
Mainframe to AIX: Performance Assurance Case Study

Agenda

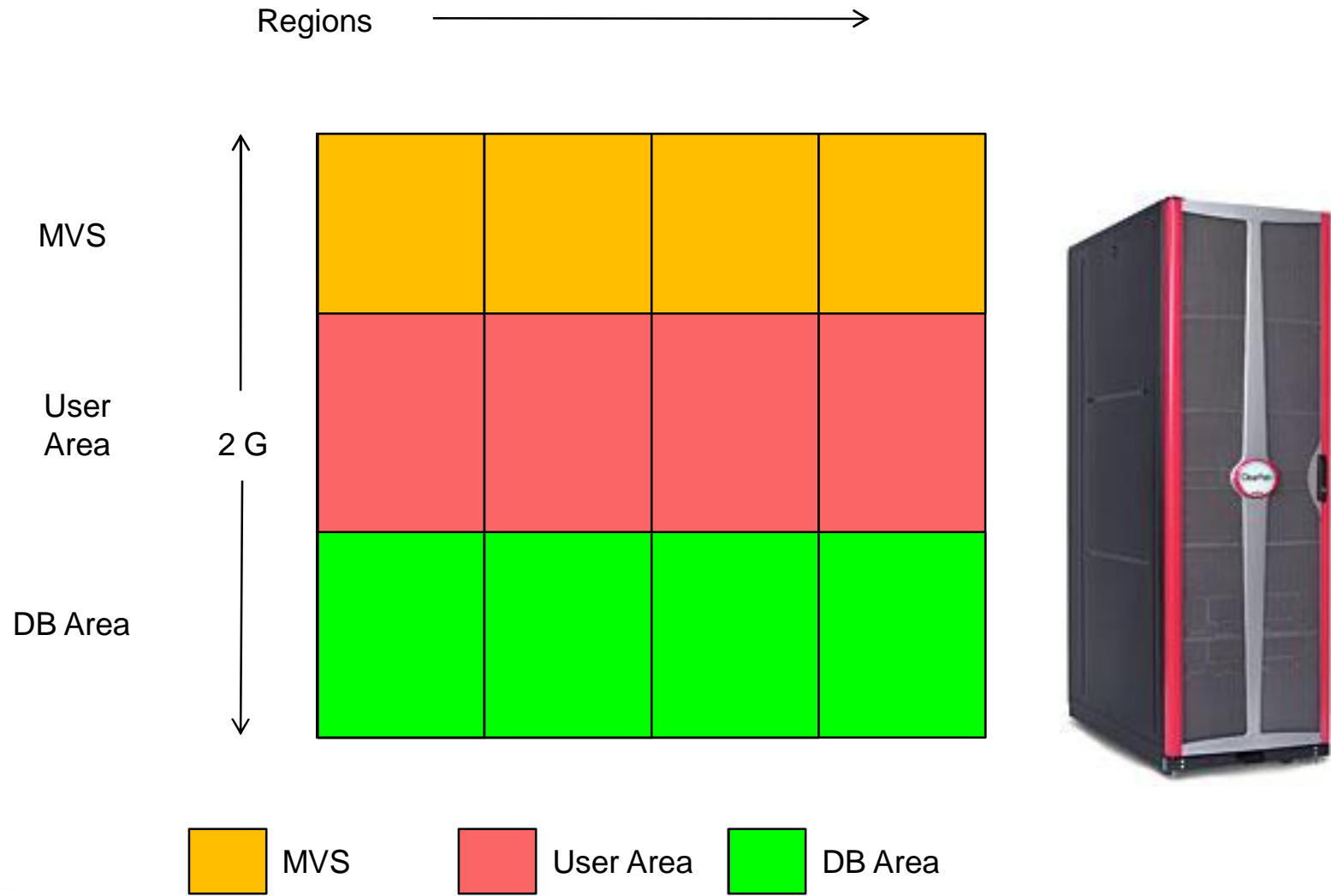
- n Why
- n Hardware Sizing
- n Early Performance Testing
- n Performance Modelling
- n Application Performance Monitoring
- n Performance Testing
- n Go-Live Weekend

Background

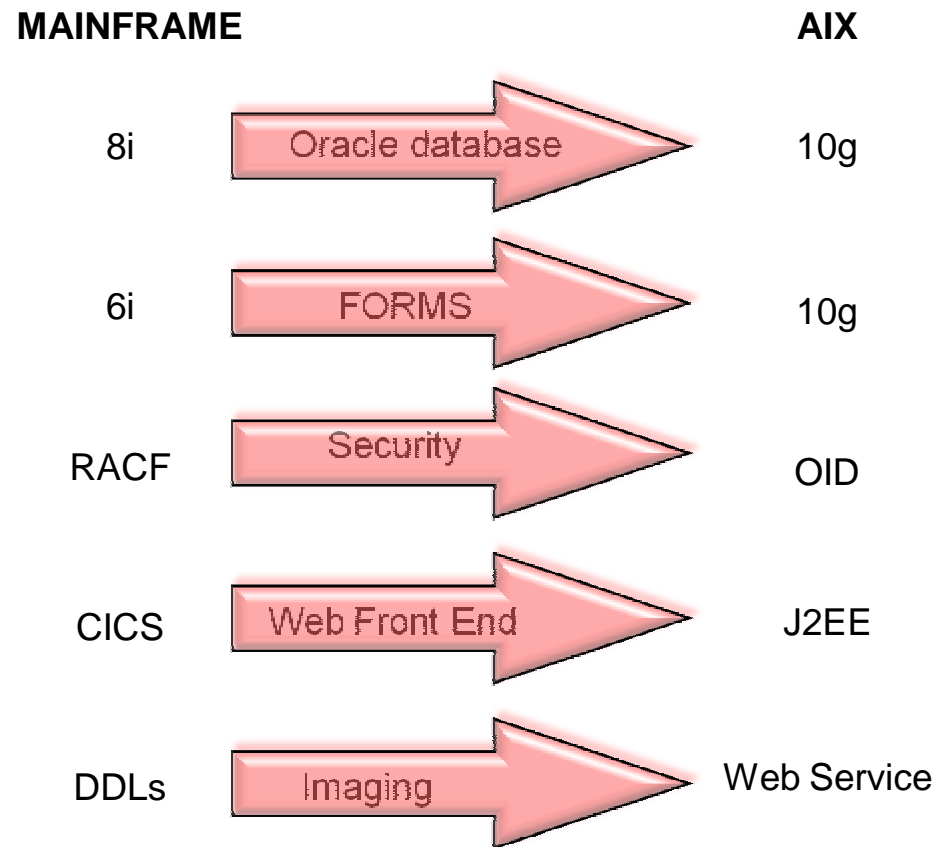
- n Multinational company
 - n 4 major offices within in Europe
 - n Data centre in the UK
 - n 3000 workstation across the globe
- n Core Application hosted on Mainframe
- n Internal Application written Oracle Forms 6
- n External Application web based
- n User access direct or via Citrix
- n Test environments also on Mainframe
- n Scalability of Mainframe limited
- n Continued reports of performance problems
- n No performance engineering capability on site



Scalability on the Mainframe.



