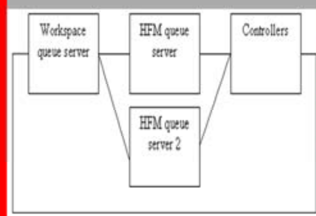


Multi-tiered applications sizing methodology based on load testing and queuing network models

Leonig Grinshpan, PhD
Consulting member of Technical Staff
Oracle Performance Engineering
July 2008

Multi-tiered enterprise applications (MTA) feature complex architecture with server farms on web, application, and database layers. Permanent growth of a number of users, volume of operational and financial data, as well as complexity of business transactions requires MTA customers periodically proactively estimate capacity of their installations it terms of a number of servers, CPU’s per server, speed of CPU, IO, and network, as well as an impact of capacity on transaction response time.

The paper presents MTA sizing methodology employed by Oracle’s Hyperion performance engineering group for enterprise performance management application. The methodology uses both load testing and queuing network modeling tools. Load generation software emulates workload and collects data to feed queuing network models of MTA. After calibration models generate estimates of transaction response times and server utilizations for different what-if sizing scenarios (number of servers, number of CPUs per server, CPU speed, number of concurrent users etc).

Presented approach provides more accurate sizing estimates and recommendations than empirical methods.

Agenda

- **Presentation goal**
- **Queuing network as a model of computer system**
- **What is application sizing?**
- **Methodology of application sizing**
- **Example 1. Model of production system**
  - Building model
  - Calibrating model
  - What-if scenarios

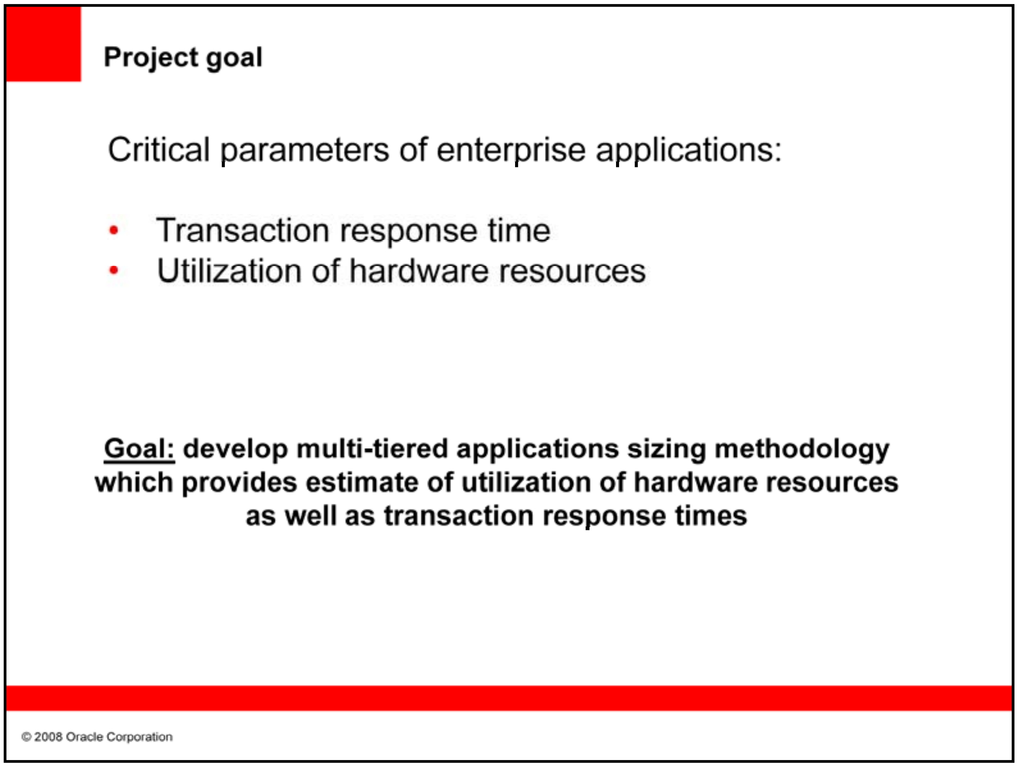
Multi-tiered enterprise applications (MTA) have common characteristics which is essential from a performance engineering perspective:

➢Having significantly fewer users than Internet applications because their user communities are limited to corporation business departments. That number still can be pretty large reaching thousands of users, but it is never even close to millions.

➢End user works with MTA not only through browser as in case of Internet application, but also through multiple Windows front-end programs like Excel, Power Point, as well as programs specifically designed for different business tasks user interfaces. Pretty often a front-end program does significant processing of information delivered from servers before making it available to a user.

➢MTA are always evolving because they have to stay in sync with ever changing demands from business they support. Businesses fluctuate going through economic cycles with prevailing trend directed toward business growth. That generates a permanent need for MTA performance tuning and sizing due to changes in a number of users, volume of data, and complexity of business transactions.

➢Processing much larger volume of data per a user request than Internet applications because they sift through terabytes of business records and often implement massive on-line analytical processing in order to deliver business data rendered as reports, tables, sophisticated forms and templates.



Presented how a sizing methodology differs from capacity planning. The term “capacity planning” means “resource planning”; sizing methodology provides estimates of resources as well as transaction times.

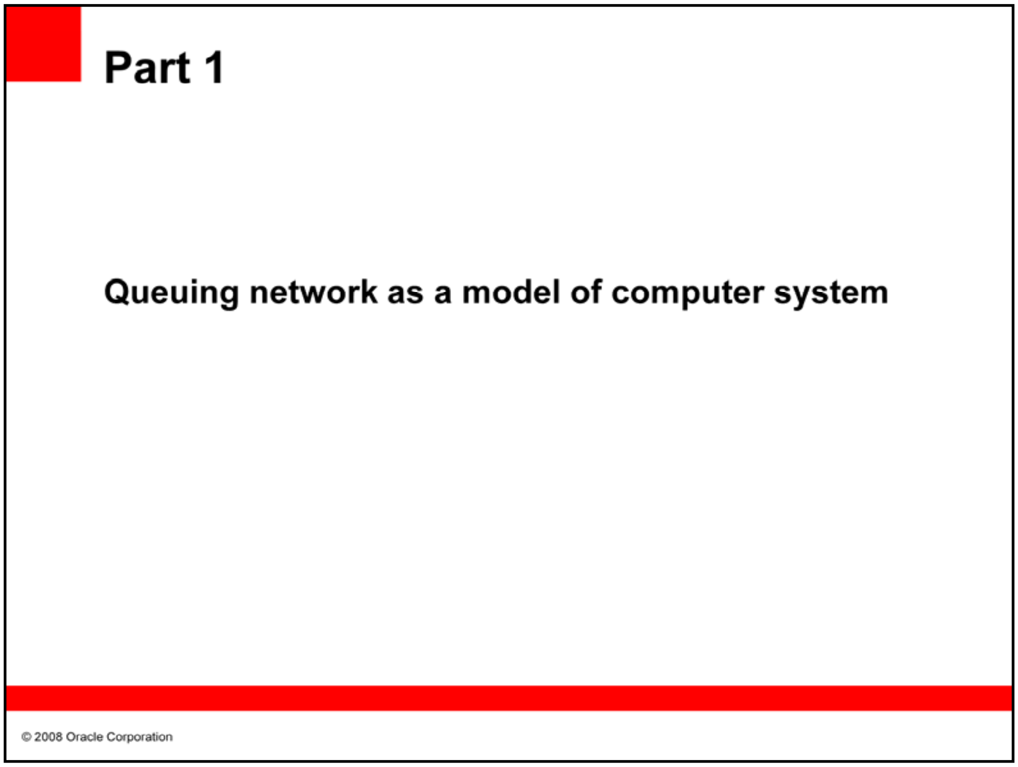
Transaction response time – main concern of user

Utilization of hardware - main concern of IT departments; it is hot parameter today with the onset of green datacenters

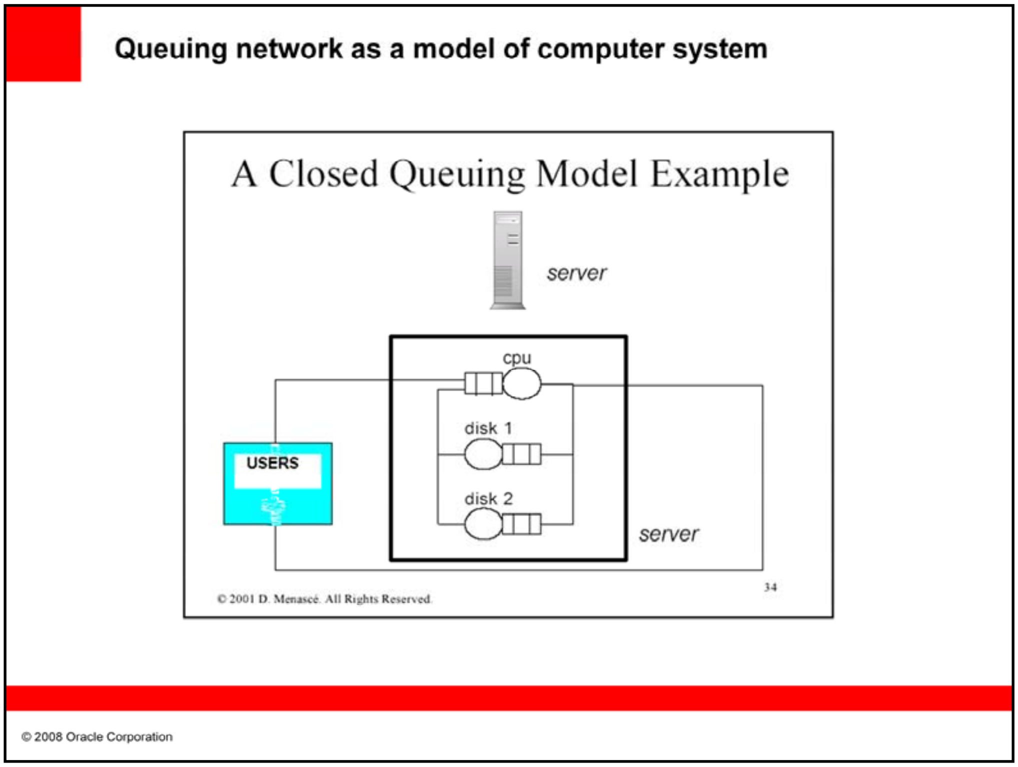
Wikipedia: “In the context of capacity planning, "capacity" is the maximum amount of work that an organization is capable of completing in a given period of time. “

Whatis.com: “In information technology, capacity planning is the science and art of estimating the space, computer hardware, software and connection infrastructure resources that will be needed over some future period of time. A typical capacity concern of many enterprises is whether resources will be in place to handle an increasing number of requests as the number of users or interactions increase. The aim of the capacity planner is to plan so well that new capacity is added just in time to meet the anticipated need but not so early that resources go unused for a long period. The successful capacity planner is one that makes the trade-offs between the present and the future that overall prove to be the most cost-efficient”.

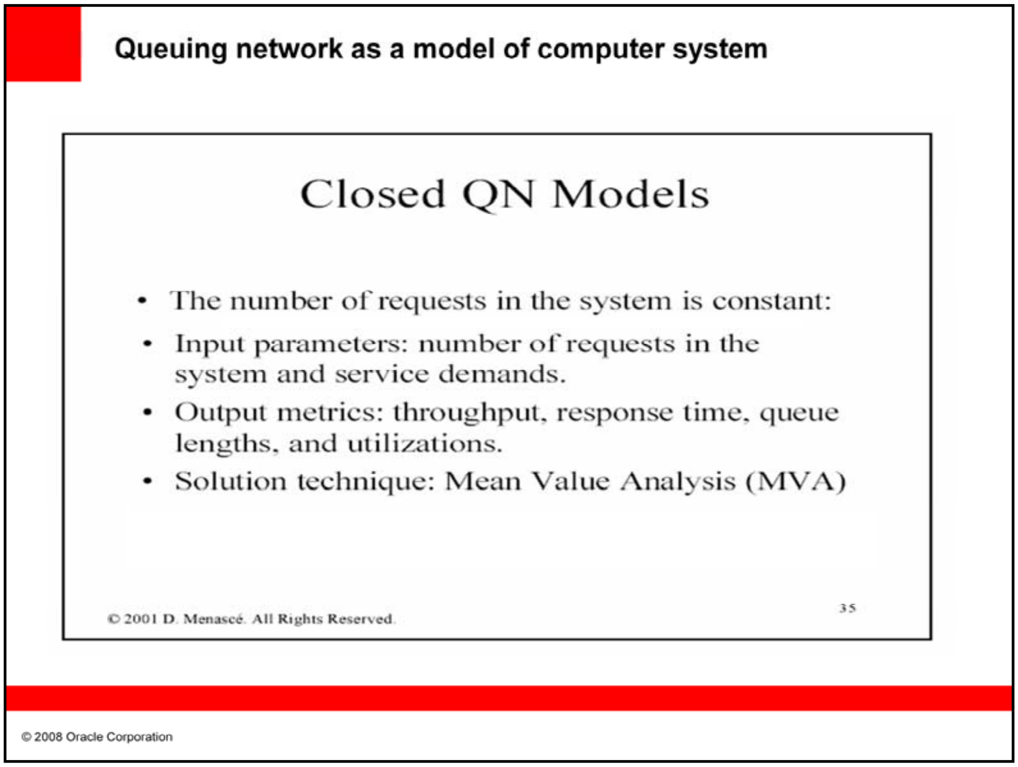
Presented methodology predicts not just resource utilizations, but also transaction response times which is must-have metric for business users.



