Environmental Systems Research Institute, Inc., 380 New York St., Redlands, CA 92373-8100 USA • TEL 909-793-2853 • FAX 909-307-3014

Capacity Planning and Performance Benchmark Reference Guide

v. 1.8

Prepared by:

ESRI Professional Services
Enterprise Implementation Services Team
Redlands, California

July 12, 2009

1 Objective

The objective of this document is to provide guidance for understanding and applying the performance benchmarks published in the Implementation Gallery site. This document describes each section of a typical published benchmark and provides additional information regarding methodology and definitions.

2 Performance Benchmark Usage

The ESRI enterprise testing team conducts and publishes performance benchmarks of representative test results in the Implementation Gallery. Understanding these performance benchmarks can help with,

- System sizing
- Capacity planning model calibration and input
- Selection of optimal technology architectures
- Selection of optimal application architecture

Without relevant performance benchmarks, it may be challenging or impossible to conduct effective capacity planning. In many cases, estimating capacity with no test data or using outdated capacity models can lead to increased potential error when planning a solution.
Performance benchmarks provide only a short description and summary of key results. To apply these reports for capacity planning of different systems, refer to the information contained in this document or request support from ESRI Professional Services.

3 Benchmark Information Type

Found in the Implementation Gallery, a performance benchmark report typically contains key test result information and focuses on a single test. In many cases, ESRI conducts several variations of a similar test by changing only one configuration parameter at a time. For example, benchmarking an ArcGIS Server map service may involve varying data source type (e.g., file geodatabase, shapefile, or ArcSDE) or image compression type (e.g., JPEG, PNG 8, PNG 24). Since the tested application and methodology remain the same, variation results are grouped as supplemental information within the performance benchmark report.

3.1 Benchmark Results

See Walk-through of Benchmark Results section that describes the information presented within a typical benchmark report in more detail.

The key part of a performance benchmark report its Capacity Planning section, which provides important information for sizing estimates, including:

- CPU service time
- Network Mbits/transaction
- Machine SpecRate (for extrapolation of results for different hardware)
- Maximum throughput
- Response time for a given user load

This information should be used as an input for capacity planning tools. The benchmark package includes a simple capacity calculator, described in the next section.

3.2 Capacity Calculator

Along with the performance benchmark report, a simple capacity calculator is provided to help calculate the required number of CPUs and network bandwidth under specified user load, service time, SpecRate etc. The calculator is an Excel spreadsheet as shown in the following figure.

Parameter	Value	Unit
RT_t	2.80	sec
$Users_t$	51.00	users
Think	6.00	sec
$ST_{ArcSOC\ Zoom\ b}$	0.60	sec
$Mbits/tr_b$	1.81	Mbits/tr
$SpecRate_bPerCPU$	13.425	
$SpecRate_tPerCPU$	13.425	
%CPU	95	%
TH_t	20,864	tr/hr
$\#CPU_t$	**3.63**	**CPU cores**
Mbps	10.51	Mbps

input values
calculated values

The calculations are based on the capacity planning formulas as described in Fundamental Laws, Formulas and Definition section. This tool can be used for capacity planning the following:

- Estimating number of CPU cores required to support a defined user load
- Estimating required bandwidth to support a defined user load
- Estimate throughput based on a defined user load