Mainframe Platform Migration – Moving to AIX



Architecture

nDB

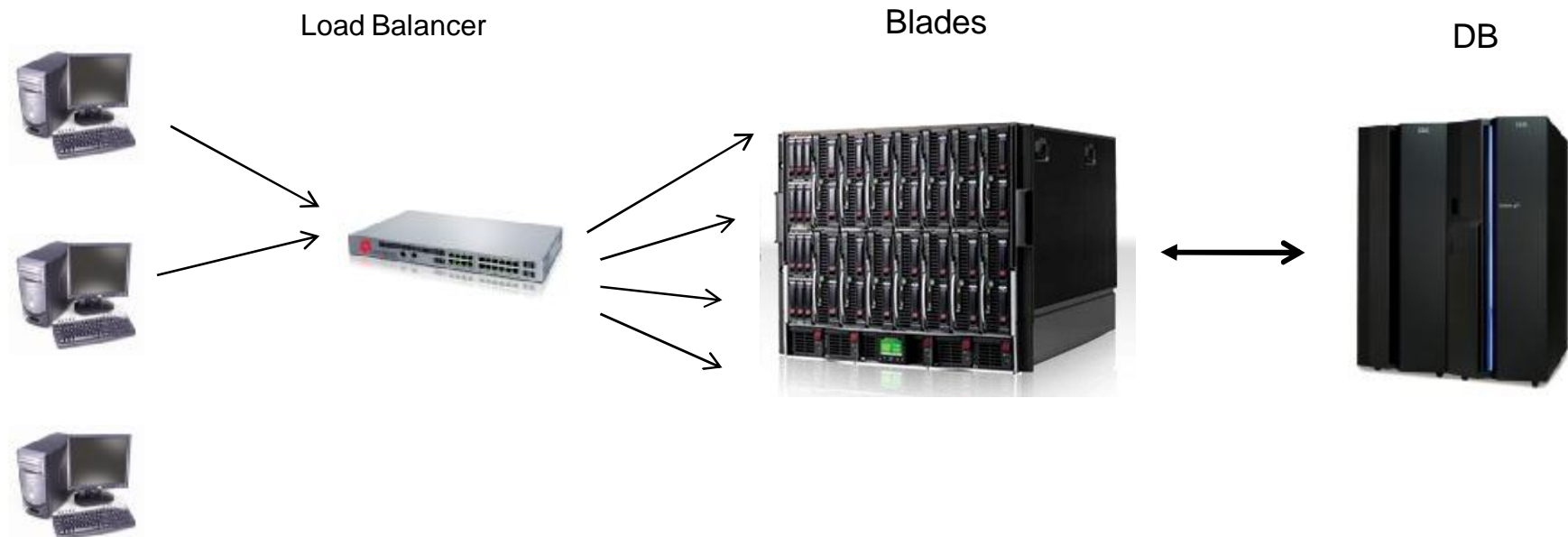
- n AIX P595

nWebforms

- n Linux Blades

n OID

- n LPAR's on P595

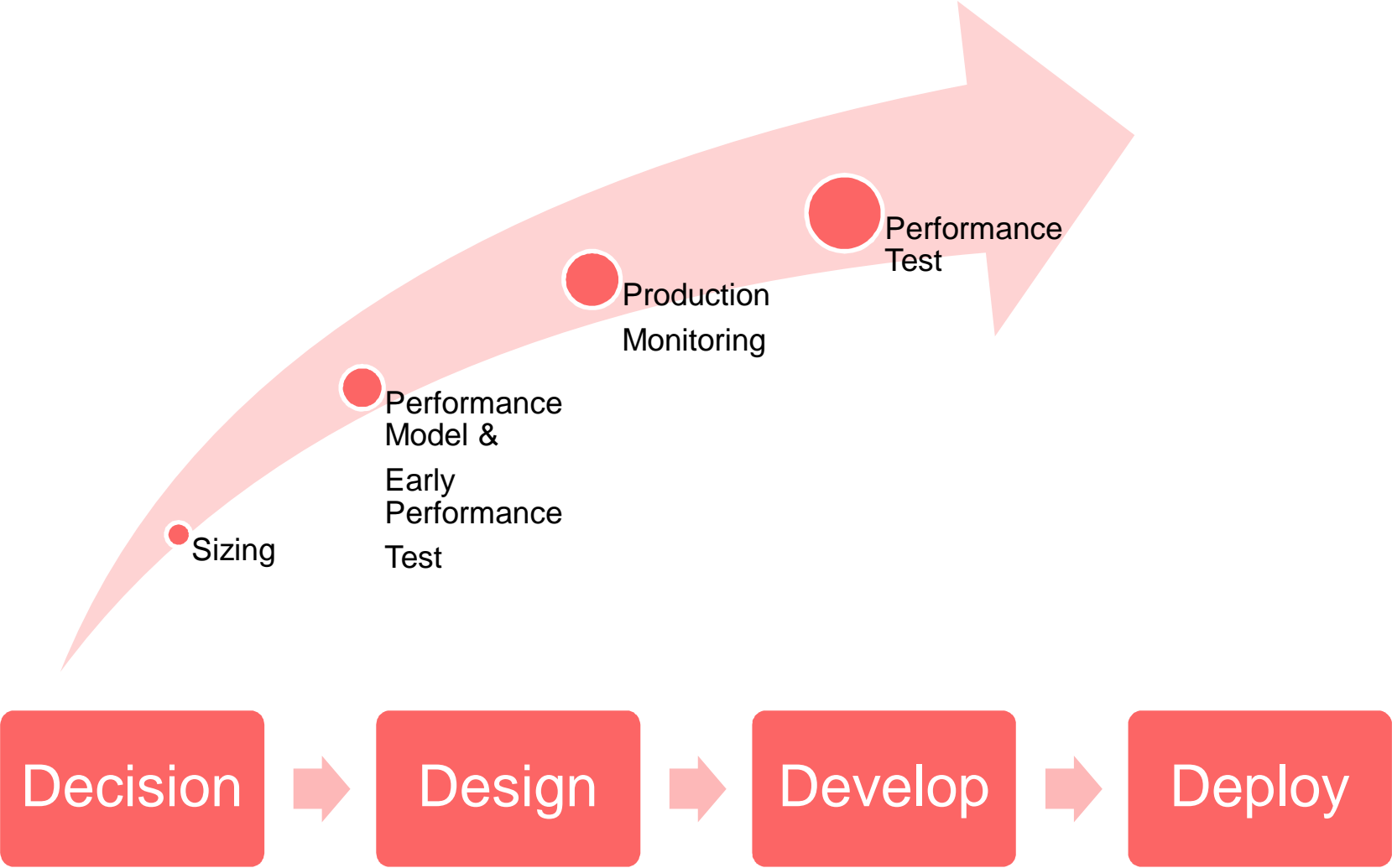


So what about performance

- n Performance requirement for the project
 - n “Performance should be no worse than the current system”
 - n System growth is estimated at 40% per year
- n Test requirements
 - n Performance Testing will be undertaken
 - n 200 User will participate in the User Business Performance Acceptance Test

- n No Volumetrics defined
- n No Scalability metrics defined
- n No response time baseline
- n A final requirements was that Citrix is to be removed

Performance Lifecycle



Early Performance Testing

- n Early Performance Testing is any testing that can be undertaken before the system has been functionally tested and/or performance test environment has been built
- n There are two drivers for Early Performance Testing
 - n What needs to be done based on risk
 - n What is available
- n Advantage that each test can build confidence
- n Beware that people we say the test is not valid

- n Corroborate System Sizing
 - n Can a blade support 200 users?
 - § CPU, Memory
 - n Network sizing?
- n Test Software Components
 - n Name Search
- n Single Thread Testing

Performance Modelling – Finding problems early

- n Small Scale or Development environment can mask problems that will occur in production
 - n Network effects
 - n Cache
 - n Connection pools
- n Small Scale or Development environment can create problems that will not occur in production
 - n Logging
 - n Development code
- n Simple performance models can help “extrapolate” up performance some types are
 - n Transaction path
 - n Resource model
- n More complex models give more information....
- n Example Transaction path model

Wire Shark / Ethereal used for Transaction Path Analysis

The screenshot shows the Ethereal network protocol analyzer interface. The main window displays a list of captured packets with columns for No., Time, Source, Destination, Protocol, and Info. Packet 66 is selected, and its details are shown in the lower pane.

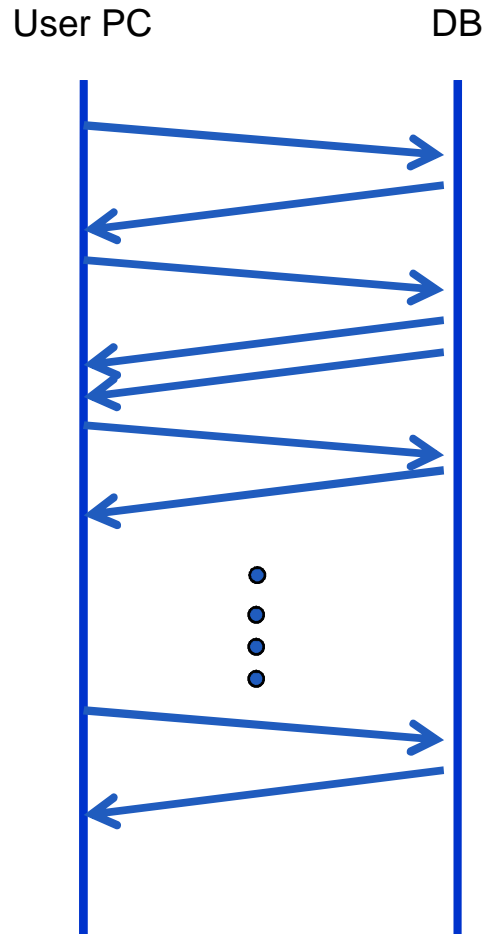
No.	Time	Source	Destination	Protocol	Info
59	21.820594			TCP	2424 > http [FIN, ACK] Seq=487 Ack=229 win=64284 [TCP CHECKSUM
60	21.820717			TCP	http > 2424 [FIN, ACK] Seq=229 Ack=487 win=4866 Len=0
61	21.820728			TCP	2424 > http [ACK] Seq=488 Ack=230 win=64284 [TCP CHECKSUM INCO
62	21.821192			TCP	http > 2424 [FIN, ACK] Seq=229 Ack=488 win=4866 Len=0
63	21.921177			TCP	2425 > http [SYN] Seq=0 Len=0 MSS=1460
64	21.921830			TCP	http > 2425 [SYN, ACK] Seq=0 Ack=1 win=4380 Len=0 MSS=1460
65	21.921848			TCP	2425 > http [ACK] Seq=1 Ack=1 win=64512 [TCP CHECKSUM INCORREC
66	21.922451			HTTP	POST /forms/!serv!et;jsessionid=0ad2232330d7bb3a669d8e90423387
67	22.022821			TCP	http > 2425 [ACK] Seq=1 Ack=472 win=4851 Len=0
68	22.350183			TCP	[TCP segment of a reassembled PDU]
69	22.350293			TCP	[TCP segment of a reassembled PDU]
70	22.350316			TCP	2425 > http [ACK] Seq=472 Ack=2921 win=64512 [TCP CHECKSUM INC
71	22.350636			TCP	[TCP segment of a reassembled PDU]
72	22.350756			HTTP	HTTP/1.1 200 OK (application/octet-stream)
73	22.350768			TCP	http > 2425 [FIN, ACK] Seq=5756 Ack=472 win=4851 Len=0
74	22.350780			TCP	2425 > http [ACK] Seq=472 Ack=5757 win=64512 [TCP CHECKSUM INC
75	22.350837			TCP	2425 > http [FIN, ACK] Seq=472 Ack=5757 win=64512 [TCP CHECKSUM
76	22.351426			TCP	http > 2425 [ACK] Seq=5757 Ack=473 win=4851 Len=0

Frame 66 (525 bytes on wire, 525 bytes captured)
 Ethernet II, Src: ..., Dst: All-HSRP-routers_83
 Internet Protocol, Src: ..., Dst: ...
 Transmission Control Protocol, Src Port: ..., Dst Port: http (80), Seq: 1, Ack: 1, Len: 471
 Hypertext Transfer Protocol
 POST /forms/!serv!et;jsessionid=0ad2232330d7bb3a669d8e9042338783e5df92443cfe.e3aPanyQbxi5e3qoan80anuKb41ynknvrkLo1QzNp651
 Pragma: -91\r\n
 Content-type: application/octet-stream\r\n
 Cache-Control: no-cache\r\n
 User-Agent: Mozilla/4.0 (windows XP 5.1) Java/1.5.0_13-rev\r\n
 Host: symphony-test.atradiusnet.com\r\n
 Accept: text/html, image/gif, image/jpeg, *, q=.2, */*; q=.2\r\n

0000 00 00 0c 07 ac 83 00 18 8b 5e 55 48 08 00 45 00 ^UH..E.
 0010 01 ff fc 0c 40 00 80 06 3a fe 0a d2 83 3b 0a d2@... :...;..
 0020 29 0f 09 79 00 50 f7 12 8a 16 e9 17 ae db 50 18).y.P..P.
 0030 fc 00 c3 df 00 00 50 4f 53 54 20 2f 66 6f 72 6dPO ST /Form
 0040 73 2f 6c 73 65 72 76 6c 65 74 3b 6a 73 65 73 73 s/!serv! et;jsess
 0050 69 6f 6e 69 64 3d 30 61 64 32 32 33 32 33 33 30 ionid=0a d2232330
 0060 64 37 62 62 33 61 36 36 39 64 38 65 39 30 34 32 d7bb3a66 9d8e9042
 0070 33 33 38 37 38 33 65 35 64 66 39 32 34 34 33 63 338783e5 df92443c

File: "C:\DOCUMENT~1\GBALEE1\LOCAL5~1\Temp\etherXXXXH577AU" 15 KB 00:00:25 | P: 94 D: 94 M: 0 Drops: 0

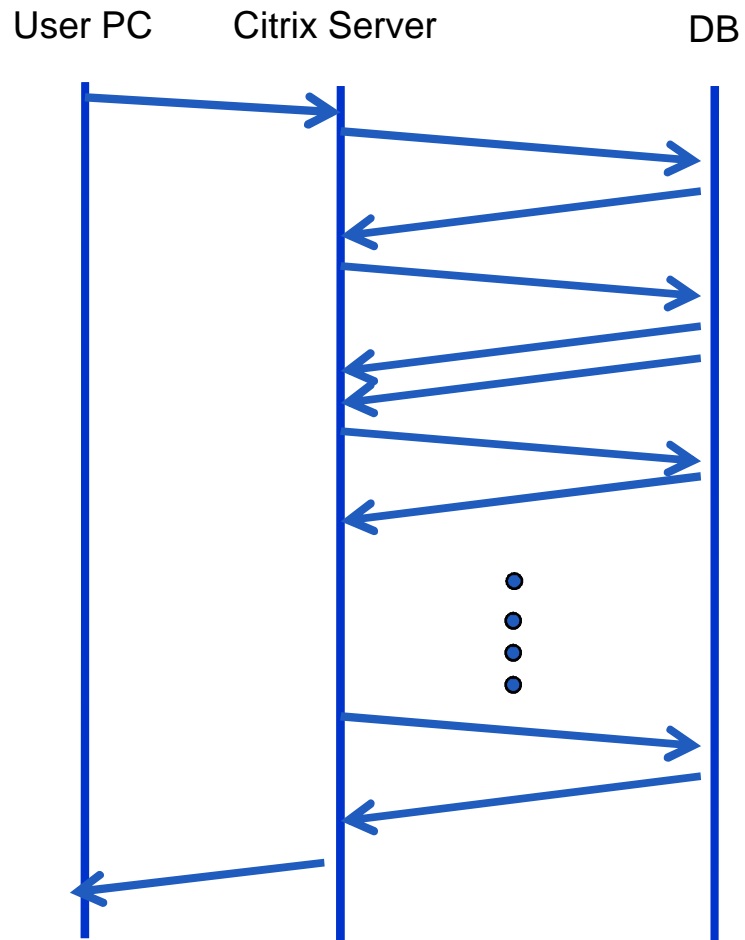
Oracle Form 6i – Execution Path



- Chatty Application
- Typical form takes 20-40 network hops
- Average form size 30 Kbytes