Part 1 provides basic information of queuing network models

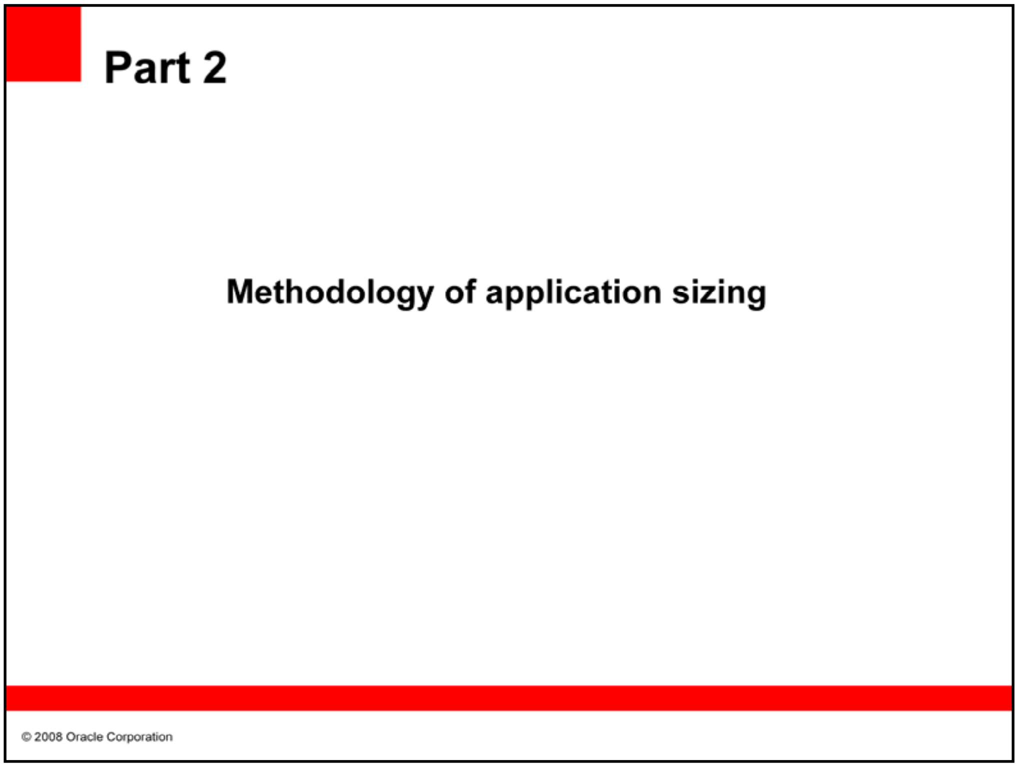


A user initiates transaction. Transaction is processed in a server for some period of time. User waits for processing to be completed BEFORE submitting a request for new transaction. Server is characterized by service time, user is characterized by think time. Think time is time between a moment a user receives a reply to transaction and the moment he/she submits a new transaction.

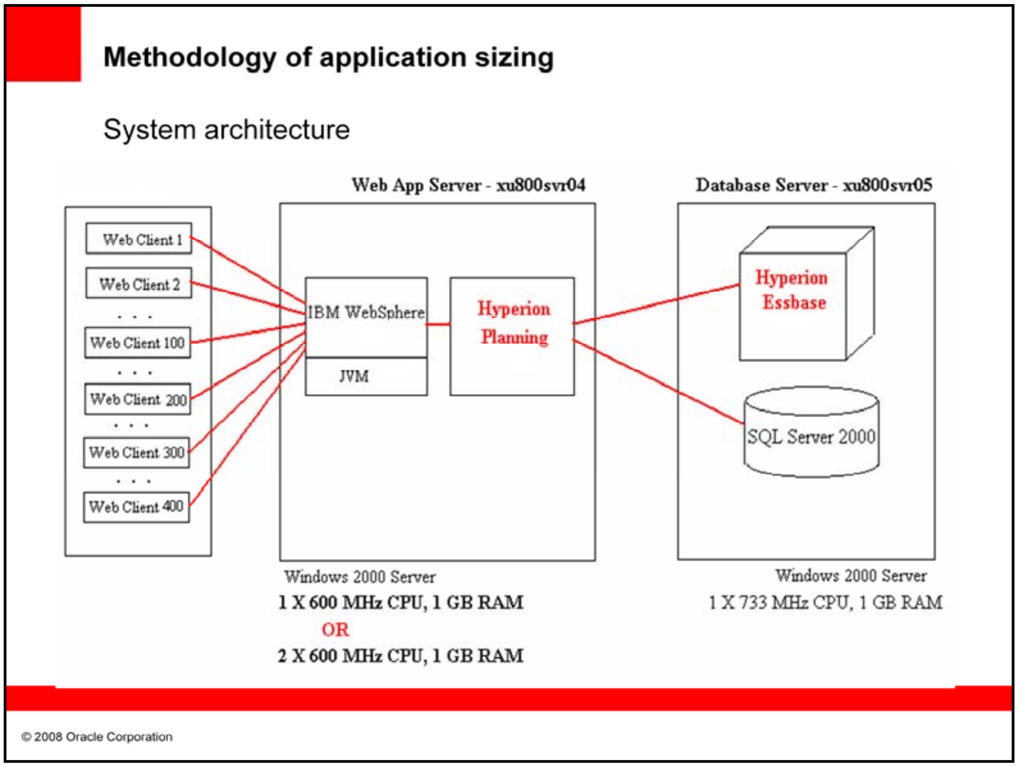


A few facts on models:

- Number of requests in system is equal to the number of system users.
- A request is an equivalent of a business transaction
- By solving model we getting metrics on transaction response times and server utilization.



Part 2 describes step by step methodology of application sizing which is based on load testing and queuing network modeling.



This picture presents a real production system which has application and database servers and has to support 400 concurrent users.

Methodology of application sizing
Workload specification

1. **Specify business transactions:**

   Calculation 1

   Open Form "Salaries"

   Report "Capital Expenses"

2. **For each transaction, specify its rate per user per hour and the number of concurrent users:**

Transaction name	Number of transactions per user per hour	Number of users
Calculation 1	**2**	**10**
Open Form "Salaries"	**4**	**20**
Report "Capital Expenses"	**10**	**50**

Workload is the most important input parameter for load testing and modeling. Testing and modeling results can be only as good as the workload specification.

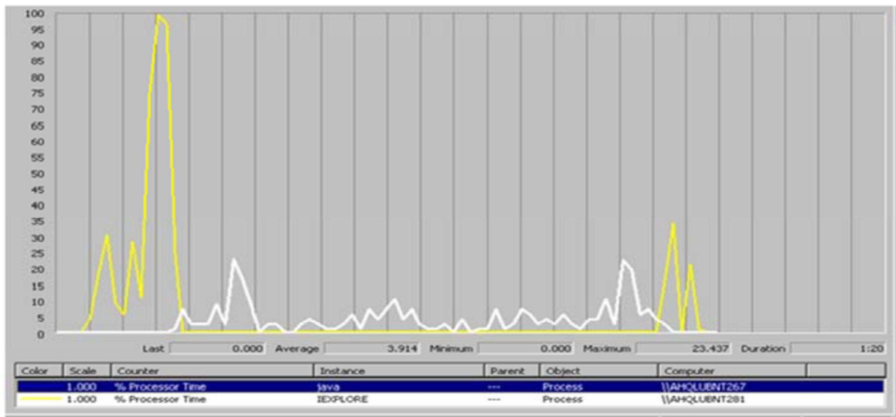
For real production systems, a workload has to describe as closely as possible the kinds of transactions executed by system users, as well as the number of transaction executions by one user per hour. A total number of users per each transaction has to be defined also.

Methodology of application sizing

Workload specification (continued)

3. Breaking down total transaction time by intervals spent on each server

Transaction time breakdown:
Total transaction time - 60.8 sec
Time spent on Planning server AHQLUBNT281 (yellow chart) - 16 seconds
Time spent on Essbase server AHQLUBNT267 (white chart) - 48 sec



A transaction can be compared to a car traveling on highway with toll booths. A toll booth can be considered as a server. A car (transaction) moves from one toll booth to another (from one server to another), spending some time in each toll booth (server). Total time in all toll booths (servers) is the transaction processing time.

Yellow line – utilization of Planning server by transaction

White line – utilization of Database server by transaction

This is how to find time spent by transaction on each server:

1. Turn on monitor and set it up to record CPU utilization on all servers
2. Run one transaction for a user
3. Note CPU activity on each server and time of that activity.

The time a transaction spends on a server is equal to the time a server's CPU is working. This is why by monitoring CPU utilization, we can find out how much time a given transaction spent on a server.

Methodology of application sizing

Workload specification (continued)

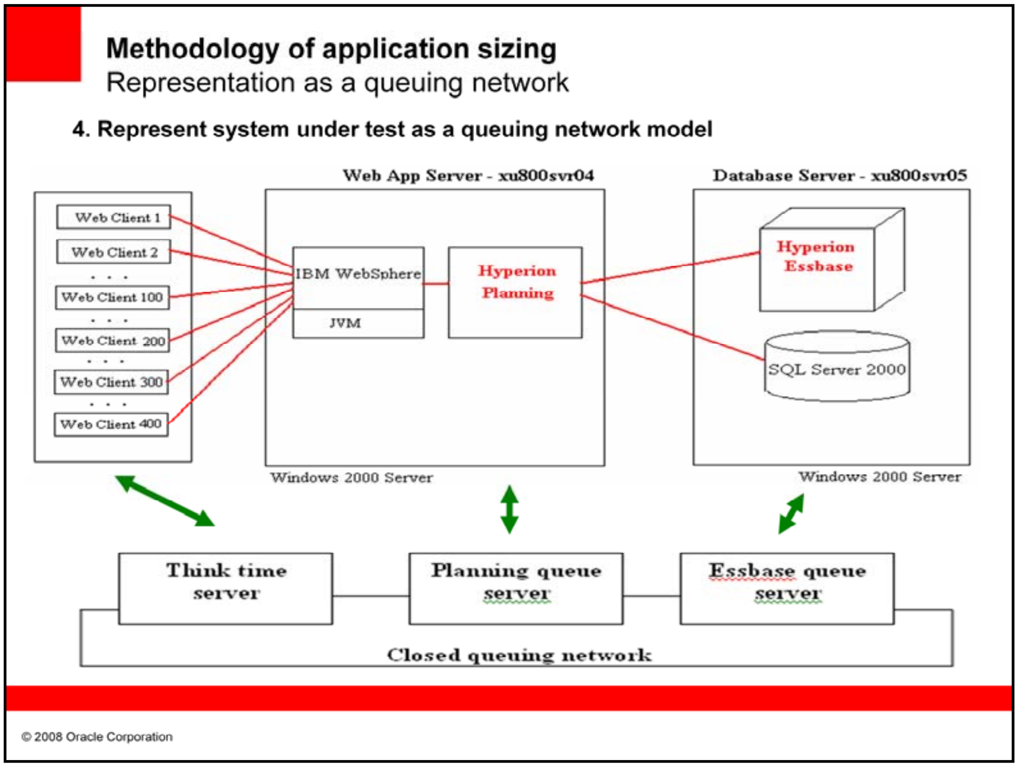
3.1. Workload description example

Transaction name and time for single user	Number of transactions per user per hour	Number of concurrent users	Transaction time breakdown obtained by monitoring (seconds)		
			Planning	Essbase	Think time
Calculation 1 7.5 seconds	20	10	2.0	5.5	3600 sec / 20 =180sec
Open Form "Salaries" 1.0 seconds	40	20	0.45	0.55	3600 sec /40 = 90 sec

Transaction time is broken down by monitoring a single transaction.

Think time is the time between two transactions that have been requested by the same user. Think time is calculated by dividing one hour by the number of transactions executed by one user in an hour.

The number of transactions per user per hour is actually a business metric, not a technical parameter. It can be found by interviewing business users or by monitoring their activity.



This step is all about morphing a real system into a closed queuing model.

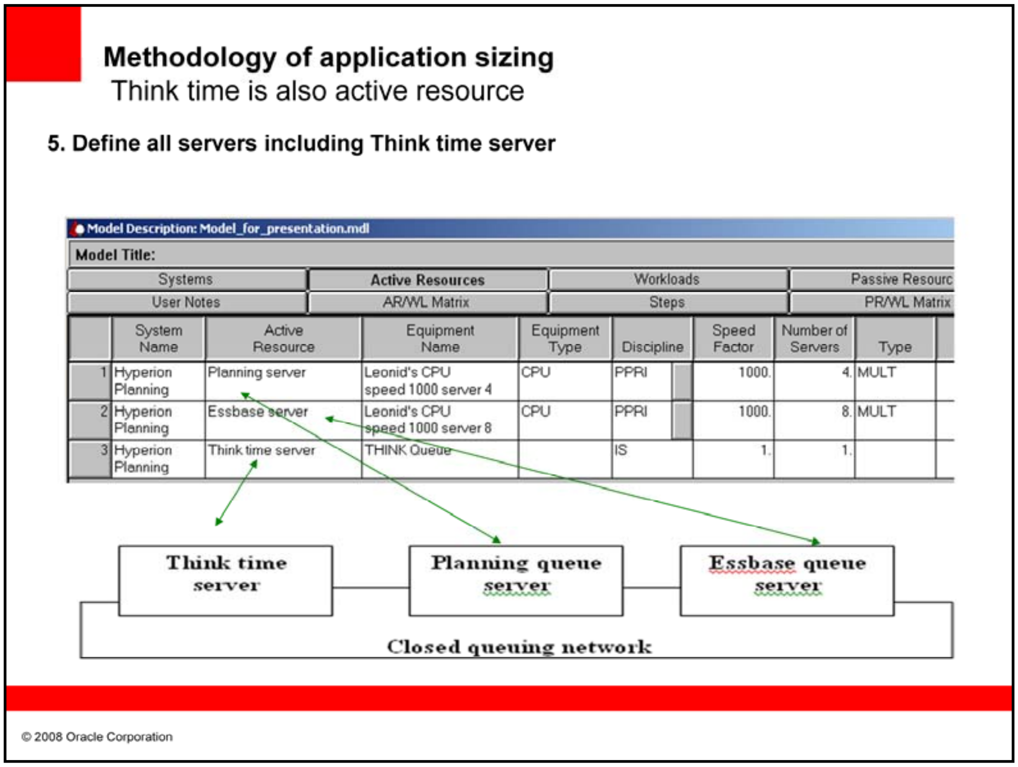
User is represented by a think time queue

Web and application servers are represented by Planning queue

Database is represented by database queue.

Transaction leaves think time queue, then receives service in the Planning server queue as well as in the database queue and returns back to the user. Total time spent by transaction in both Planning queue and Database queue is transaction response time.

If there is only one user in a system, than response time is equal to processing times in both queues. But when there are a number of concurrent users in a system, than waiting time becomes a substantial component of response time in addition to processing time.



Discipline
One of the following active resource queue disciplines:

FCFS First-come-first-served. Customers are serviced in the order they arrive. The customer is given its entire service requirement in one burst when its turn comes up.

FS Fair shared. Each customer receives service at a rate proportional to the relative shares assigned to this workload.

IS Infinite server. Any customer receives immediate service because enough servers exist to provide the requirements.

PPRI Preemptive priority. The customer in service is interrupted by any customer of higher priority. The interrupted customer's service is resumed after completion of the interrupting customer's service. Within a priority level, the discipline is FCFS.

PRI Non-preemptive priority. The customer in service cannot be interrupted. Within a priority level, the discipline is FCFS.

PS Processor shared. All customers are slowed down by the same ratio due to contention at the servers.

