.0Gb heap sizes) to compare simultaneously. This is the purpose of the Duncan Multiple Range test, which can be used with SAS PROC GLM (General Linear Model). Now the two hypotheses become:

$H_0 : \mu_{1.0} = \mu_{1.5} = \mu_{2.0}$ (the null hypothesis)

$H_1 : \mu_{1.0} \neq \mu_{1.5} \neq \mu_{2.0}$ (the alternative hypothesis)

Given enough groups, you would almost always reject the null hypothesis because it would take only the mean of one group to be unequal to the mean of any one other group to render H_0 false, hence rejecting H_0 and accepting H_1. The Duncan test gives you the ability to group or cluster the various heap size groups into subsets. Below is the average report time for heap sizes of 1.0, 1.5, and 2.0Gb for 25 and 125 users.

Figure 12. Effects of WAS Heap Size.

The SAS code for the analysis is:

```
proc glm data=heap;by users;
      class heap_max;
      model report_elapsed=heap_max;
      means heap_max / duncan;
run;
quit;
```

The GLM output for 25 users is below.

The GLM Procedure

Dependent Variable: report_elapsed

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	6.32457	3.16228	1.17	0.3102
Error	5529	14933.83981	2.70100		
Corrected Total	5531	14940.16437			
heap_max	2	6.32456906	3.16228453	1.17	0.3102

Output 1. Output From PROC GLM.

Since Pr > F is large (0.3102), you can accept the null hypothesis that the average report response time is statistically the same for heap values of 1.0, 1.5, and 2.0Gb when there are 25 users. Since the response times are equivalent, there is nothing the Duncan output shows that you do not already know. This is intuitive: with only 25 users it does not make much difference how big the heap is, 1 Gb is enough. Now examine the GLM output when users=125

The GLM Procedure

Dependent Variable: report_elapsed

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	50.45082	25.22541	104.18	< .0001
Error	59997	14527.02234	0.24213		
Corrected Total	59999	14577.47315			

heap_max	2	50.45081733	25.22540867	104.18	< .0001

In this case, since Pr > F is <.0001, the null hypothesis is rejected and accept the alternative hypothesis which says the response time is not equal for 1.0, 1.5, and 2.0Gb heap sizes for 125 users. But you do not know, for example if the average report time is equal for 1.0 and 1.5Gb, but not equal for 2.0Gb. Or, all three might not be equal to each other. The Duncan test examines all pairs:

1. Is the report time equal for 1.0Gb heap and 1.5Gb heap?
2. Is the report time equal for 1.0Gb heap and 2.0Gb heap?
3. Is the report time equal for 1.5Gb heap and 2.0Gb heap?

Below is the Duncan output:

```
-------------------------------- users=125 ----------------------
                  Duncan's Multiple Range Test for report_elapsed
NOTE: This test controls the Type I comparison wise error rate,
      not the experiment wise error rate.
    Alpha                            0.05
    Error Degrees of Freedom        59997
    Error Mean Square            0.242129
    Harmonic Mean of Cell Sizes 19999.92

    Number of Means           2           3
    Critical Range       .00964      .01015

Means with the same letter are not significantly different.
Duncan Grouping          Mean      N    heap_max

              A      0.495419  19944    1

              B      0.436828  20028    1.5
              B
              B      0.431275  20028    2
```

The important thing to observe in this GLM output is the 'Duncan Grouping' column. Note that there are two groupings, A and B. The A group is for heap size=1.0Gb. Since both heap size=1.5 and heap size=2.0 are in the B group, they are statistically equal, hence the same grouping. N is the number of reports and the 'Mean' column is the average report elapsed time. The response time shown for a heap size of 1.0 is statistically different (that is, larger) than both 1.5 and 2.0Gb. Furthermore, response times for 1.5 and 2.0Gb are equal. That is more than the standard GLM output, which was only that they are not equal. The Duncan groupings could have ended up with several different possibilities. For example

Heap sizes all unequal:

```
Means with the same letter are not significantly different.

Duncan Grouping          Mean      N    heap_max

              A      0.635419  19944    1

              B      0.536828  20028    1.5

              C      0.431275  20028    2
```

Or, heap size of 1.5Gb is equal to both 1.0Gb and 2.0Gb, but heap size 1.0Gb not equal to 2.0Gb (that is, overlap):

```
Means with the same letter are not significantly different.

Duncan Grouping          Mean      N    heap_max

           B   A     0.440419  19944    1

       C   B         0.436828  20028    1.5

       C             0.421275  20028    2
```

CONCLUSIONS

- SAS Web Report Studio response times were drastically reduced, most notably when the SAS Content Server was deployed to its own WebSphere application server.
- Creating a detailed SAS data set with all the details of each test run enables the analysis to be done with powerful SAS tools such as SAS/STAT and SAS/GRAPH.
- Through the iterative tuning approach, the median response time for 675 users in the final mid-tier configuration is approximately equal to 275 users (about 4 seconds) under the initial, out of the box, configuration.
- Even with only 3 processors, the IBM zEnterprise z196 running the server-tier on z/OS could service up to 725 WRS users.
- Carefully specify the number of application server threads maximum based on the number of users since more threads is not necessarily better. More work is needed on this topic in order to systematically derive the optimum web container thread setting based on the expected number of users and workload characteristics.
- The IBM zBX Blade, containing the SAS EBI Mid Tier and running RHEL 6.2, has proven to be able to handle up to 725 users with plenty of capacity left, integrating well into SAS z/OS components on the server tier.
- WebSphere version 7, running on Linux Red Hat, provides a comprehensive administrator console where changes to the various parameters can easily be performed.
- When compared to previous benchmarks where a single, smaller z/OS image contains both the server and mid-tiers of similar capacity, the zBX with its increased capacity gives superior performance:

REFERENCES

- HP ALM Performance Center version 11.0
- SAS 9.3 Intelligence Platform: Middle-Tier Administration Guide, Third Edition
- WebSphere Admin Console
- SAS/STAT 12.1 User's Guide
- SAS/GRAPH 9.3 User's Guide
- NMON manual http://www.ibm.com/developerworks/wikis/display/WikiPtype/nmon+Manual
- Installation Instructions for SAS 9.3 Electronic Software Delivery for Planning Installations on z/OS

Your comments and questions are valued and encouraged. Contact the author at:

Name:	Fred Forst
Enterprise:	System Z R&D
Address:	10 Hampton Place
City, State ZIP:	Noblesville, IN, 46060
Work Phone:	919-531-4793
E-mail:	fred.forst@sas.com