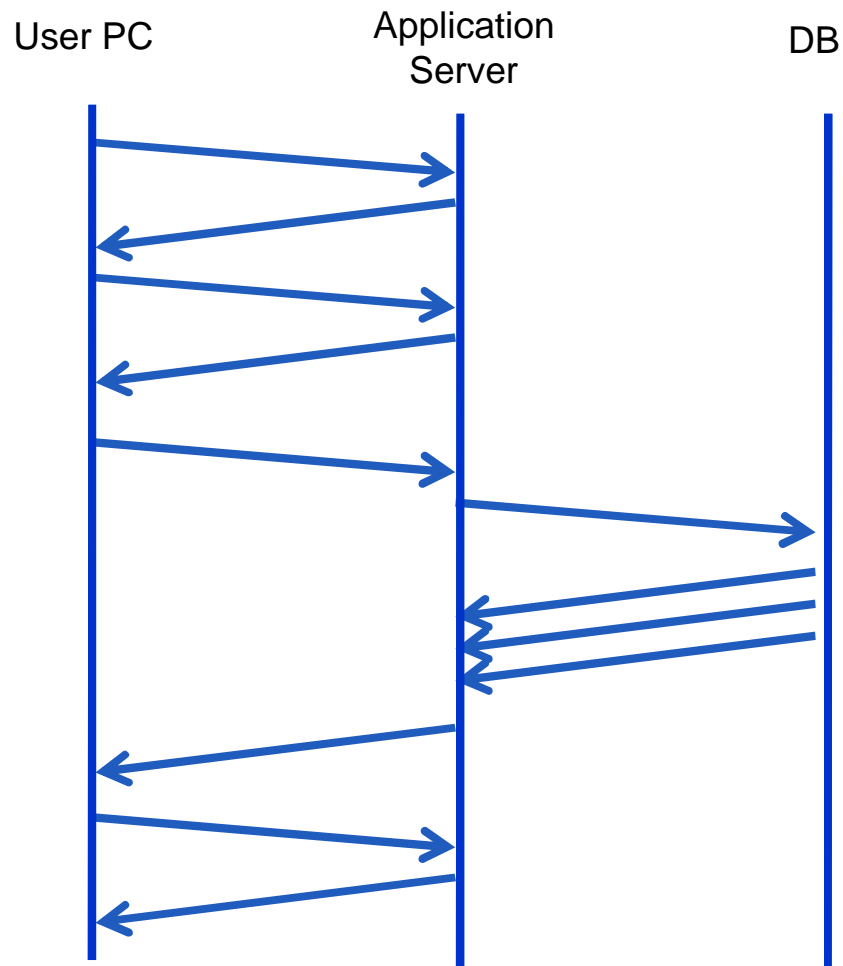
- 20 hops at 1 ms = 20 ms
- 20 hops at 40 ms = 0.8 seconds

Oracle Form 6i over Citrix – Execution Path



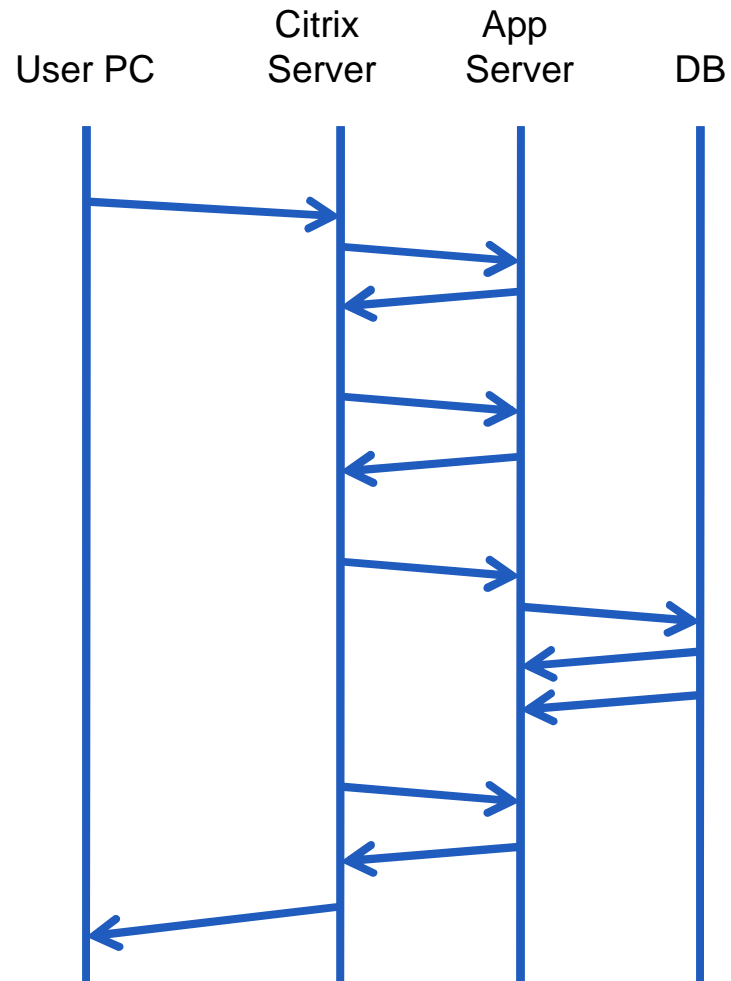
- Typical form takes 1 network hops
- Average form size 33 Kbytes
- 1 hop at 40 ms
= 40 ms + Delta in Citrix Server

Oracle Form 10g



- Typical form takes 4-7 network hops
- Average form size 32 Kbytes
- 4 hops at 1 ms = 4 ms
- 4 hops at 40 ms = 160 ms
- 4 hops at 400 ms = 1.6 seconds

Oracle Form 10g over Citrix



- Typical form takes 1 network hops
- Average form size 35 Kbytes
- 1 hop at 40 ms
= 40 ms + Delta in Citrix Server

Model

$$RespTime_{Native} = 7 \times WanDelay + \left(\frac{MessageSize}{AvailableBandwidth \times \left(\frac{100 - Utilization}{100} \right)} \right) + ServerTime$$

$$RespTime_{Citrix} = 1 \times WanDelay + \left(\frac{MessageSize}{AvailableBandwidth \times \left(\frac{100 - Utilization}{100} \right)} \right) + CitrixServerTime + ServerTime$$

Where does the time go?

On a 1Mbit/sec Line used 40% of the time with a round trip latency of 50mS

Form A

Web Forms



Citrix

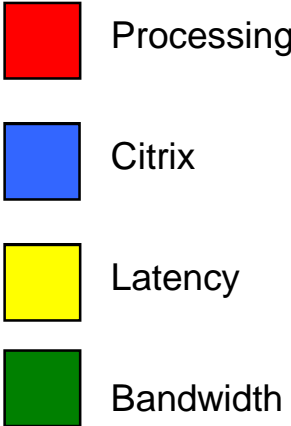


Form B

Web Forms



Citrix



Where does the time go?

On a 100M Line used 40% of the time with a round trip latency of 200mS

Form A

Web Forms



Citrix

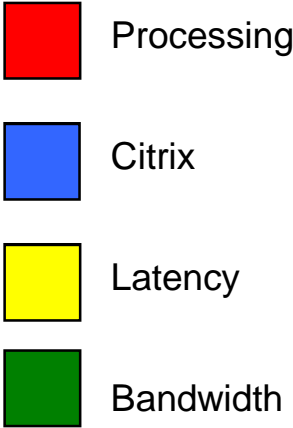
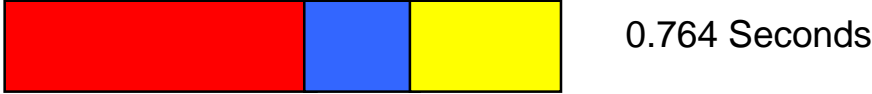


Form B

Web Forms



Citrix



Where does the time go?

On a 1Mbit/sec Line used 40% of the time with a round trip latency of 200mS

Form A

Web Forms



Citrix

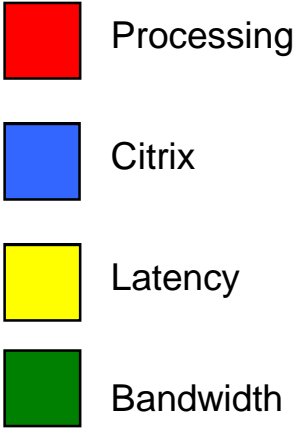
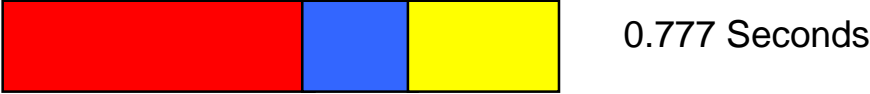