Methodology of application sizing

Workloads definition

6. Define workloads (one workload is a stream of the same business transaction)

Model Description: Model_for_presentation.mdl

Model Title:

Systems	Active Resources	Workloads	Passive Re
User Notes	AR/WL Matrix	Steps	PR/WL N

	System Name	Workload	Type	Measured Throughput	Throughput Adjustment Active Resource	Environment
1	Hyperion Planning	Calculation 1	CLOSED	1.	Think time server	INTERACTIVE
2	Hyperion Planning	Open Form "Salaries"	CLOSED	1.	Think time server	INTERACTIVE

A request in a closed workload does not enter or leave the system, there is a finite number of requests. A request traveling in a model represents one transaction initiated by one user. A number of requests in a queuing model is equal to a number of application users.

Open workloads have an infinite number of requests.

Methodology of application sizing

Resource/Workload matrix definition

7. Describe how each transaction travels across queuing model

Transaction name and time for single user	Number of transactions per user per hour	Number of concurrent users	Transaction time breakdown (seconds)		
			Planning	Essbase	Think time
Calculation 1 7.5 seconds	20	10	2.0	5.5	180
Open Form "Salaries" 1.0 seconds	40	20	0.45	0.55	90

	System Name	Workload	Active Resource	Visit Count	Service Required	Contribute to Response Time'
1	Hyperion Planning	Calculation 1	Planning server	1.	2000.	yes
2	Hyperion Planning	Calculation 1	Essbase server	1.	5500.	yes
3	Hyperion Planning	Calculation 1	Think time server	1.	180.	no
4	Hyperion Planning	Open Form "Salaries"	Planning server	1.	450.	yes
5	Hyperion Planning	Open Form "Salaries"	Essbase server	1.	550.	yes
6	Hyperion Planning	Open Form "Salaries"	Think time server	1.	90.	no

Resource/Workload matrix describes per each transaction which servers each transaction visited and how long time a transaction was processed on each one.

A column "Service required" defines time spent on a server.

Methodology of application sizing

Model verification for single user

8. Set to 1 a number of users for each workload and solve model

Transaction name and time for single user	Number of transactions per one user per one hour	Number of concurrent users	Transaction time breakdown (seconds)		
			Planning	Essbase	Think time
Calculation 1 7.5 seconds	2	10	2.0	5.5	1800
Open Form "Salaries" 1.0 seconds	4	20	0.45	0.55	900

Principal Results | AR Statistics

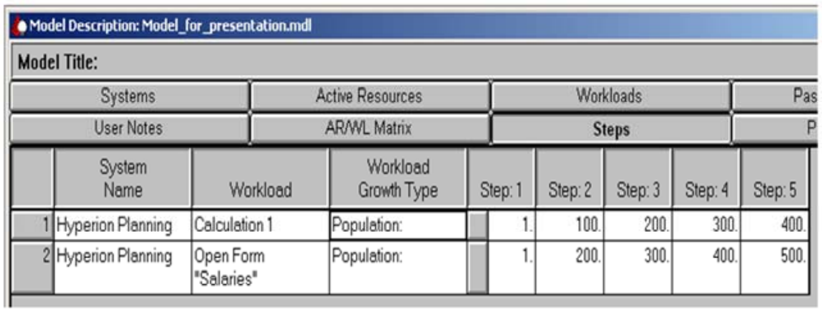
WL by AR Statistics | WL by PR Statistics

	System Name	Workload	Throughput	Response	Population
1	Hyperion Planning	Calculation 1	0.005333	7.5	1.
2	Hyperion Planning	Open Form "Salaries"	0.01099	1.	1.

We calibrate the model for a single user. Calibration means a model calculation for a single user and comparison of results with sizing requirements. If there are discrepancies, than the model has to be modified.

Methodology of application sizing
Setting up user population

9. Set different number of users



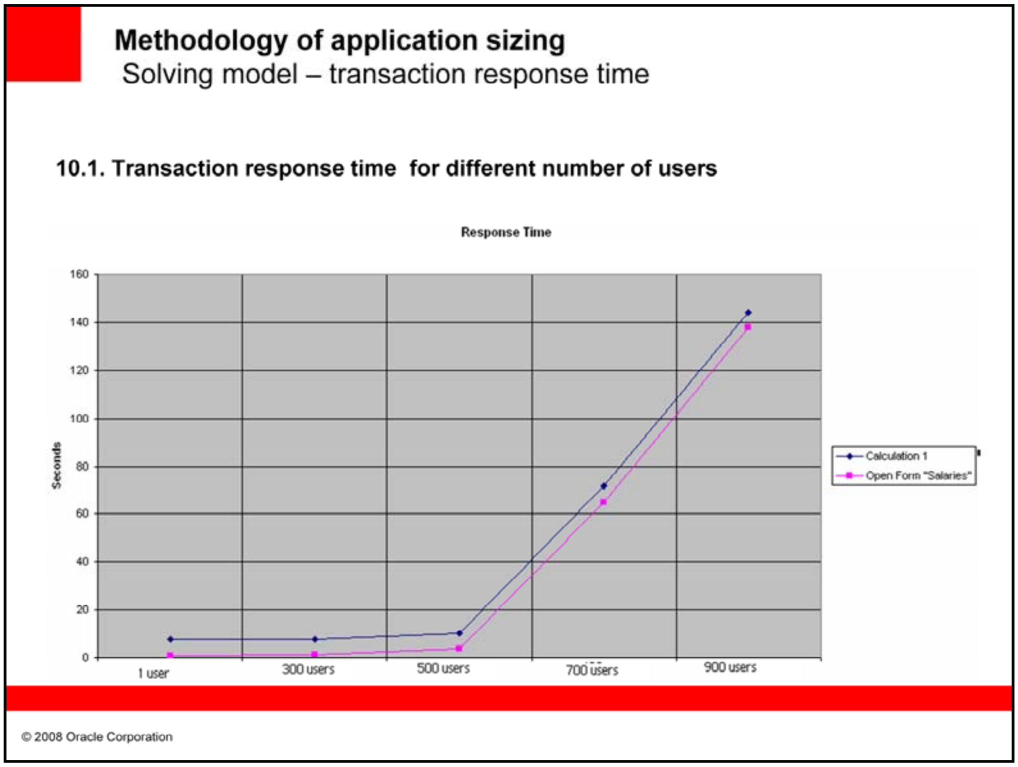
Model Description: Model_for_presentation.mdl

Model Title:

Systems | Active Resources | Workloads | Pas
User Notes | AR/WL Matrix | Steps | P

	System Name	Workload	Workload Growth Type		Step: 1	Step: 2	Step: 3	Step: 4	Step: 5
1	Hyperion Planning	Calculation 1	Population:		1.	100.	200.	300.	400.
2	Hyperion Planning	Open Form "Salaries"	Population:		1.	200.	300.	400.	500.

Model can predict system characteristics for different number of users.

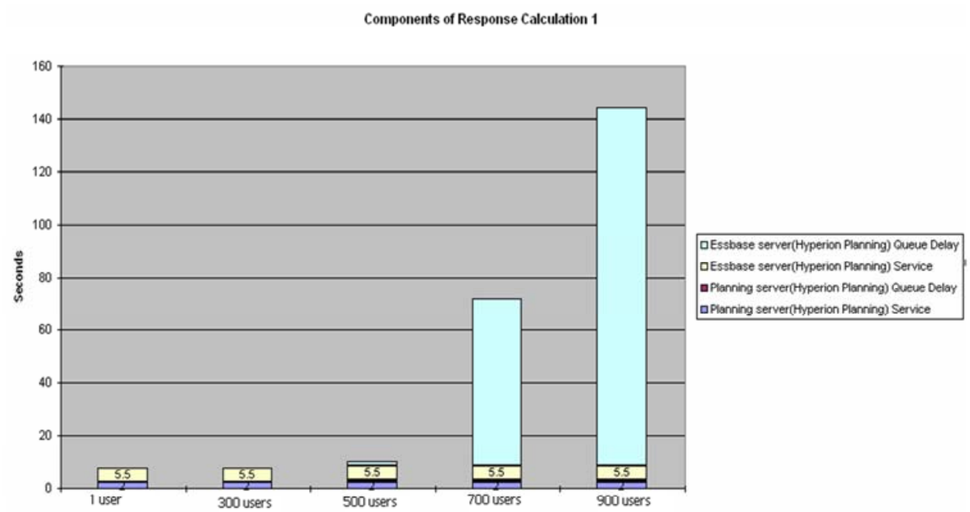


Transaction response time is flat or increases only a little when the number of users increases, up to the point where queuing starts happening. Then, response time jumps exponentially. A chart in this slide demonstrates the classical “hockey stick”, with its angle at step 3 when there were 500 concurrent users.

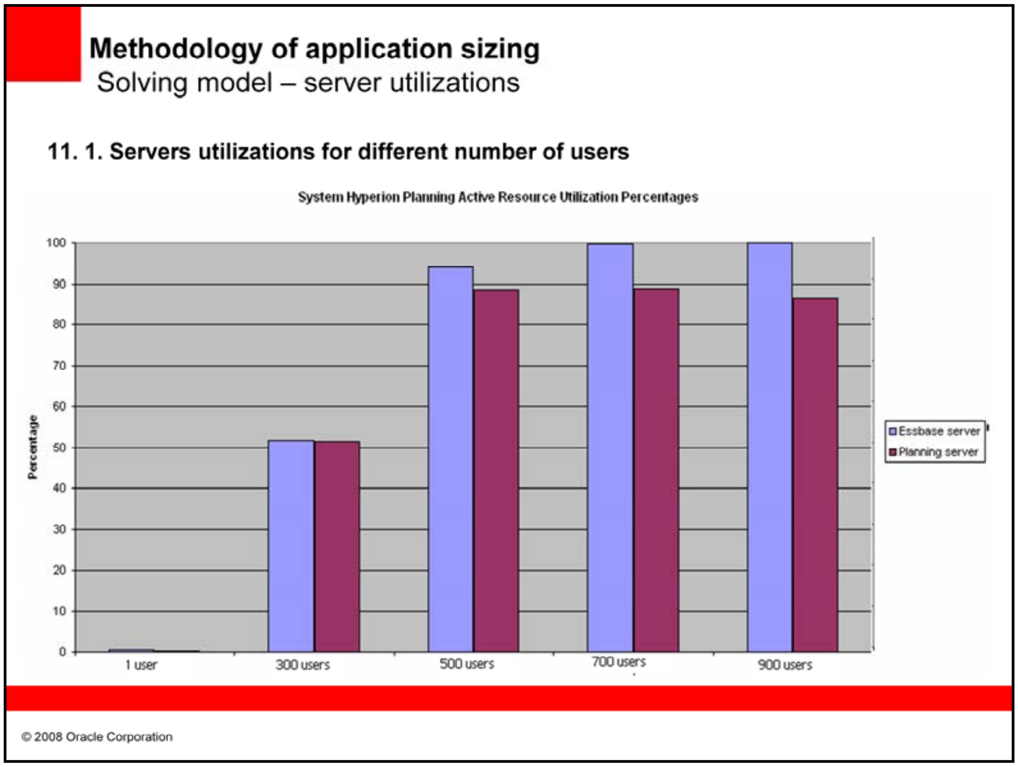
Methodology of application sizing

Solving model – transaction time breakdown

10.2. Breakdown of *Calculation 1* transaction for different number of users



Solved model delivers time spent by each transaction on each server (which is equal to time in CPU and time waiting for CPU).



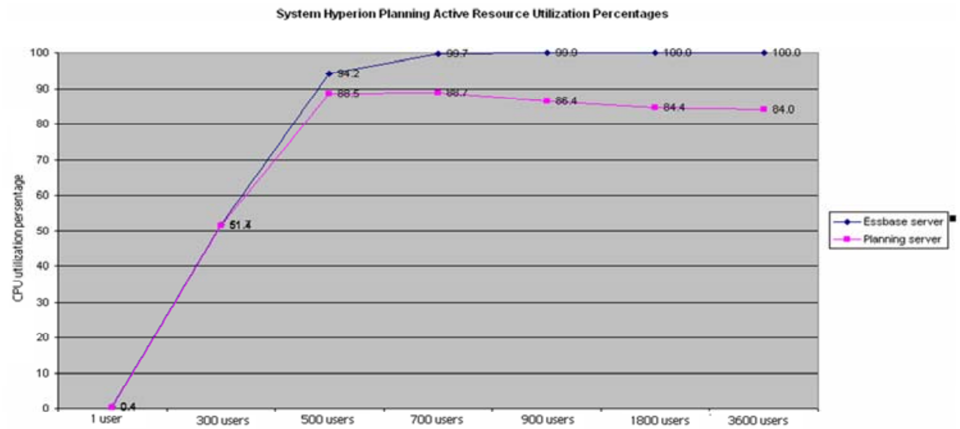
Model is solved for 1, 300, 500, 700, and 900 users

Methodology of application sizing

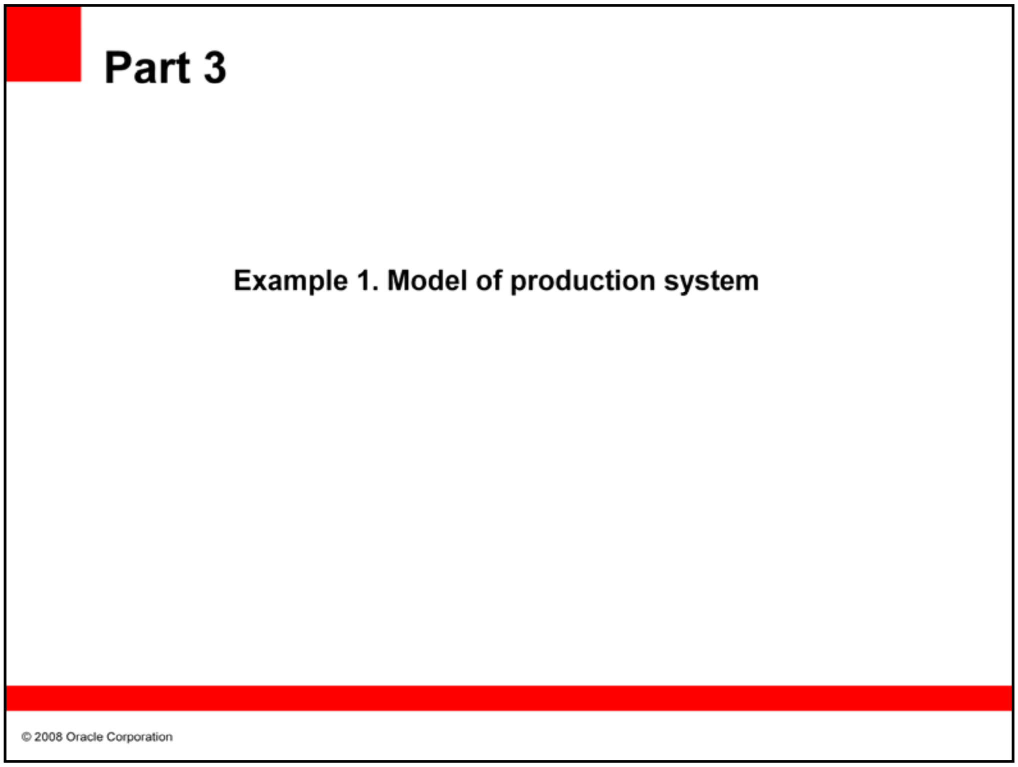
Solving model – server utilizations

11.2. Servers utilizations as a number of users grows significantly

	System Name	Workload	Workload Growth Type		Step: 1	Step: 2	Step: 3	Step: 4	Step: 5	Step: 6	Step: 7
1	Hyperion Planning	Calculation 1	Population:		1.	100.	200.	300.	400.	800.	1600.
2	Hyperion Planning	Open Form "Salaries"	Population:		1.	200.	300.	400.	500.	1000.	2000.



Utilization of Planning server has a downtrend as the number of users grows. Explanation: more and more requests are queued in Essbase server which reached almost 100% of its capacity on Step 4. That means Planning server has a less intense flow of requests.

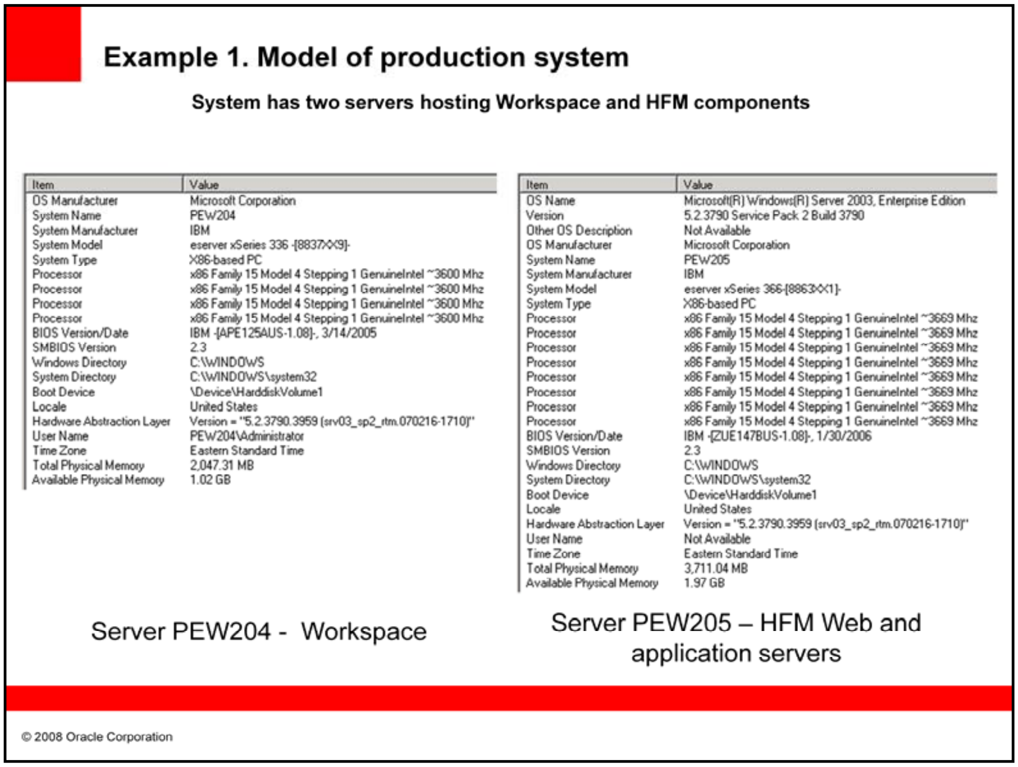


This part of presentation demonstrates sizing methodology “in action”.

On the first step we collected the information necessary for modeling data by applying a load from concurrent users to a real production system with an enterprise application.

On the second step we built a queuing model of a system and solved the model using collected performance data as model input.

On the third step we evaluated results and analyzed different what-if scenarios for various system architectures



Example 1. Model of production system

System has two servers hosting Workspace and HFM components

Item	Value
OS Manufacturer	Microsoft Corporation
System Name	PEW204
System Manufacturer	IBM
System Model	eserver xSeries 336 -[8837XX9]-
System Type	X86-based PC
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3600 Mhz
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3600 Mhz
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3600 Mhz
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3600 Mhz
BIOS Version/Date	IBM -[APE125AUS-1.08]-, 3/14/2005
SMBIOS Version	2.3
Windows Directory	C:\WINDOWS
System Directory	C:\WINDOWS\system32
Boot Device	\Device\HarddiskVolume1
Locale	United States
Hardware Abstraction Layer	Version = "5.2.3790.3959 (srv03_sp2_rtm.070216-1710)"
User Name	PEW204\Administrator
Time Zone	Eastern Standard Time
Total Physical Memory	2,047.31 MB
Available Physical Memory	1.02 GB

Server PEW204 - Workspace

Item	Value
OS Name	Microsoft(R) Windows(R) Server 2003, Enterprise Edition
Version	5.2.3790 Service Pack 2 Build 3790
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	PEW205
System Manufacturer	IBM
System Model	eserver xSeries 366-[8863XX1]-
System Type	X86-based PC
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3669 Mhz
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3669 Mhz
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3669 Mhz
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3669 Mhz
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3669 Mhz
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3669 Mhz
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3669 Mhz
Processor	x86 Family 15 Model 4 Stepping 1 GenuineIntel ~3669 Mhz
BIOS Version/Date	IBM -[ZUE147BUS-1.08]-, 1/30/2006
SMBIOS Version	2.3
Windows Directory	C:\WINDOWS
System Directory	C:\WINDOWS\system32
Boot Device	\Device\HarddiskVolume1
Locale	United States
Hardware Abstraction Layer	Version = "5.2.3790.3959 (srv03_sp2_rtm.070216-1710)"
User Name	Not Available
Time Zone	Eastern Standard Time
Total Physical Memory	3,711.04 MB
Available Physical Memory	1.97 GB

Server PEW205 – HFM Web and application servers

To build a model we have to know system architecture as well as specifications of servers. This slide indicates that system has two servers. It also shows the number of CPUs per each server and CPU speeds.

Example 1. Model of production system

Load test results – transaction consolidation and transaction time for single user

Transaction times from LoadRunner runs for one user (seconds)

(HFM 9.3.1 scripts HFM01Z)

Transaction Name	Minimum	Average	Maximum	Std. Devi	90 Percent	Pass	Fail	Model transaction times (seconds)	Average
DP-00-LogonPage	1.179	1.179	1.179	0	1.179	1	0	LogonOpenApp	5.1
DP-01-LogonWS	2.811	2.811	2.811	0	2.811	1	0		
DP-02-OpenApplication	1.134	1.134	1.134	0	1.134	1	0		
vuser_init_Transaction	5.125	5.125	5.125	0	5.125	1	0		
DP-11-ConsolidateParent	10.213	10.244	10.289	0.023	10.265	13	0	ConsolidateParent	10.2
DP-04-LoadFile	3.086	3.102	3.127	0.012	3.127	13	0	LoadFile	3.1
DP-08-ForceCalculate	6.735	6.793	6.976	0.06	6.851	13	0	ForceCalculate	6.8
DP-05-GotoProcessControl	0.37	0.413	0.638	0.078	0.541	13	0	Navigate	3.2
DP-06-SetPOV	0.519	0.583	0.743	0.071	0.735	13	0		
DP-07-SelectEntity	0.776	0.817	1.164	0.101	0.807	13	0		
DP-09-GotoProcessControl	0.265	0.275	0.282	0.004	0.278	13	0		
DP-10-SelectParent	0.78	0.816	0.979	0.05	0.834	13	0		
DP-12-GotoTasks	0.009	0.011	0.016	0.002	0.016	13	0		
DP-03-GotoLDtask	0.265	0.278	0.327	0.015	0.283	13	0		
DP07_Transaction	23.12	23.333	23.786	0.209	23.767	13	0		
DP-13-LogoffWS	0.241	0.241	0.241	0	0.241	1	0	Logoff	0.241
vuser_end_Transaction	0.241	0.241	0.241	0	0.241	1	0		

We will model only framed transactions because LogonOpenApp and Logoff are one-time transactions

Load test application collected response times for 16 transactions. Logon and Logoff transactions are executed only once by each user and can be excluded from model workload.

Workload has three main transactions: ConsolidateParents, LoadFile, and ForceCalculate. All remaining transactions will be consolidated into the one called “Navigate”, because each of the remaining transactions are pretty light in terms of resource demand. By consolidating transactions we minimize our modeling efforts without compromising the applicability of the model.

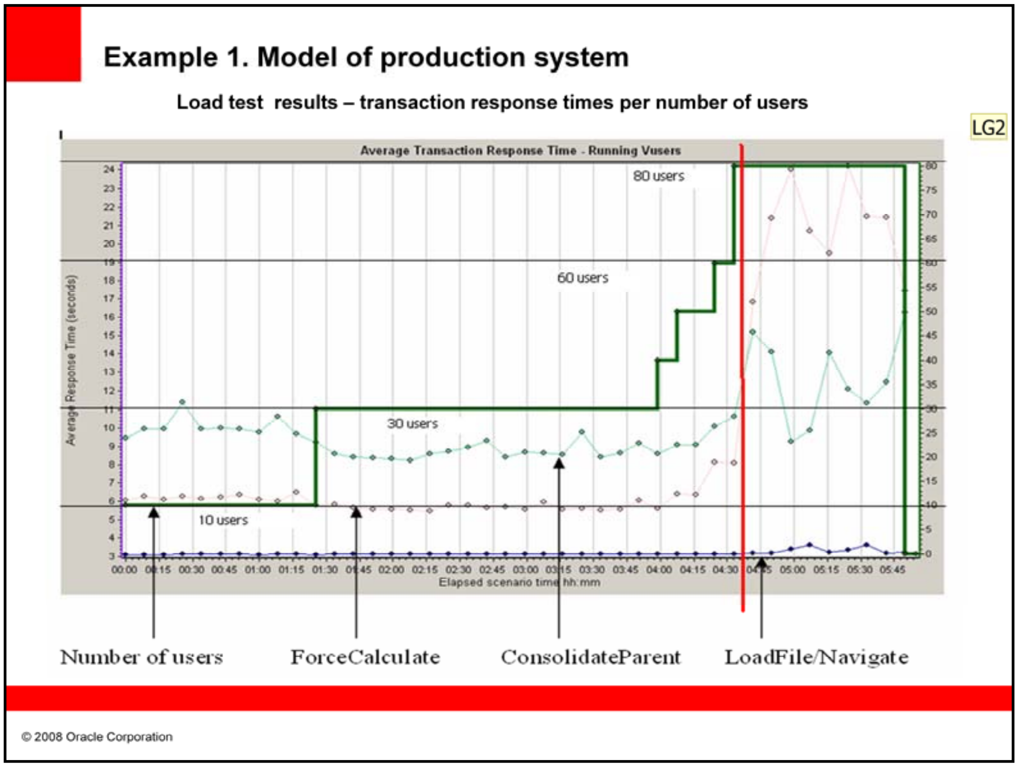


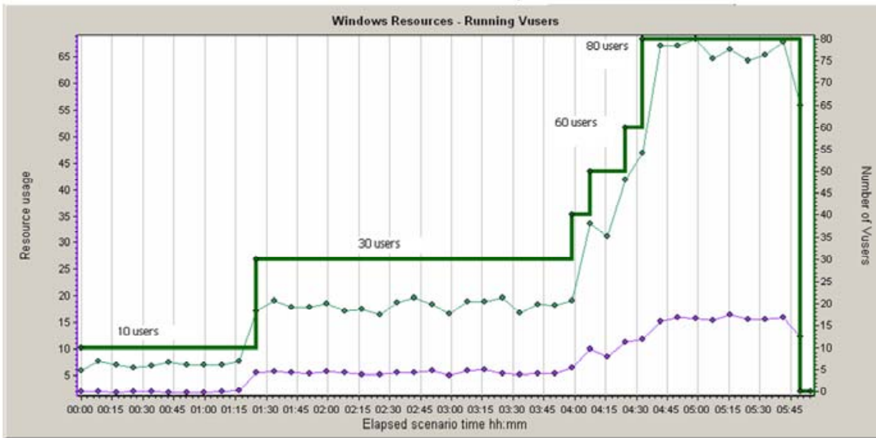
Chart demonstrates response time per each transaction for different number of users: 10, 30, 60, and 80.

There is a pretty interesting effect – transaction ‘ForceCalculate” is faster than transaction “ConsolidateParents” for 10 and 30 users, but when a number of users is reaching 60 it becomes significantly slower. This is an indication that transaction “ForceCalculate” started experience some limitation at the software level – limited number of threads, or database locking, or shortage of memory.

Later on we will show how that effect can be reflected in a model.

Example 1. Model of production system

Load test results – server utilizations per number of users



Legend | Alerts | Graph Details | User Notes | Graph Data | Raw Data

Color	Graph	Scale	Measurement	Graph's Mini...	Graph's Ave...	Graph's Max...	Graph's Me...	Graph's Std....	Machine
☑	Windows Resources	1	% Processor Time (Processor _Total):pew204	1.743	7.207	16.416	5.486	4.861	Pew204
☑	Windows Resources	1	% Processor Time (Processor _Total):pew205	5.897	27.536	68.481	18.609	21.541	Pew205
☑	Running Vusers	1	Run	0	33.75	80	40	27.358	N/A

Chart shows utilization of both servers for different numbers of users.