Form B

Web Forms

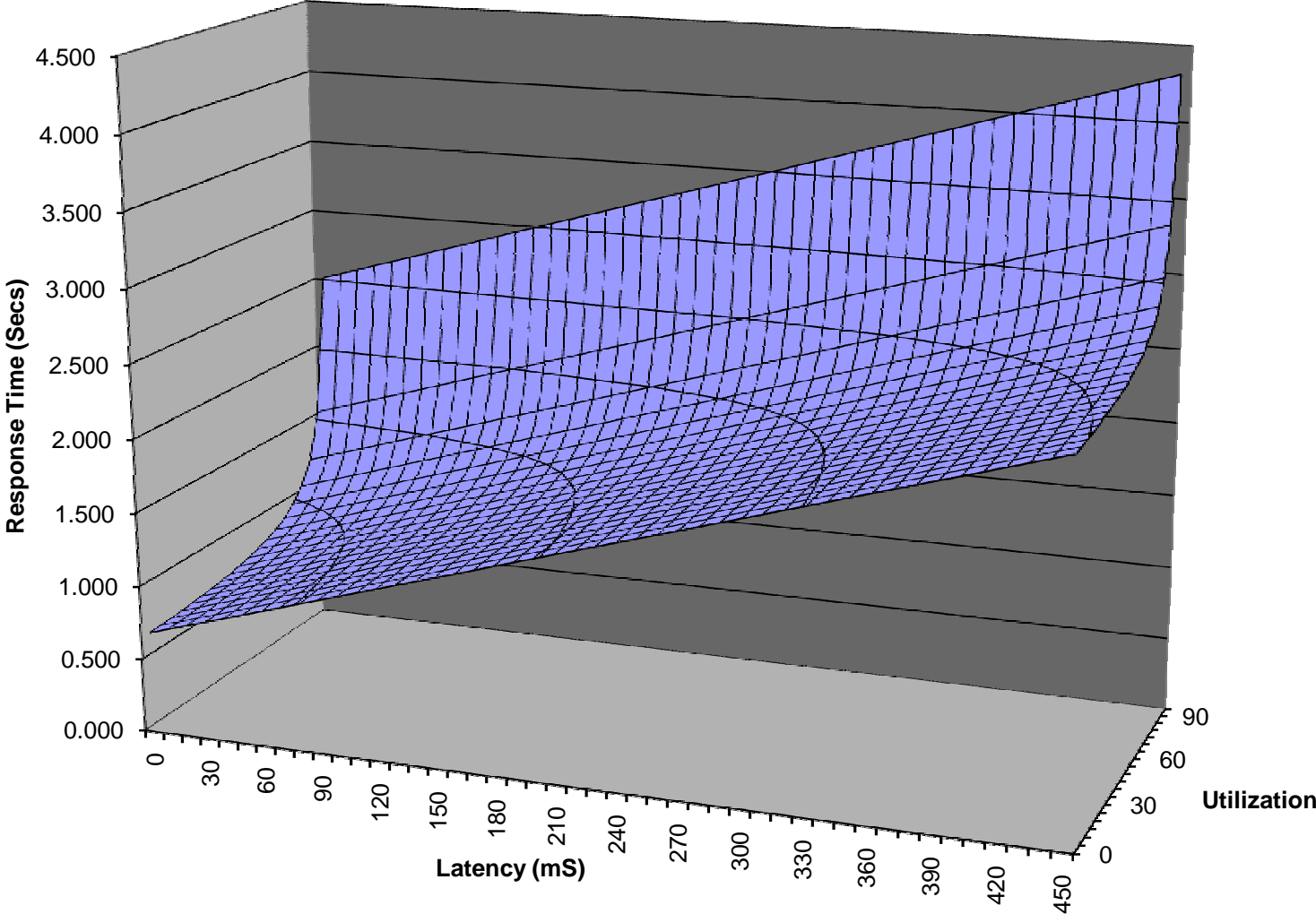


Citrix



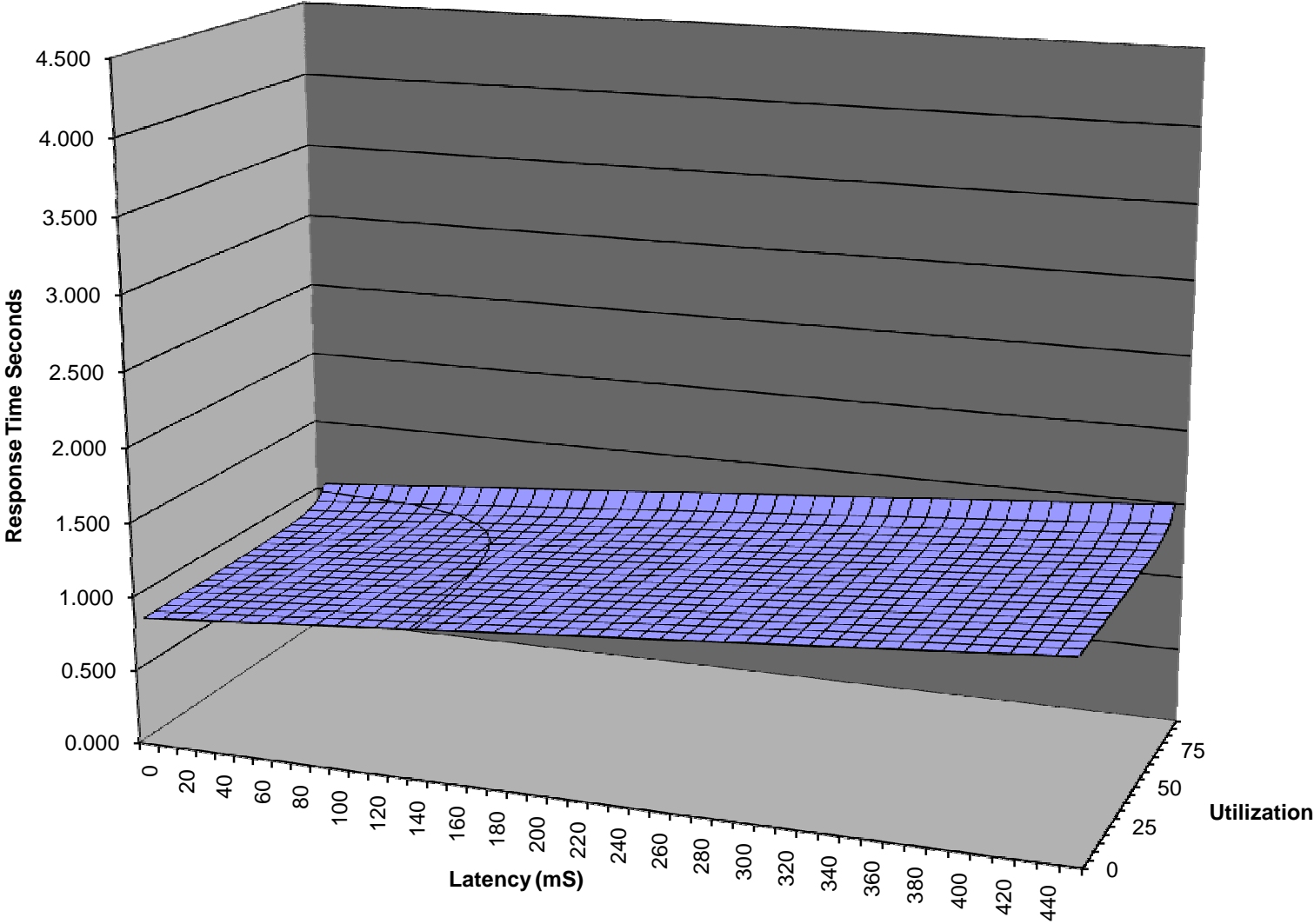
Comparison of Citrix verses Native

Webforms (1 Mbps Connection)

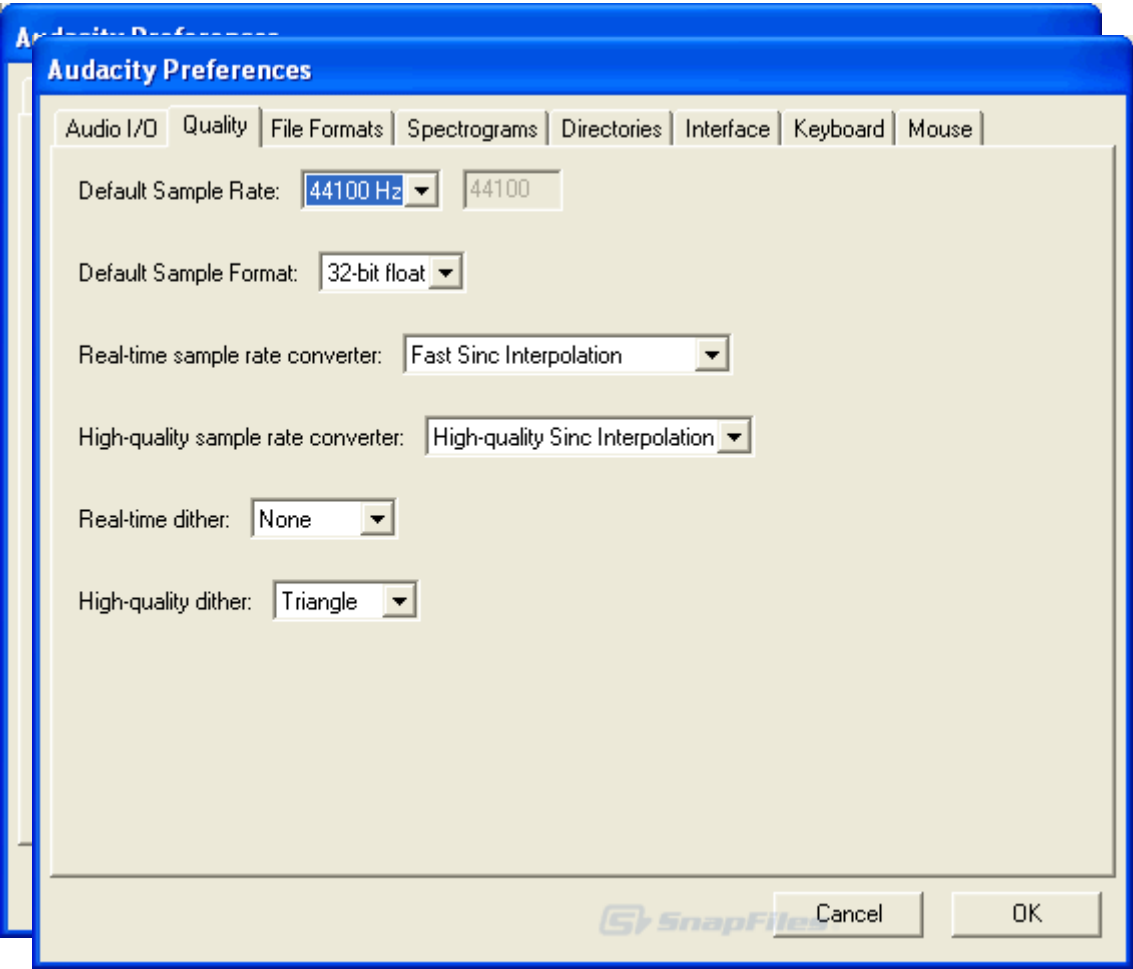
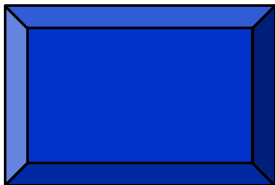


Comparison of Citrix verses Native

Citrix (1 Mbps Connections)



Screen Timings

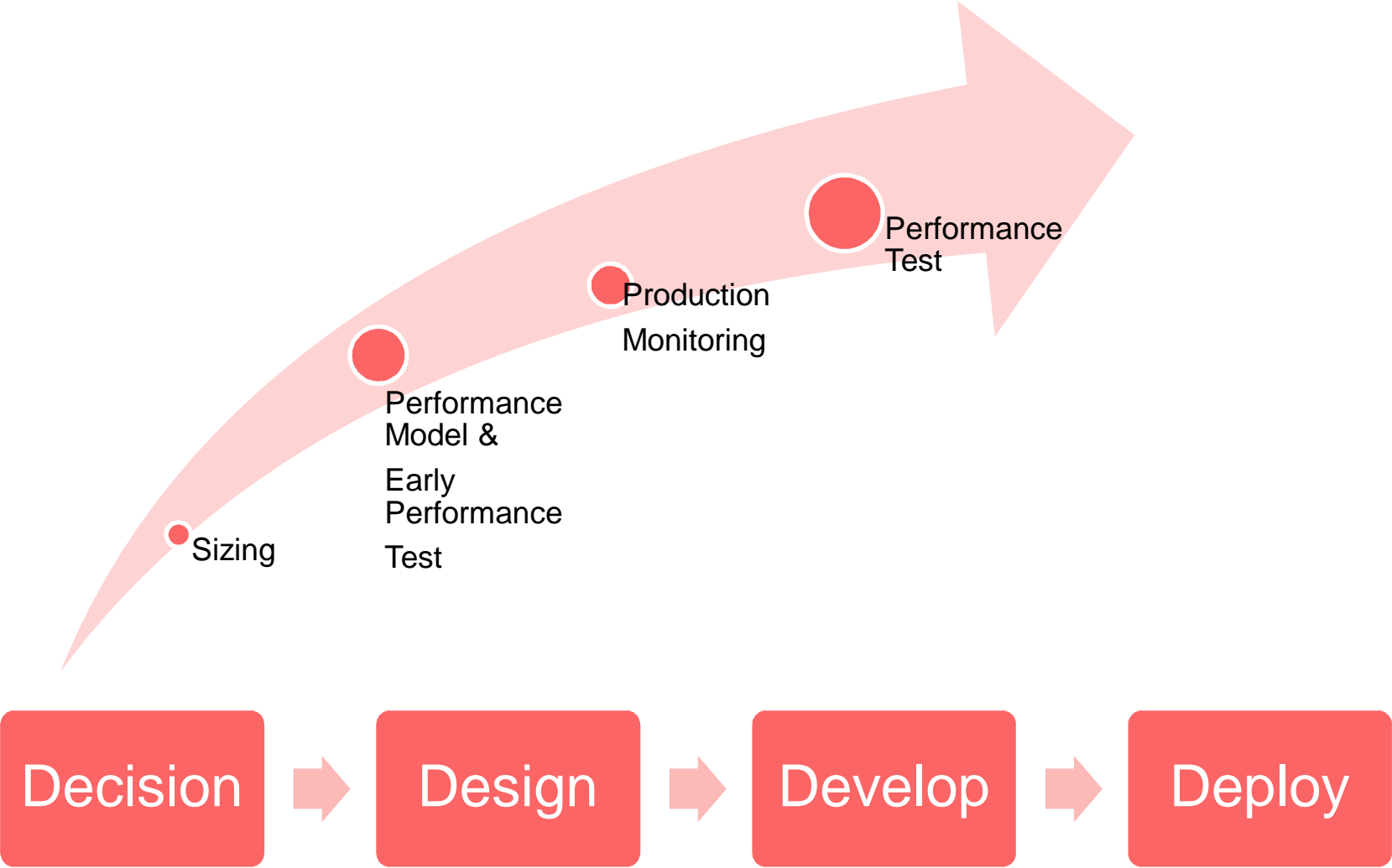


The screenshot shows the Audacity Preferences dialog box, specifically the 'Audio I/O' tab. The dialog has a blue title bar and a light beige background. It contains several settings:

- Default Sample Rate: 44100 Hz (dropdown menu) and 44100 (text field)
- Default Sample Format: 32-bit float (dropdown menu)
- Real-time sample rate converter: Fast Sinc Interpolation (dropdown menu)
- High-quality sample rate converter: High-quality Sinc Interpolation (dropdown menu)
- Real-time dither: None (dropdown menu)
- High-quality dither: Triangle (dropdown menu)

At the bottom right, there are 'Cancel' and 'OK' buttons. A SnapFile logo is visible in the bottom left corner of the dialog.

Performance Lifecycle



Production Response Time Monitoring Requirements

- n The general requirement of production monitoring is to provide an insight into the response time experience by the users. In particular it can highlight trends and discrepancies in response times. There are two main types of approaches
 - n Synthetic Benchmark
 - n User driven
- n The production monitoring had to provide before and after measurements for comparison
- n Implication of these requirement are
 - n Response times must capture the “user experience”
 - n At least 12 key forms are measured. Ideally this will include updates forms.
 - n Timing points are gathered from at least 5 geographic regions (up to 10 is favourable)
 - n Statistically reliable results are needed (high volume)
 - n Need to understand typical response time variances (Hourly, Weekly, Peaks)
 - n Need to monitor up to the go-live date
 - n Accurate measurement is paramount
 - n Measurements mechanism must be applied to all 3 environment (Current, Test and New)

Explanation Measurement Points



User Invokes Form

User PC

System

User Response Time

First Byte to Last Byte

time

PC Processing

PC Processing

Screen is Drawn



Options

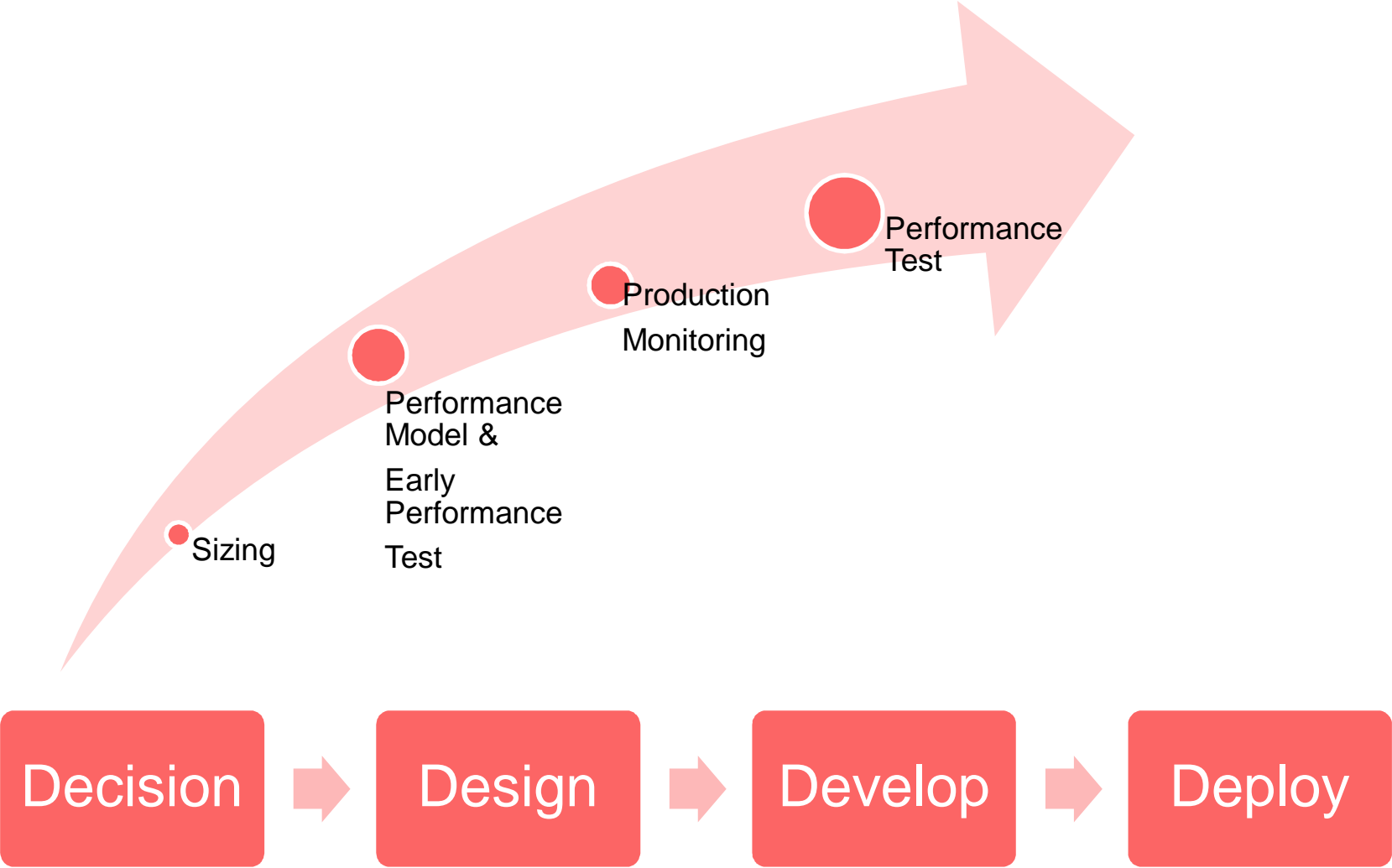
- n We have several options (plus variants)
 - n Protocol based Remote Probes e.g. Mercury Business Availability Monitor
 - § Provided only a first byte out to last byte back measure
 - n GUI based Remote Probes
 - § Provided user experience
 - n Stop Watch timings
 - § Provided user experience but expensive to obtain accurate results
 - n Automated Stop Watch timings in code
 - § Does not work for a citrix environment
 - § Not consistent
 - n Agent Less Solution
 - § First byte out to last byte our measure
 - § Not a consistent measure as users do not repeat exactly what they do each day



Tivoli

- n Pros
 - n Currently Procured
 - n Repeatable measure
- n Cons
 - n Out of the box Accuracy
 - n No data archiving
 - n Overhead of maintaining probe
 - n Limited diagnostics of probe