Example 1. Model of production system

Load test results provide data for model calibration

Transaction response times

Number of users		1	10	30	60	80
CalculateParent (sec)	Load test	10.2	9.8	9.8	10.5	13.1
	Model					
ForcCalculate (sec)	Load test	6.7	6.0	6.1	10.0	21.8
	Model					
LoadFile (sec)	Load test	3.3	3.2	3.2	3.2	3.3
	Model					
Navigate (sec)	Load test	3.3	3.2	3.2	3.2	3.3
	Model					

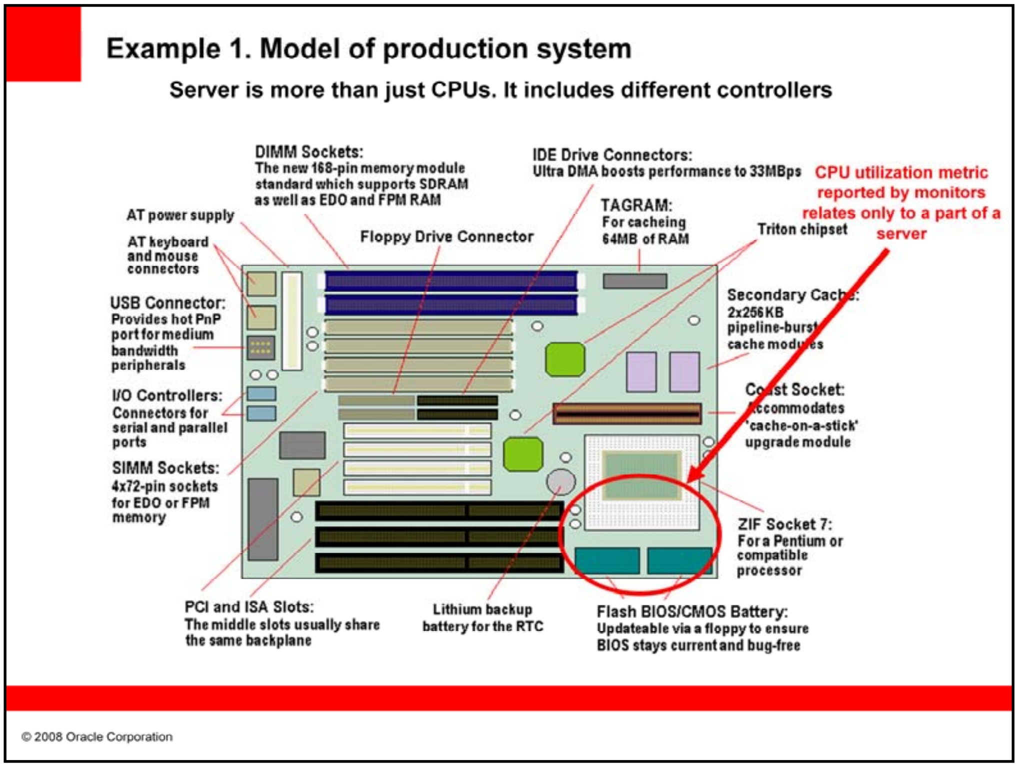
Server utilizations

Number of users		10	30	60	80
HFM server (%)	Load test	8.0	18.1	43.2	65.1
	Model				
Workspace server (%)	Load test	2.0	5.1	10.0	13.5
	Model				

Calibrated model has to deliver transaction response times and serve utilizations as close as possible to the values measured during load test

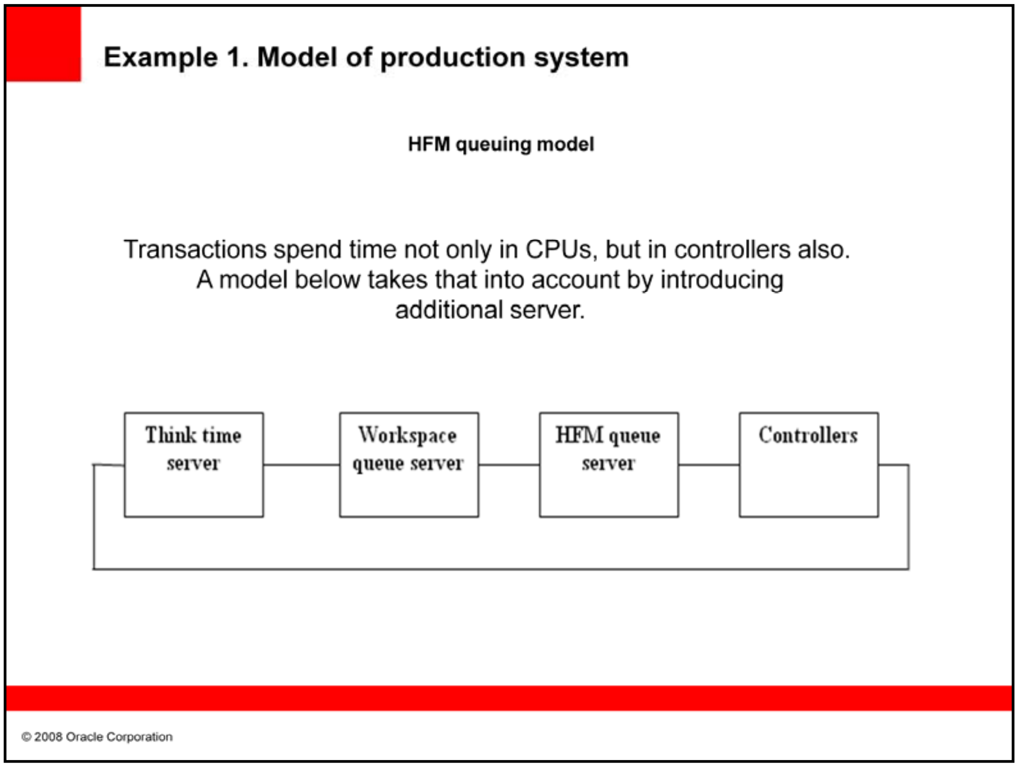
After running Load test we collected data needed for building and solving model. We obtained response time for each transaction for different number of users, as well as utilizations of both servers.

Important to note; calibrated model has to deliver transaction response times and server utilizations as close as possible to the values measured by during load test.



A picture of a computer motherboard demonstrates that a server is much more that a collection of CPUs and memory. It includes different controllers which by their nature are specialized computers managing I/O operations, memory operations, video processing etc.

CPU utilization reported by monitoring tools only relates to the part of a server which is CPU, but does not reflect processing carried out by other controllers.

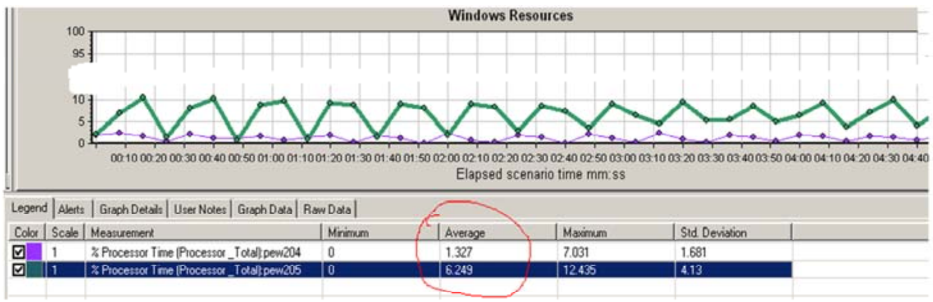


To factor in the impact of controllers on system performance we included an additional queuing module representing all controllers.

Example 1. Model of production system

Workload description - 1

Initial transaction time breakdown based on server utilizations



Based on the recorded average utilizations of the servers, each transaction spent 0.17th of its total time in pew204 and 0.83rd of its total time in pew205

Breaking down transaction time based on server utilization might be correct or somewhat correct. We should not worry about it for now, we will change those number while calibrating model, but at that point of modeling process we have to have the values to begin with.

To find out transaction time breakdown we set up a run for a single user repeatedly executing. Time breakdown is proportional to servers utilizations.

Time transaction spent on a server 1 = 1/ average service rate 1

Time transaction spent on a server 2 = 1/ average service rate 2

Average server 1 utilization =

= average arrival rate / average service rate 1

Average server 2 utilization =

= average arrival rate / average service rate 2

Average server 1 utilization / Average server 2 utilization =

=average service rate 2/ average service rate 1

Finally:

Time transaction spent on a server 1 / Time transaction spent on a server 2 =

= Average server 1 utilization / Average server 2 utilization

Example 1. Model of production system
- Workload Description 2

Transaction name and time for single user	Number of transactions per one user per one hour	Number of concurrent users	Transaction time breakdown (seconds)			Think time
			Workspace pew204	HFM pew205	Controllers	
ConsolidateParent 10.2 seconds			0.73	5.54	3.9	
LoadFile 3.3 seconds			0.11	1.43	1.74	
ForceCalculate 6.7 seconds			0.25	3.48	2.99	
Navigate 3.3 seconds			0.11	1.43	1.74	

Now we started to consolidate all input data that describes the workload into the table. This is the transaction time breakdown

Example 1. Model of production system

Workload description - 3

Think time and the number of concurrent users

Transaction name and time for single user	Number of transactions per one user per one hour	Number of concur-rent users	Transaction time breakdown (seconds)			Think time
			Workspace pew204	HFM pew205	Controllers	
ConsolidateParent 10.2 seconds	20	10 - 80	0.73	5.54	3.9	3600 / 20 =180
LoadFile 3.3 seconds	20	10 - 80	0.11	1.43	1.74	3600 / 20 =180
ForceCalculate 6.7 seconds	20	10 - 80	0.25	3.48	2.99	3600 / 20 =180
Navigate 3.3 seconds	20	10 - 80	0.11	1.43	1.74	3600 / 20 =180

After getting filled all the numbers into table we have pretty good realistic description of production workload generated by a SINGLE user.

We can start populating model with data now.

Example 1. Model of production system

Model description - 1

Systems	**Active Resources**	Workloads	Passive Resources
User Notes	AR/WL Matrix	Steps	PR/WL Matrix

	System Name	Active Resource	Equipment Name	Equipment Type	Discipline	Speed Factor	Number of Servers	Type	Path
1	HFM FIAT	HFM server	Intel Xeon 7150N 3.5GHz/1MB/16MB	CPU	PPRI	1756.33	8.	MULT	
2	HFM FIAT	Workspace server	Intel Xeon 5080 3.73GHz/2MB	CPU	PPRI	1749.85	4.	MULT	
3	HFM FIAT	Think time	THINK Queue		IS	1.	1.		
4	HFM FIAT	Controllers		Unknown Type	IS	1000.	1.		

First we define servers.

Time in HFM server per one visit: 1 sec / 1756.33 = 0.000569 sec

Time in Workspace server per one visit: 1 sec / 1749.85 = 0.000571 sec

Time in Controllers server per one visit: 1 sec / 1000 = 0.001 sec

Example 1. Model of production system

Model description - 2

Systems | Active Resources | **Workloads** | Passive Resources
User Notes | AR/WL Matrix | Steps | PR/WL Matrix

	System Name	Workload	Type	Measured Throughput	Throughput Adjustment Active Resource	Environment
1	HFM FIAT	ConsolidateParent	CLOSED	1.	Think time	INTERACTIVE
2	HFM FIAT	LoadFile	CLOSED	1.	Think time	INTERACTIVE
3	HFM FIAT	ForceCalculate	CLOSED	1.	Think time	INTERACTIVE
4	HFM FIAT	Navigate	CLOSED	1.	Think time	INTERACTIVE

Systems | Active Resources | Workloads | **Passive Resources**
User Notes | AR/WL Matrix | Steps | PR/WL Matrix

	System Name	Passive Resource	Equipment Type	Capacity
1	HFM FIAT	Database locking	Software Queue	120.
2	HFM FIAT	Database locking_2	Software Queue	120.

Because we observed an impact of software limitations on transaction response time, we analyzed the system more closely and found that database locking is affecting response time. This is why we introduced into model passive resources called “Database_locking” and ‘Database_locking_2”.

Those resources are affecting transactions “ForceCalculate” and “ CalculateParents”. We indicated total capacity of each resource as 120 and later on we will indicate the size of the resource’s capacity each transaction will take during its execution.

The process of defining passive resource capacity and the chunk a transaction takes while execution is iterative – we have to define and redefine those values during model calibration process.

Example 1. Model of production system

Model description - 3

Systems	Active Resources	Workloads	Passive Resources
User Notes	**AR/WL Matrix**	Steps	PR/WL Matrix

	System Name	Workload	Active Resource	Visit Count	Service Required	Contribute to Response Time'	Affected Passive Resource	Fair Share
1	HFM FIAT	ConsolidateParent	HFM server	1.	9737.	yes	Database locking	
2	HFM FIAT	ConsolidateParent	Workspace server	1.	1277.	yes	Database locking	
3	HFM FIAT	ConsolidateParent	Think time	1.	180.	no		
4	HFM FIAT	ConsolidateParent	Controllers	1.	3890.	yes	Database locking	
5	HFM FIAT	LoadFile	HFM server	1.	2512.	yes		
6	HFM FIAT	LoadFile	Workspace server	1.	187.	yes		
7	HFM FIAT	LoadFile	Think time	1.	180.	no		
8	HFM FIAT	LoadFile	Controllers	1.	1744.	yes		
9	HFM FIAT	ForceCalculate	HFM server	1.	6118.	yes	Database locking_2	
10	HFM FIAT	ForceCalculate	Workspace server	1.	433.	yes	Database locking_2	
11	HFM FIAT	ForceCalculate	Think time	1.	180.	no		
12	HFM FIAT	ForceCalculate	Controllers	1.	2994.	yes	Database locking_2	
13	HFM FIAT	Navigate	HFM server	1.	2512.	yes		
14	HFM FIAT	Navigate	Workspace server	1.	187.	yes		
15	HFM FIAT	Navigate	Think time	1.	180.	no		
16	HFM FIAT	Navigate	Controllers	1.	1744.	yes		

We described that transaction "ForceCalculate" needs Database_locking_2 passive resource; transaction "ConsolidateParents" needs Database_locking resource.

Example 1. Model of production system

Model description - 4

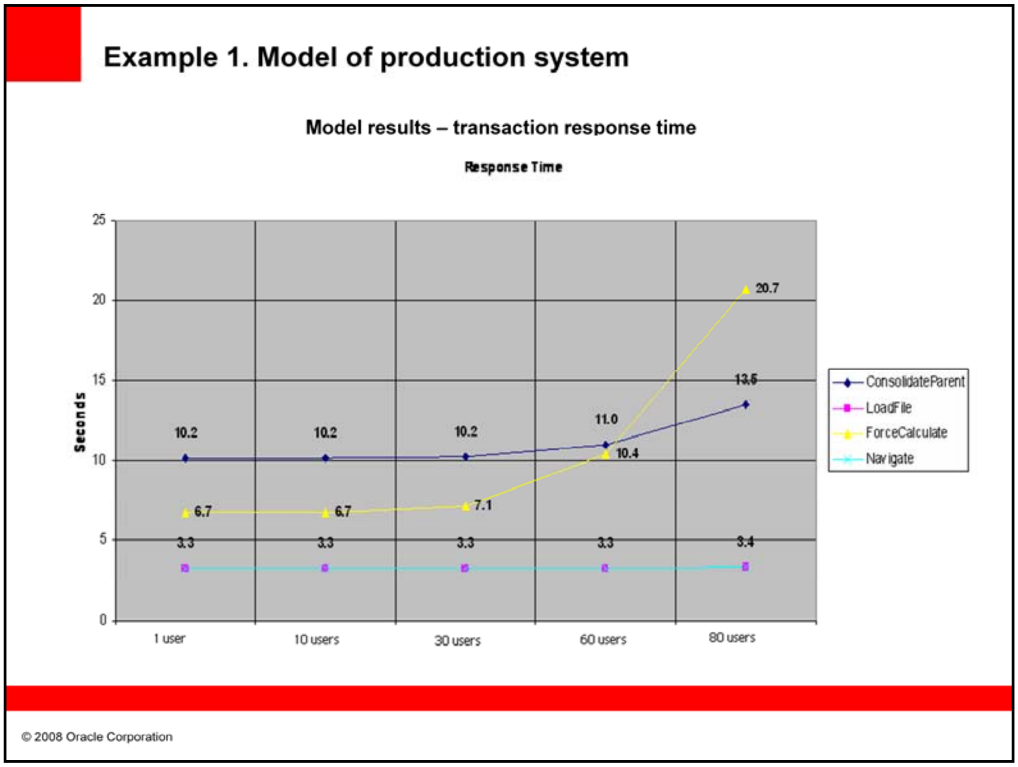
Systems	Active Resources	Workloads	Passive Resources
User Notes	AR/WL Matrix	**Steps**	PR/WL Matrix

	System Name	Workload	Workload Growth Type		Step: 1	Step: 2	Step: 3	Step: 4	Step: 5
1	HFM FIAT	ConsolidateParent	Population:		1.	10.	30.	60.	80.
2	HFM FIAT	LoadFile	Population:		1.	10.	30.	60.	80.
3	HFM FIAT	ForceCalculate	Population:		1.	10.	30.	60.	80.
4	HFM FIAT	Navigate	Population:		1.	10.	30.	60.	80.

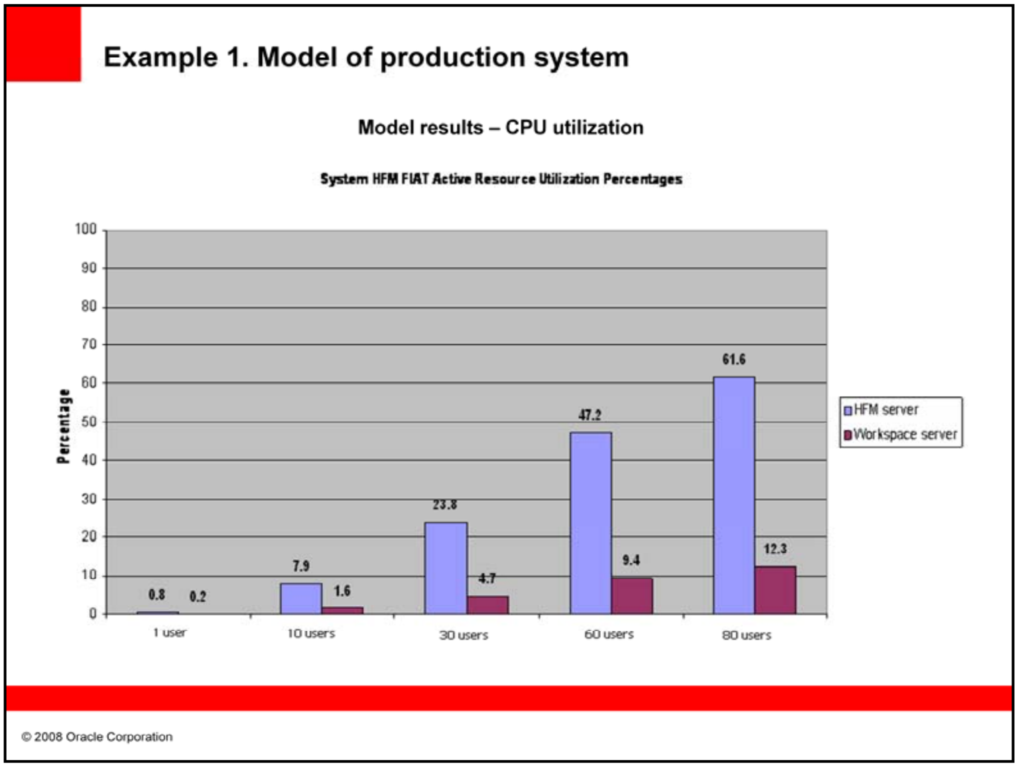
Systems	Active Resources	Workloads	Passive Resources
User Notes	AR/WL Matrix	Steps	**PR/WL Matrix**

	System Name	Workload	Passive Resource	Size Required
1	HFM FIAT	ConsolidateParent	Database locking	22.
2	HFM FIAT	ConsolidateParent	Database locking_2	0.
3	HFM FIAT	LoadFile	Database locking	0.
4	HFM FIAT	LoadFile	Database locking_2	0.
5	HFM FIAT	ForceCalculate	Database locking	0.
6	HFM FIAT	ForceCalculate	Database locking_2	41.
7	HFM FIAT	Navigate	Database locking	0.
8	HFM FIAT	Navigate	Database locking_2	0.

Here we indicated what is the size of passive resource is consumed by each transaction.



We solved the model and got transaction response times. Looks like we were able to model database locking impact.



Model also delivered utilizations of both servers for different number of concurrent users.

Example 1. Model of production system

Load test and model results comparison

Transaction response times

Number of users		1	10	30	60	80
CalculateParent (sec)	Load test	10.2	9.8	9.8	10.5	13.1
	Model	10.2	10.2	10.2	11.0	13.5
ForcCalculate (sec)	Load test	6.7	6.0	6.1	10.0	21.8
	Model	6.7	6.7	7.1	10.4	20.7
LoadFile (sec)	Load test	3.3	3.2	3.2	3.2	3.3
	Model	3.3	3.3	3.3	3.3	3.4
Navigate (sec)	Load test	3.3	3.2	3.2	3.2	3.3
	Model	3.3	3.3	3.3	3.3	3.4

Server utilizations

Number of users		10	30	60	80
HFM server (%)	Load test	8.0	18.1	43.2	65.1
	Model	7.9	23.8	47.2	61.6
Workspace server (%)	Load test	2.0	5.1	10.0	13.5
	Model	1.6	4.7	9.4	12.3

Looking into tables we can say that our model is in pretty good accord with data collected during load test. We can say that we have calibrated our model and we can now use a model to analyze what-if scenarios.

Overview of model of production system building and calibration

- Take into account all transactions generated by your transaction driving application. Never ignore short transactions, consolidate them into one long transaction.
- Introduce into model a block emulating computer controllers – the server is more than just a CPU; controllers contribute to transaction time.
- Break down the time of each transaction into the number of time segments spent in servers and controllers
- Use a closed queuing model with interactive workload
- Start building model from benchmark CPU and then change CPU to the one in the modeling tool's CPU model list that is closest to the server you are modeling.

This slide highlights some milestones in a process of model building and calibration.

Overview of model of production system building and calibration (continued)

- Column “Service required” in AR/WL matrix defines a number of time slices needed to process a transaction on a particular CPU. If the Speed Factor of a CPU is *S*, than the time spent by a request in a server per visit is equal to:
  1 / S seconds.

If the transaction time spent in a server is *TR*, than a request will require *TR/1/S = TR*S* time slices and column “Service required” will have a value of *TR/1/S = TR*S.*

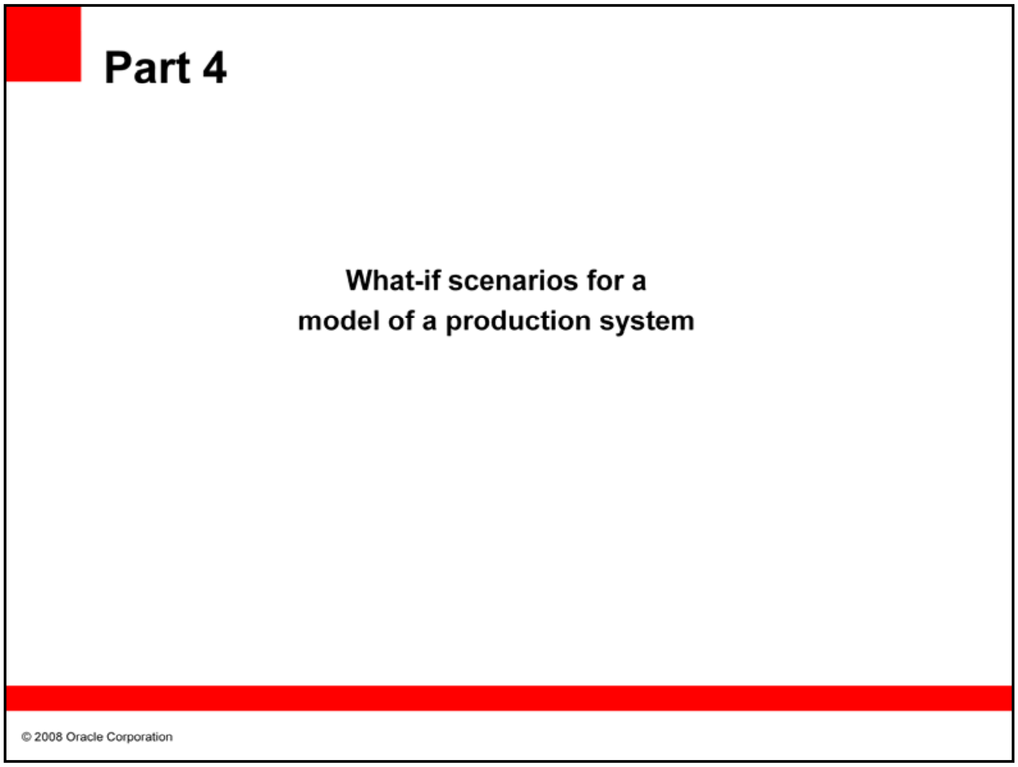
- Use Passive Resources to model effects of database locking, memory size, and thread count on transaction time and server utilization.

This slide highlights some milestones in a process of model building and calibration.

Overview of model of production system building and calibration (continued)

- Calibrate model to ensure that transaction times and server utilizations calculated by model are in line with the values delivered by your transaction emulating software. Use data showing different number of users for calibration.
- Calibration technique is based on trial and error – choose input values and calculate output values. If output is not close enough, start all over - use your engineering skills and experience to choose new input wisely and repeat.

This slide highlights some milestones in a process of model building and calibration.



Part 3 describes how to evaluate different architectures and workloads using model.

This part demonstrates the value of modeling approach for application sizing as it allows quick evaluation of multiple options of system set up.

What if locking is fixed?

Fixing locking in a model is very simple – just remove Affected Passive Resources

Frame Name: Copy 1 of Baseline | Frame 2 of 2

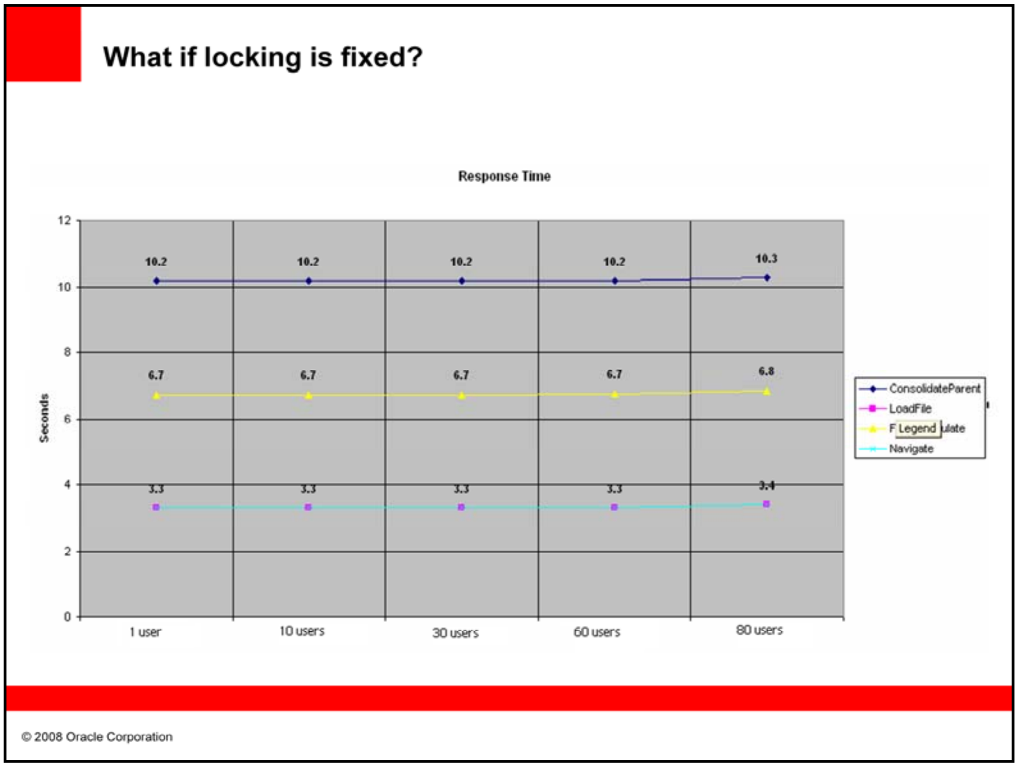
Systems | Active Resources | Workloads | Passive Resources

User Notes | **AR/WL Matrix** | Steps | PR/WL Matrix

	System Name	Workload	Active Resource	Visit Count	Service Required	Contribute to Response Time'	Affected Passive Resources	Fair Share
1	HFM FIAT	ConsolidateParent	HFM server	1.	9737.	yes		
2	HFM FIAT	ConsolidateParent	Workspace server	1.	1277.	yes		
3	HFM FIAT	ConsolidateParent	Think time	1.	180.	no		
4	HFM FIAT	ConsolidateParent	Controllers	1.	3890.	yes		
5	HFM FIAT	LoadFile	HFM server	1.	2512.	yes		
6	HFM FIAT	LoadFile	Workspace server	1.	187.	yes		
7	HFM FIAT	LoadFile	Think time	1.	180.	no		
8	HFM FIAT	LoadFile	Controllers	1.	1744.	yes		
9	HFM FIAT	ForceCalculate	HFM server	1.	6118.	yes		
10	HFM FIAT	ForceCalculate	Workspace server	1.	433.	yes		
11	HFM FIAT	ForceCalculate	Think time	1.	180.	no		
12	HFM FIAT	ForceCalculate	Controllers	1.	2994.	yes		
13	HFM FIAT	Navigate	HFM server	1.	2512.	yes		
14	HFM FIAT	Navigate	Workspace server	1.	187.	yes		
15	HFM FIAT	Navigate	Think time	1.	180.	no		
16	HFM FIAT	Navigate	Controllers	1.	1744.	yes		

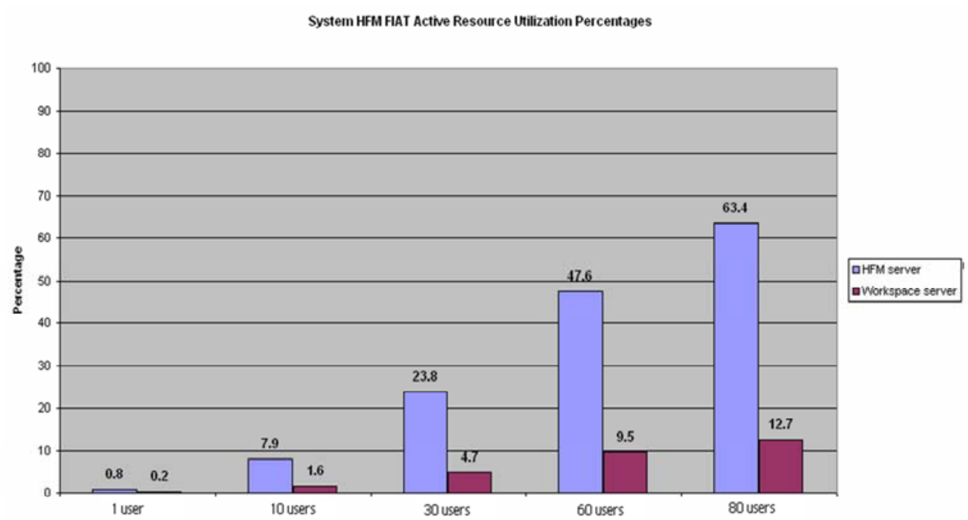
This is self explanatory – fixing locking in a model is simple – just remove Affected Passive Resources. After that we can solve model and see how good transactions look if they are not hitting a wall called “Database locking”.