Performance Lifecycle



Performance Testing

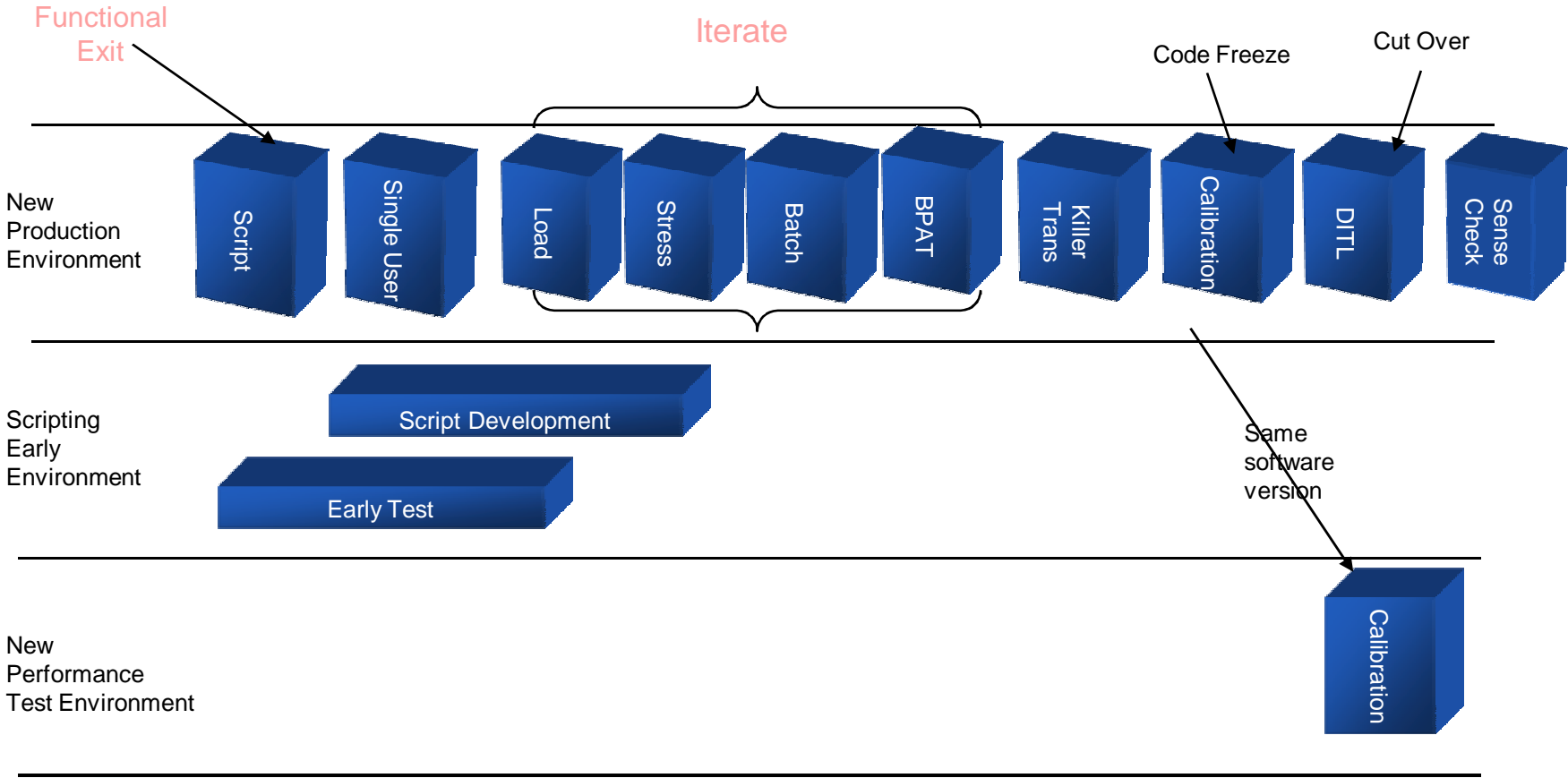
- n First job produce the test strategy/plan
 - n Refine Performance Objectives
 - n Scope
 - n Create a baseline
 - n Testing Approach
 - n Environments
 - n Required Resources
 - § Tools
 - § Staff
 - n Roles and Responsibilities
 - n Assumptions
 - n Risks
 - n Define success and who declares success

Test Approach

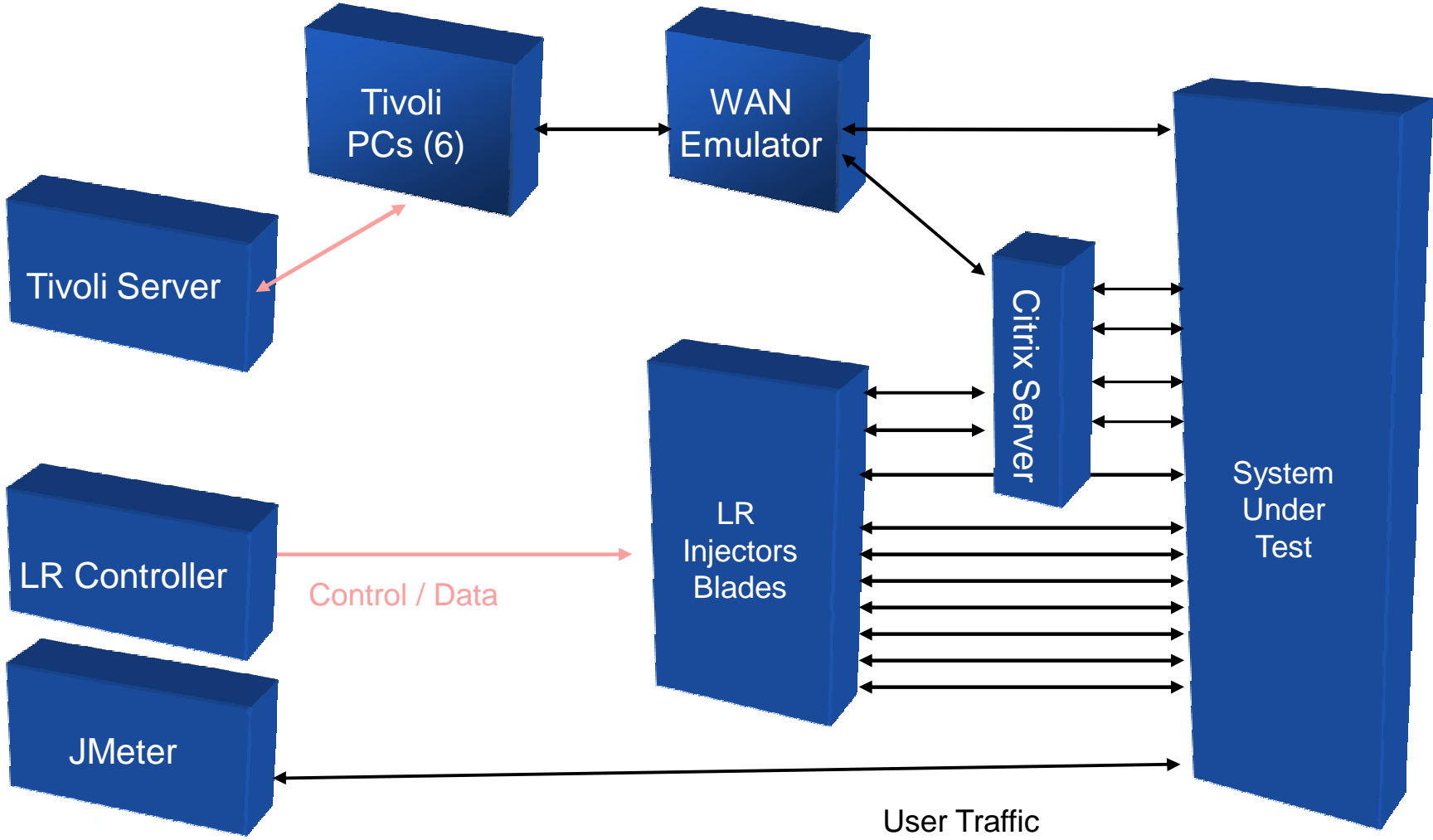
n Different Types of Tests

- § Load
- § Citrix Benchmark Test
- § Stress/Scalability
- § Batch
- § Day In The Life
- § Business Performance Acceptance Test
- § Killer transactions
- § PreProduction Sense Check
- § Calibration

Testing Approach Time Line



Test Environment



Tools Used

- n Network Sniffer
 - n Wireshark (ethereal)
- n Load Testing Tools
 - n LoadRunner
 - n Jmeter
- n Performance Monitoring
 - n Oracle OEM
 - n Native AIX measures
- n Production Monitoring
 - n Tivoli ITCAM RTT
- n WAN Emulation
 - n iTrinegy
- n User Response Time
 - n Tikker

Business Performance Acceptance Performance Test

- n Background
 - n The business case outlined that there will be 4 user test events
- n Advantages
 - n Allows users to compare the performance of Test to Production
 - n Checks functional integrity at full load
 - n Provides confidence to the business
 - n Increases test coverage
- n The Challenge is getting reliable representative results from the business.
 - n Option 1 Users choose what to test
 - n Option 2 User execute scripts created by QA
 - n Option 1 was chosen with strict guidelines

BPAT Suggested Guidelines

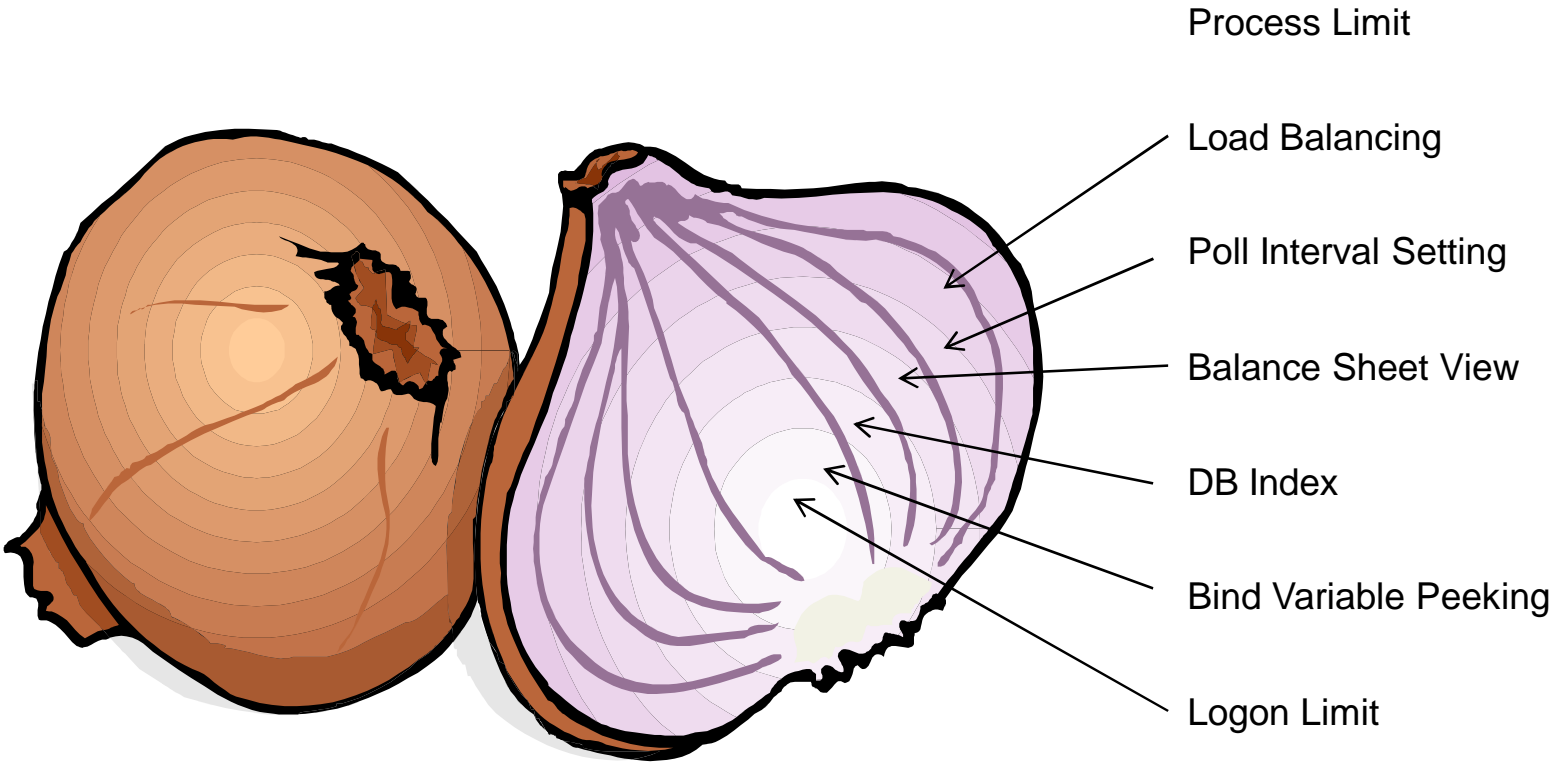
- n Choose common/critical operations
 - § There is little benefit in choosing an infrequently used long operation
- n Repeat measurement in Production and Test at least 3 times
- n Take the Production measurements at the busiest times
- n Use identical or similar data
 - § Note Test will have a data position in the past
- n Results will be entered into a pro forma
 - § Operation / Form Name
 - § Time of Measurement
 - § Start Point for operation
 - § End Point for operation
 - § Data used (policy, customer numbers etc)
 - § Duration
- n User will be told they need to follow guidelines or results will be discounted
- n Measure only the time of the computer not themselves
 - § Click to screen delivery
- n User ID will be production ?
- n Follow normal business behaviour – if you open multiple windows do the same
- n Do not run Production and Test together.