To fix locking in real system is much more challenging, but model actually encourages to do that because it shows great positive impact of that action.



This is how well transactions perform after locking is eliminated. Great incentive for application designers to take care of software limitations!

What if locking is fixed?



And the server's utilization is in a normal range. Now we are well positioned to check if our system can support more users.

What if we have more concurrent users?

Introduced three steps with more concurrent users

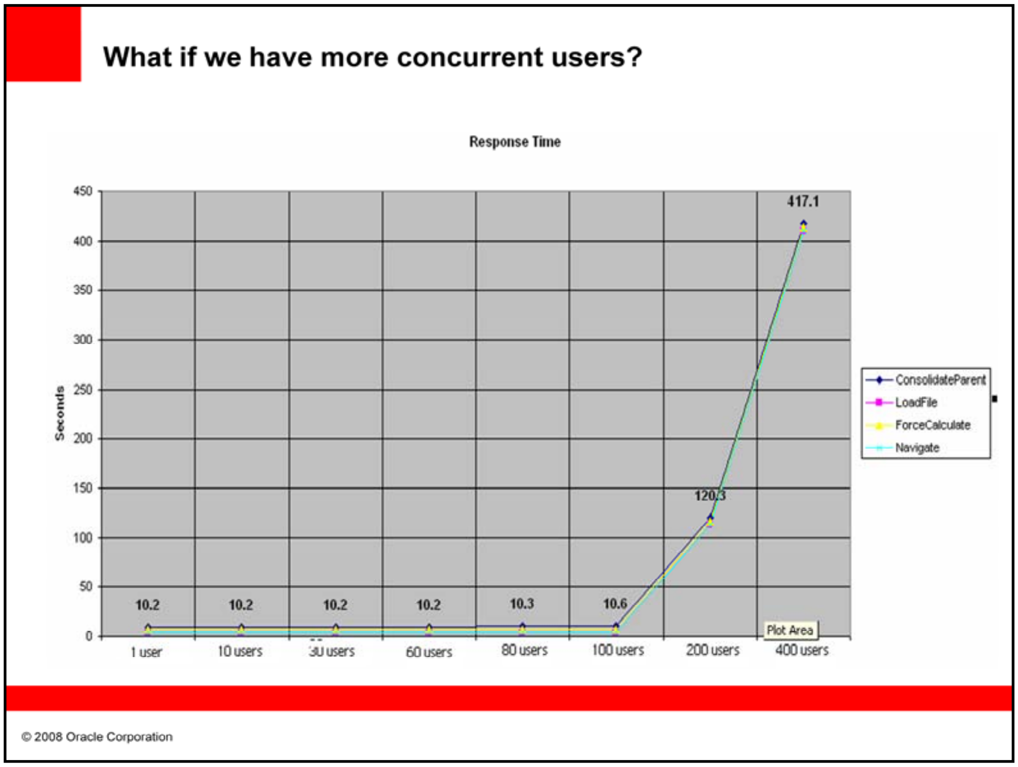
Frame Name: Copy 1 of Copy 1 of Baseline — Frame 3 of 3

Systems | Active Resources | Workloads | Passive Resources

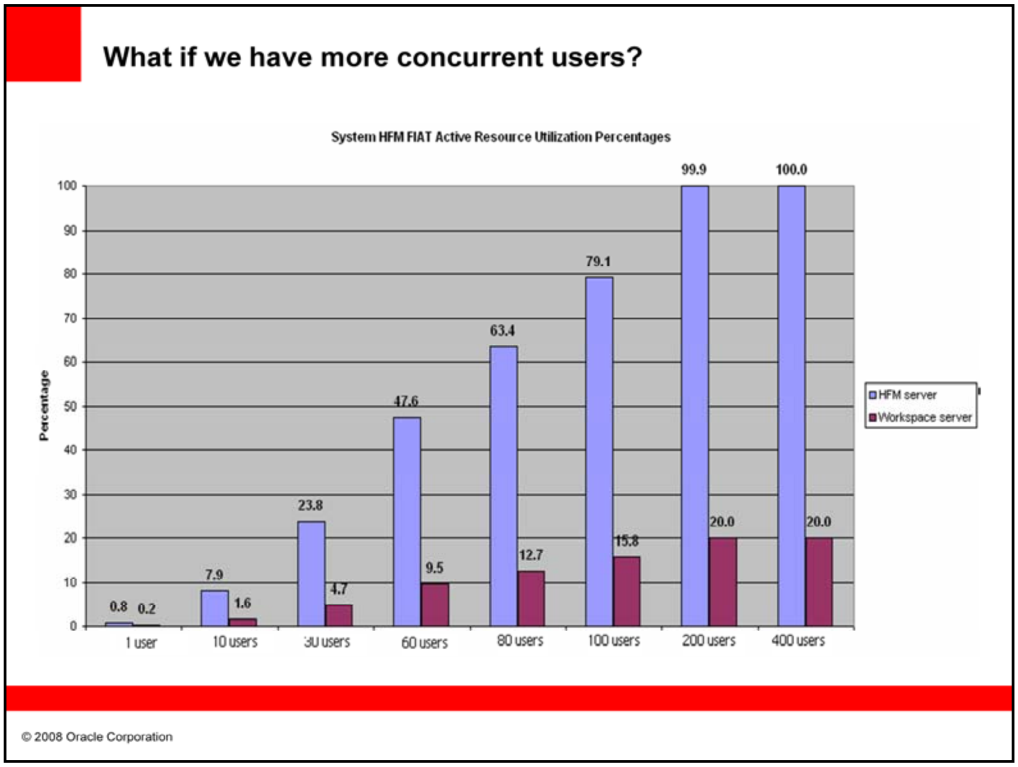
User Notes | AR/WL Matrix | **Steps** | PR/WL Matrix

	System Name	Workload	Workload Growth Type	Step: 2	Step: 3	Step: 4	Step: 5	Step: 6	Step: 7	Step: 8
1	HFM FIAT	ConsolidateParent	Population:	10.	30.	60.	80.	100.	200.	400.
2	HFM FIAT	LoadFile	Population:	10.	30.	60.	80.	100.	200.	400.
3	HFM FIAT	ForceCalculate	Population:	10.	30.	60.	80.	100.	200.	400.
4	HFM FIAT	Navigate	Population:	10.	30.	60.	80.	100.	200.	400.

Let’s try to increase a number of user to 100, 200, and 400.



We still have acceptable transaction time for 100 users, but the system cannot support more users than that.



The reason is – one of our servers reaches 100% of its capacity for 200 users. What can we do to still accommodate 200 users?

What if we deploy second HFM server?

Second HFM server is added with the same specs as the first one

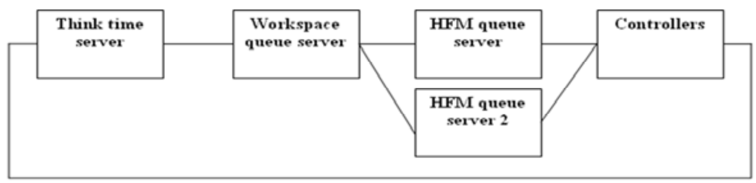
Model Title: HFM FIAT model

Frame Name: Copy 1 of Copy 1 of Copy 1 of Baseline | **Frame 4 of 4**

Systems | **Active Resources** | Workloads | Passive Resources

User Notes | AR/WL Matrix | Steps | PR/WL Matrix

	System Name	Active Resource	Equipment Name	Equipment Type	Discipline	Speed Factor	Number of Servers	Type	Path
1	HFM FIAT	HFM server	Intel Xeon 7150N 3.5GHz/1MB/16MB	CPU	PPRI	1756.33	8.	MULT	
2	HFM FIAT	HFM server 2	Intel Xeon 7150N 3.5GHz/1MB/16MB	CPU	PPRI	1756.33	8.	MULT	
3	HFM FIAT	Workspace server	Intel Xeon 5080 3.73GHz/2MB	CPU	PPRI	1749.85	4.	MULT	
4	HFM FIAT	Think time	THINK Queue		IS	1.	1.		
5	HFM FIAT	Controllers		Unknown Type	IS	1000.	1.		



Let’s try to add one more HFM server.

What if we deploy second HFM server?

Transaction to be processed in HFM_server_2 added to workload

Model Title: HFM FIAT model

Frame Name: Copy 1 of Copy 1 of Copy 1 of Baseline — **Frame 4 of 4**

Systems | Active Resources | **Workloads** | Passive Resources

User Notes | AR/WL Matrix | Steps | PR/WL Matrix

	System Name	Workload	Type	Measured Throughput	Throughput Adjustment Active Resource	Environment
1	HFM FIAT	ConsolidateParent	CLOSED	1.	Think time	INTERACTIVE
2	HFM FIAT	LoadFile	CLOSED	1.	Think time	INTERACTIVE
3	HFM FIAT	ForceCalculate	CLOSED	1.	Think time	INTERACTIVE
4	HFM FIAT	Navigate	CLOSED	1.	Think time	INTERACTIVE
5	HFM FIAT	ConsolidateParent_2	CLOSED	1.	Think time	INTERACTIVE
6	HFM FIAT	LoadFile_2	CLOSED	1.	Think time	INTERACTIVE
7	HFM FIAT	ForceCalculate_2	CLOSED	1.	Think time	INTERACTIVE
8	HFM FIAT	Navigate_2	CLOSED	1.	Think time	INTERACTIVE

We have to distribute evenly workload between two HFM servers.

What if we deploy second HFM server?

We have to take into account that a number of users serving in each HFM server is two times smaller now, but a total number of users is still the same

Model Title: HFM FIAT model

Frame Name: Copy 1 of Copy 1 of Baseline — **Frame 4 of 4**

Systems | Active Resources | Workloads | Passive Resources

User Notes | AR/WL Matrix | **Steps** | PR/WL Matrix

	System Name	Workload	Workload Growth Type	Step: 1	Step: 2	Step: 3	Step: 4	Step: 5	Step: 6	Step: 7	Step: 8
1	HFM FIAT	ConsolidateParent	Population:	1.	5.	15.	30.	40.	50.	100.	200.
2	HFM FIAT	LoadFile	Population:	1.	5.	15.	30.	40.	50.	100.	200.
3	HFM FIAT	ForceCalculate	Population:	1.	5.	15.	30.	40.	50.	100.	200.
4	HFM FIAT	Navigate	Population:	1.	5.	15.	30.	40.	50.	100.	200.
5	HFM FIAT	ConsolidateParent_2	Population:	1.	5.	15.	30.	40.	50.	100.	200.
6	HFM FIAT	LoadFile_2	Population:	1.	5.	15.	30.	40.	50.	100.	200.
7	HFM FIAT	ForceCalculate_2	Population:	1.	5.	15.	30.	40.	50.	100.	200.
8	HFM FIAT	Navigate_2	Population:	1.	5.	15.	30.	40.	50.	100.	200.

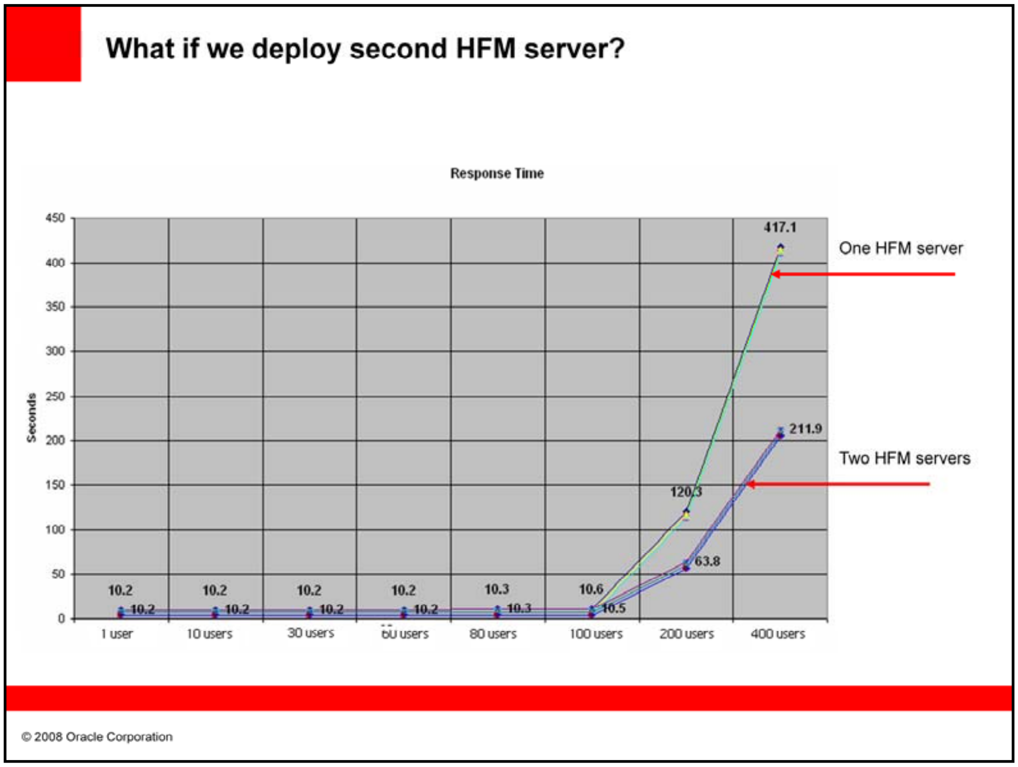
We replicated all transactions – one group of transactions is served in one server, and second group is served in second server. We have to make sure that a number of users hitting each server is two times lover that a total number of users.