BAT Lessons Learned

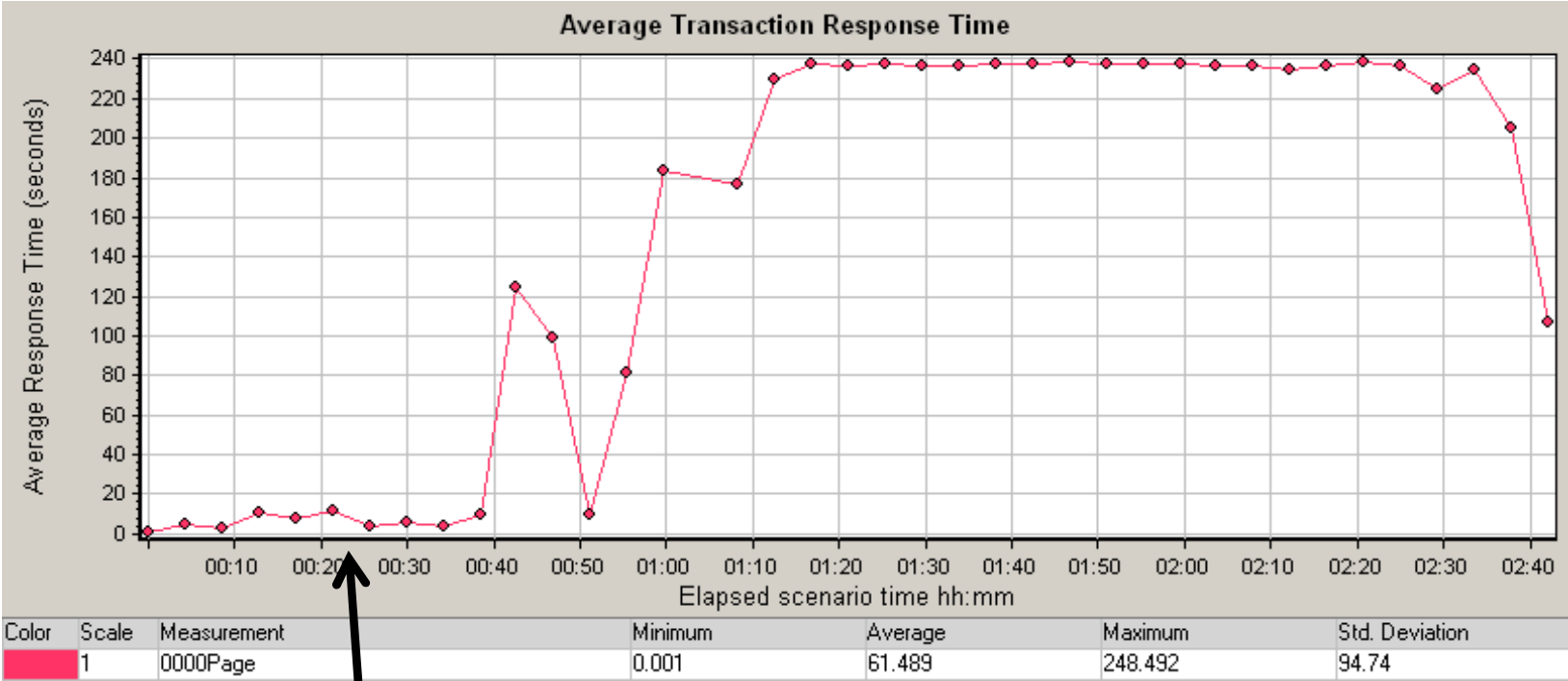
- n Organising 200 users on a new system with new processes is very time consuming
- n Users do not read the instructions
- n Use an excel spreadsheet response form
- n Specify time format
- n Give them the tools to do the measurements
 - n Tikker
 - n (<http://www.1202performance.com/tools/tikker-the-performance-engineers-stopwatch/>)
- n Where the GUI changes even slightly then perception rules!



Performance Results – The Performance Onion



Bind Variable Peeking Results



All Users Logged In

Bind Variable Peeking

What are Bind Variables

Each time an SQL statement is sent to the database, an exact text match is performed to see if the statement is already present in the shared pool. If no matching statement is found a hard parse is performed, which is a resource intensive process. If the statement is found in the shared pool this step is not necessary and a soft parse is performed. Concatenating variable values into an SQL statement makes the statement unique, forcing a hard parse. By contrast, using bind variables allow reuse of statements as the text of the statement remains the same. Only the value of the bind variable changes.

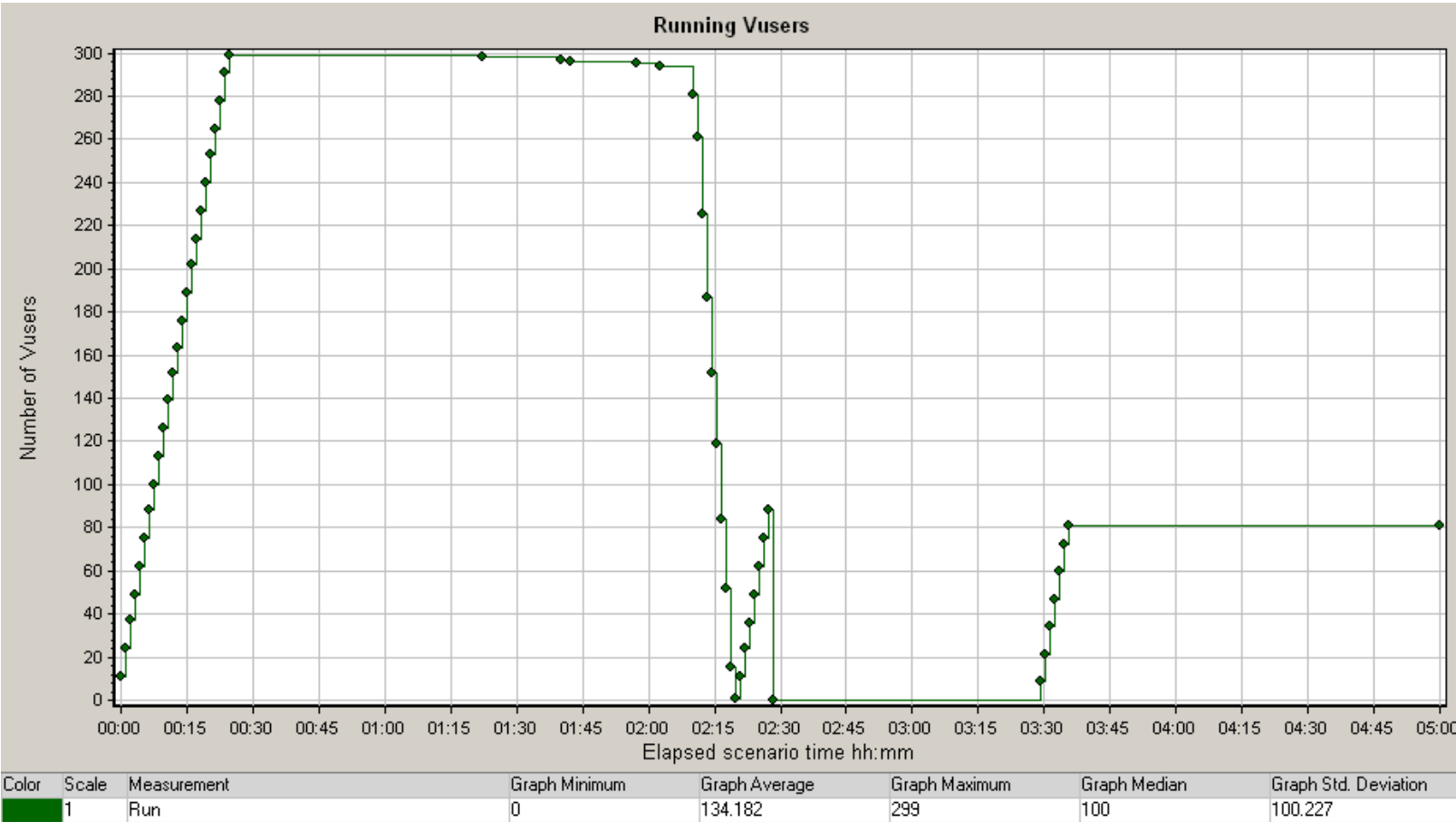
What is Bind Variable Peeking

This allows the optimizer to look inside a bind variable when a statement is hard parsed (i.e. initially when it is loaded into the shared pool). This sets the execution path for the next execution of that statement. This path may not be optimum next time the statement is executed. This is particular when tables have data distribution that are heavily skewed

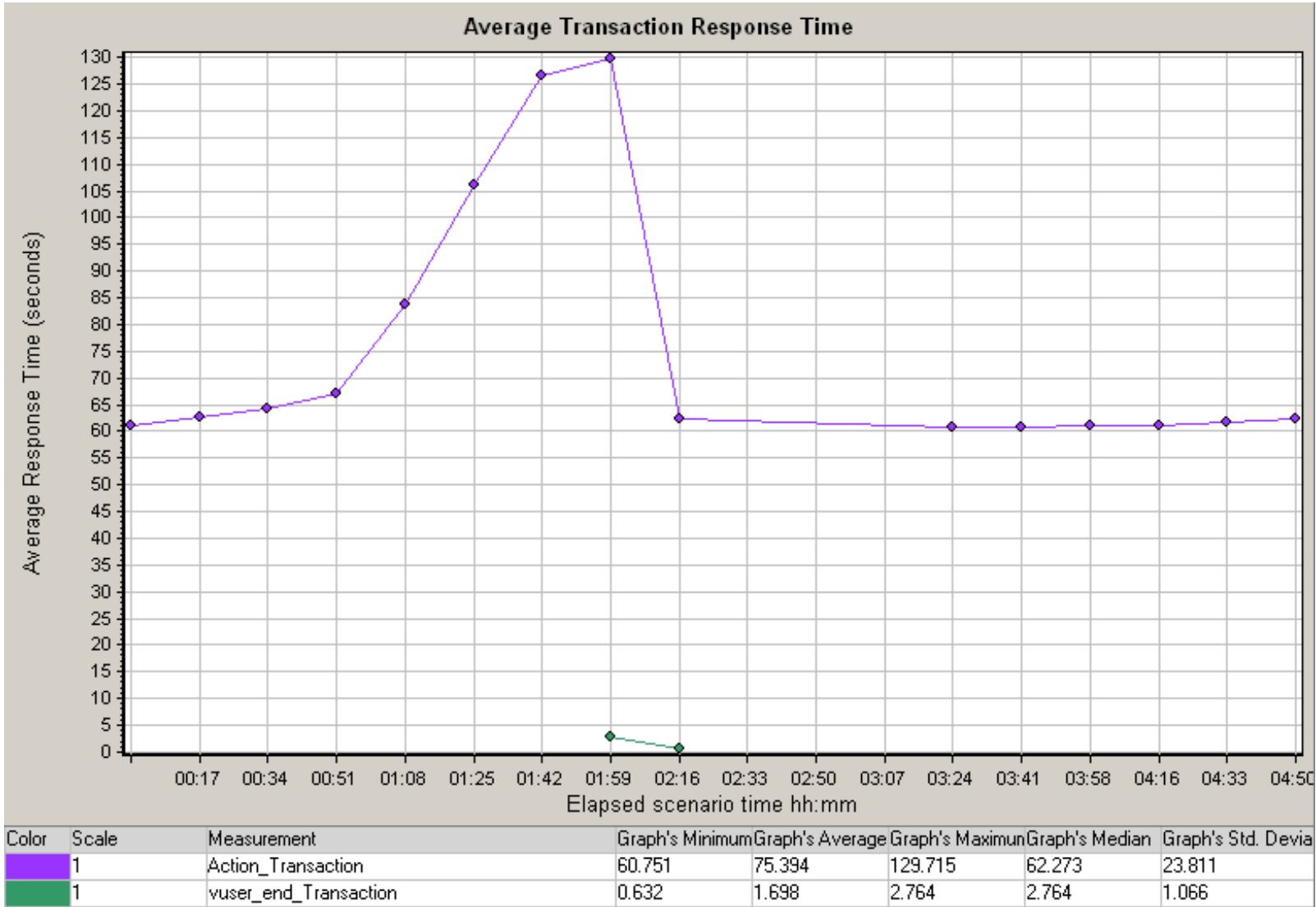
Solution

Add a hint to the SQL statement

Logon Limit



Logon Limit



System Stability

- n Not an easy problem solve
- n No hardware bottleneck was detected
- n Configuration of Oracle Application Server was as per installation guide
- n Some improvement could be made by turning keep alive off
 - n Keep alive off increased response times by 190 mS
 - n Improved the user capacity of the server by 25%

- n Problem eventually found to be Linux ulimit
- n Ulimit effects the number of open files/processes on linux

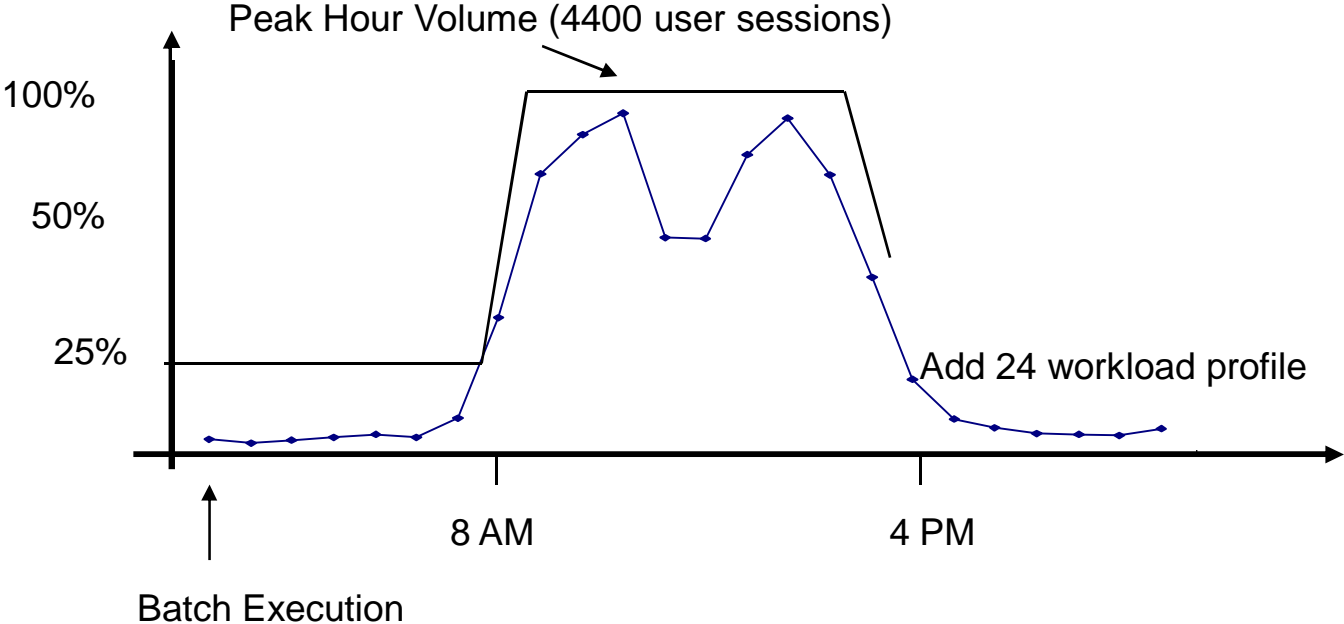
Performance Testing Results

A Quick Summary of the Performance Tests

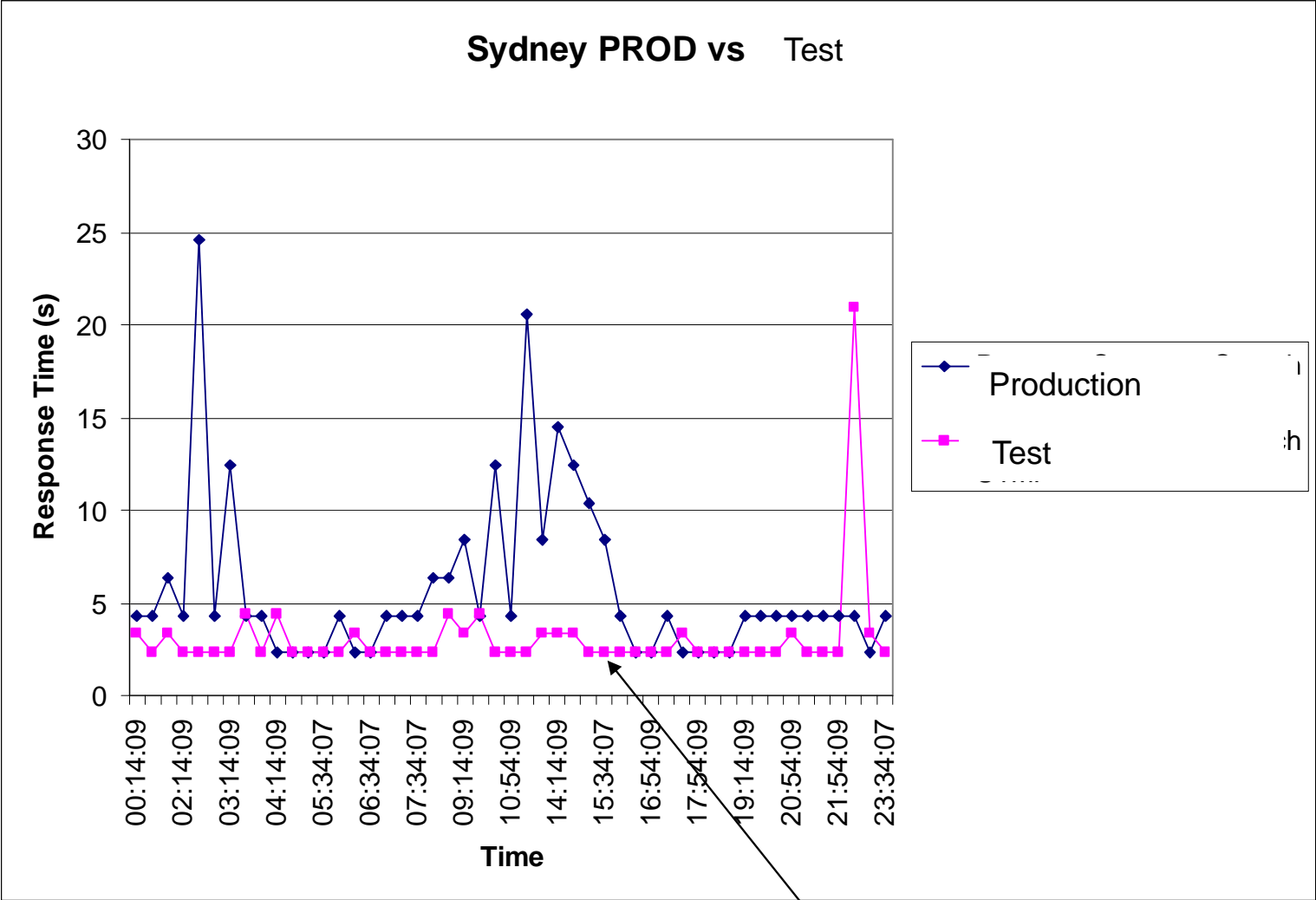
- n Peak Hour
 - n Repeated many times and used to tune the system
 - n CPU, Disk and Memory utilization is low
- n Batch
 - n Execution time estimated to be 50% faster than current production
 - n No noticeable degradation in user response time when batch runs
- n Killer Transactions
 - n Do difficult transactions crash the system?
 - § No
 - § At most CPU jumped from 40% to 50% but no difference in response times
- n Scalability
 - § System can support a 50% (over a years) growth in business volumes
- n Day in the Life / 200 User PAT Test
 - n Aggressive 24 hour test
 - n Performance is compared to production using Tivoli monitor
 - n PAT User compare response times between TEST & PROD

Day in the Life Workload

- nLoad Runner Simulates a 24 hours load
- nPeak load volume is for +6 months from go-live date
- nExcessive number of Name Searches



Response Time Monitors (Tivoli)



Note no response time degradation under load

Pre Go-Live Risks

- n Bind Variable Peeking
 - n Oracle attempts to select the best path through the database for a query
 - n First time the DB peeks at the variables in the SQL query
 - n Next execution follows the same path but that path may not be optimal
 - n Mitigated by DBAs

- n Larger than expected impact of the true production environment
 - n Test DBs are replaced by production DB
 - § Mitigated by JMeter testing of Image Subsystem in Prod
 - n Post PAT Test Code Changes could effect performance

- n Test Scenario not realistic
 - n Mitigated in part by user Business users
 - n Analysis of DB stats between TEST and PROD
 - n Addition of headroom into the test scenario



Go-Live Sunday 10 PM

- n Sunday 9 AM PreProduction Sense check deviation ??????
- n System Health check complete 8,30 PM
- n Asia users use the system from 10PM on Sunday nights
- n All is well at 8 AM on Monday Morning
- n At 8.25 AM excessive I/O to disc thought to be caused by Bind Variable Peeking
 - n Poor User Experience for the rest of the day
 - n Hints added to DB to reduce the excessive I/O at night
 - n Poor I/O performance identified

- n Tuesday at 8.30 AM excessive I/O to disc thought to be caused by BVP
 - n Poor User Experience for the rest of the day
 - n Hints added to the DB to reduce excessive I/O
- n Wednesday at 8.35 AM excessive I/O to disc
 - n Poor User Experience for the rest of the day
 - n More memory and disc redistributed at night
- n Thursday
 - n Fine...