What if we deploy second HFM server?

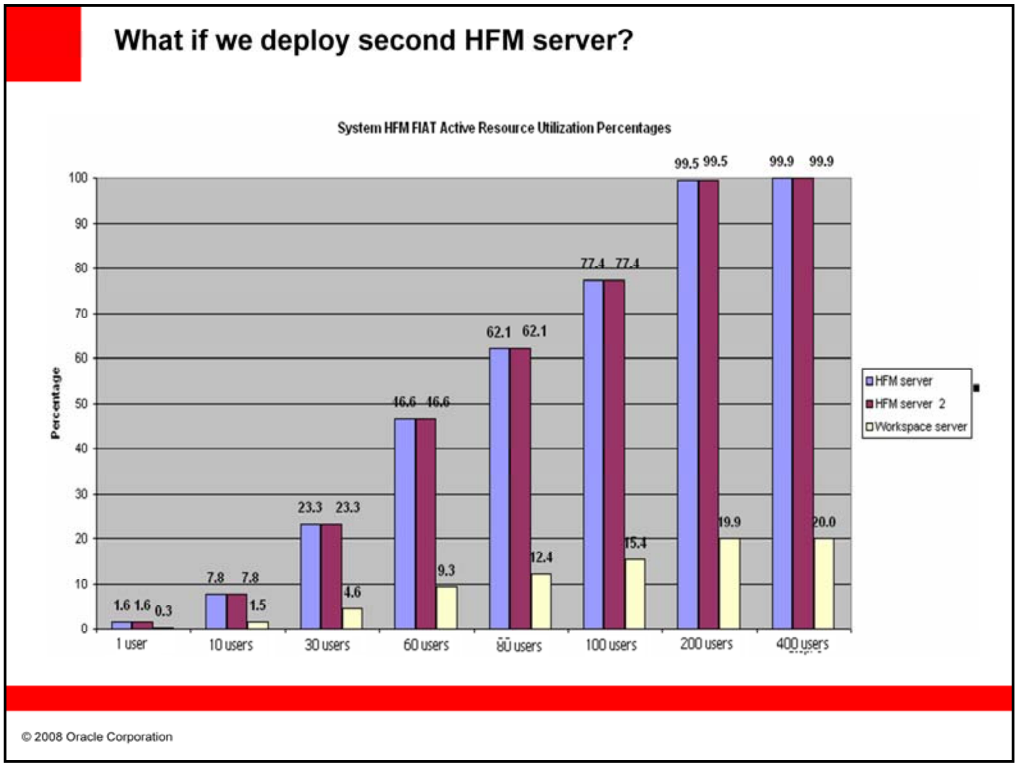
Resources/Workload matrix for system with two HFM servers

	System Name	Workload	Active Resource	Visit Count	Service Required	Contribute to Response Time'	Affected Passive Resource
1	HFM FIAT	ConsolidateParent	HFM server	1.	9737.	yes	
2	HFM FIAT	ConsolidateParent	HFM server 2	1.	9737.	no	
3	HFM FIAT	ConsolidateParent	Workspace server	1.	1277.	yes	
4	HFM FIAT	ConsolidateParent	Think time	1.	180.	no	
5	HFM FIAT	ConsolidateParent	Controllers	1.	3890.	yes	
6	HFM FIAT	LoadFile	HFM server	1.	2512.	yes	
7	HFM FIAT	LoadFile	HFM server 2	1.	2512.	no	
8	HFM FIAT	LoadFile	Workspace server	1.	187.	yes	
9	HFM FIAT	LoadFile	Think time	1.	180.	no	
10	HFM FIAT	LoadFile	Controllers	1.	1744.	yes	
11	HFM FIAT	ForceCalculate	HFM server	1.	6118.	yes	
12	HFM FIAT	ForceCalculate	HFM server 2	1.	6118.	no	
13	HFM FIAT	ForceCalculate	Workspace server	1.	433.	yes	
14	HFM FIAT	ForceCalculate	Think time	1.	180.	no	
15	HFM FIAT	ForceCalculate	Controllers	1.	2994.	yes	
16	HFM FIAT	Navigate	HFM server	1.	2512.	yes	
17	HFM FIAT	Navigate	HFM server 2	1.	2512.	no	
18	HFM FIAT	Navigate	Workspace server	1.	187.	yes	
19	HFM FIAT	Navigate	Think time	1.	180.	no	
20	HFM FIAT	Navigate	Controllers	1.	1744.	yes	
21	HFM FIAT	ConsolidateParent_2	HFM server	1.	9737.	no	
22	HFM FIAT	ConsolidateParent_2	HFM server 2	1.	9737.	yes	
23	HFM FIAT	ConsolidateParent_2	Workspace server	1.	1277.	yes	
24	HFM FIAT	ConsolidateParent_2	Think time	1.	180.	no	
25	HFM FIAT	ConsolidateParent_2	Controllers	1.	3890.	yes	
26	HFM FIAT	LoadFile_2	HFM server	1.	2512.	no	
27	HFM FIAT	LoadFile_2	HFM server 2	1.	2512.	yes	
28	HFM FIAT	LoadFile_2	Workspace server	1.	187.	yes	
29	HFM FIAT	LoadFile_2	Think time	1.	180.	no	
30	HFM FIAT	LoadFile_2	Controllers	1.	1744.	yes	
31	HFM FIAT	ForceCalculate_2	HFM server	1.	6118.	no	
32	HFM FIAT	ForceCalculate_2	HFM server 2	1.	6118.	yes	
33	HFM FIAT	ForceCalculate_2	Workspace server	1.	433.	yes	
34	HFM FIAT	ForceCalculate_2	Think time	1.	180.	no	
35	HFM FIAT	ForceCalculate_2	Controllers	1.	2994.	yes	
36	HFM FIAT	Navigate_2	HFM server	1.	2512.	no	
37	HFM FIAT	Navigate_2	HFM server 2	1.	2512.	yes	
38	HFM FIAT	Navigate_2	Workspace server	1.	187.	yes	
39	HFM FIAT	Navigate_2	Think time	1.	180.	no	
40	HFM FIAT	Navigate_2	Controllers	1.	1744.	yes	

This is the task of describing how transactions travels across model.



And now we can solve the model and see that two HFM servers still do not deliver transaction times we are looking for.



The reason – still bottleneck on HFM servers.

What if we add eight more CPUs to each HFM server?

Include 16-CPU servers into model

Model Title: HFM FIAT model

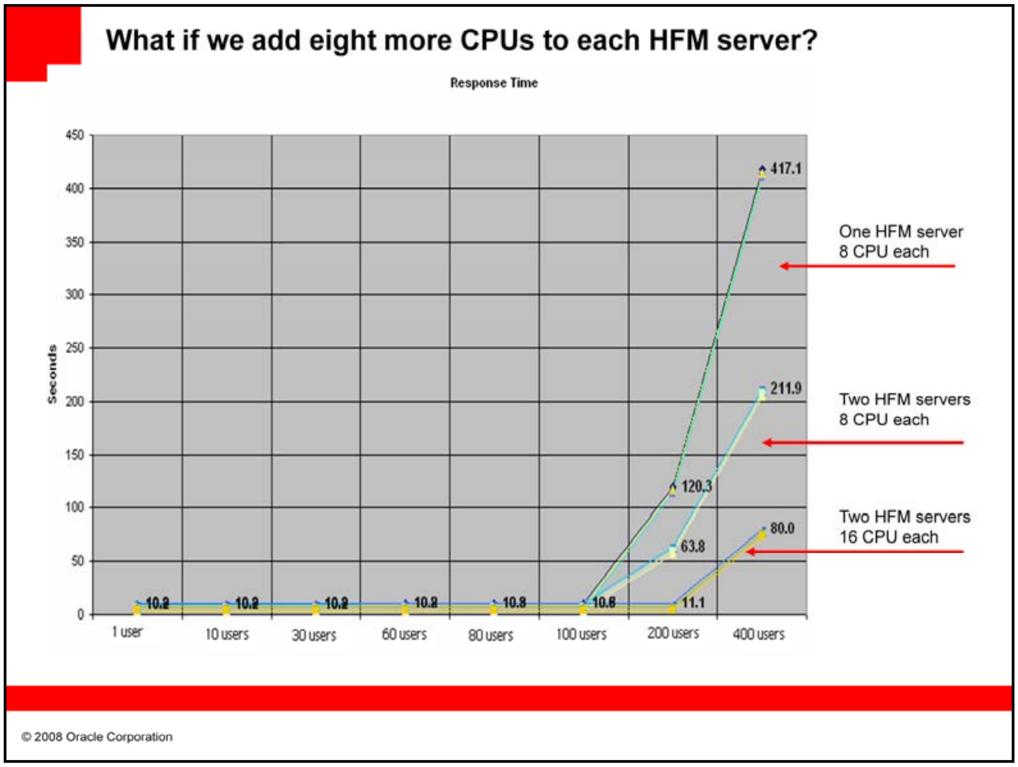
Frame Name: Copy 1 of Copy 1 of Copy 1 of Baseline — **Frame 5 of 5**

Systems | **Active Resources** | Workloads | Passive Resources

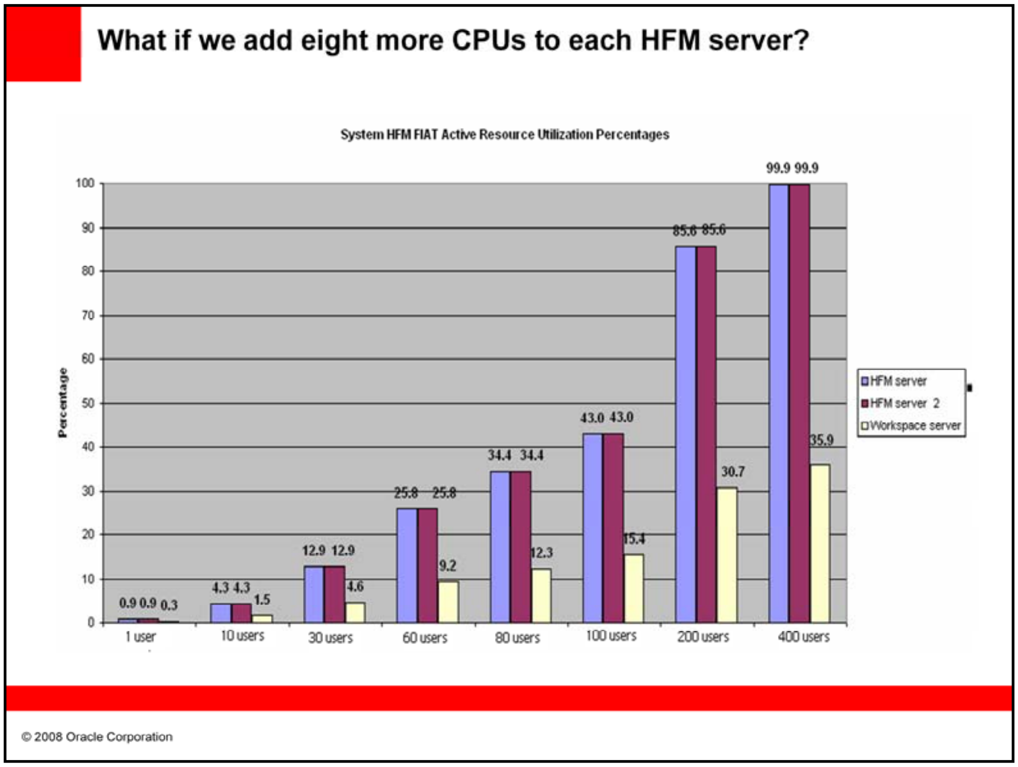
User Notes | AR/WL Matrix | Steps | PR/WL Matrix

	System Name	Active Resource	Equipment Name	Equipment Type	Discipline	Speed Factor	Number of Servers	Type	Path
1	HFM FIAT	HFM server	Intel Xeon 7150N 3.5GHz/1MB/16MB /Leonid/	CPU	PPRI	1580.7	16.	MULT	
2	HFM FIAT	HFM server 2	Intel Xeon 7150N 3.5GHz/1MB/16MB /Leonid/	CPU	PPRI	1580.7	16.	MULT	
3	HFM FIAT	Workspace server	Intel Xeon 5080 3.73GHz/2MB	CPU	PPRI	1749.85	4.	MULT	
4	HFM FIAT	Think time	THINK Queue		IS	1.	1.		
5	HFM FIAT	Controllers		Unknown Type	IS	1000.	1.		

OK, we invested in servers.



And our investment pays back – system delivers acceptable response time now for 200 users!



Servers have some extra capacity for 200 users, but are maxed out for 400 users.

Lessons learned

- Model is an extension but not a substitute to your experience and gut feelings – use both to make sure model projections are right. If there is a conflict between model prediction and your senses revisit both until they are in concert.
- Model predictions are only as good as input data. Go extra mile to make sure you feed model with data you can trust. Using load testing increases input data quality.
- Calibrate model! As more calibration points, as higher your confidence in model accuracy.
- Queuing network models are capable to factor in not only hardware, but also application constrains like number of threads, database connections, data locks etc.
- Solving model is a breeze – do not limit a number of what-if scenarios you evaluate. You might come up with architecture which saves a lot of money.

Conclusions

- Presented methodology of multi-tiered applications sizing using load testing and queuing network models
- Load testing collects input data for model as well as data for model calibration
- Queuing models predict transaction response times as well as server utilizations
- Methodology can be used to evaluate different what-if scenarios (different number of servers, CPUs, different system architectures etc.)

References

- Edward D. Lazowska. Quantitative System Performance, Computer System Analysis Using Queuing Network Models. Prentice Hall, 1984.
- Neil J Gunter. Guerilla Capacity Planning. Springer-Verlag New York, LLC , 2005, ISBN-13: 9783540261384
- Performance testing guidance for Web applications. Microsoft Corporation, 2007

Q & A