- Validating test results through comparison with Capacity Calculator values

3.3 Benchmark and Modeled Data

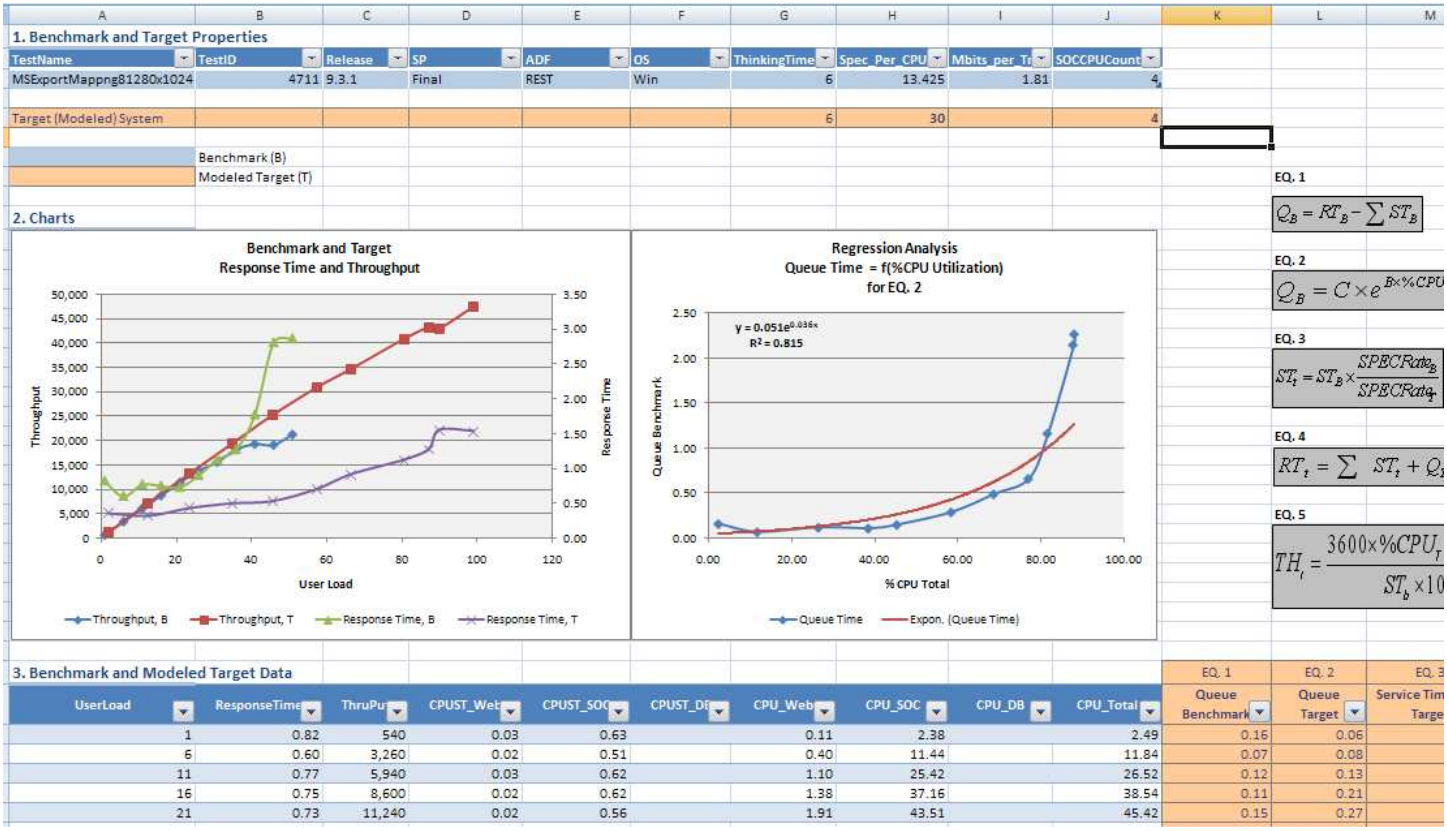
As discussed in the previous section, performance benchmark results can be used to estimate maximum capacity of a system using the Capacity Calculator. However, to better analyze the performance and scalability profile of the benchmark, "raw" benchmark, modeled data and comparison charts are provided. This information allows an analysis of the entire user load range, not just a certain point of system capacity, as is the case using the Capacity Calculator.

Beyond this, "raw" benchmark data can be used for modeling purposes by altering specific parameters. This allows an investigation of the impact of changes such as,

1. What happens if a faster CPU is used?
2. What happens if more CPU cores are added?
3. What happens if transaction rate varies (think time)?

The figure below shows the benchmark and modeled data worksheet.

Figure 1: Benchmark and Modeled Data Worksheet



1. Benchmark and Target Properties

TestName	TestID	Release	SP	ADF	OS	ThinkingTime	Spec_Per_CPU	Mbits_per_Tr	SOCCPUCount
MSExportMappng81280x1024	4711	9.3.1	Final	REST	Win	6	13.425	1.81	4
Target (Modeled) System						6	30		4

Benchmark (B)
Modeled Target (T)

2. Charts

EQ. 1 $Q_B = RT_B - \sum ST_B$

EQ. 2 $Q_B = C \times e^{B \times \%CPU}$

EQ. 3 $ST_t = ST_B \times \frac{SPECRate_B}{SPECRate_t}$

EQ. 4 $RT_t = \sum ST_t + Q_t$

EQ. 5 $TH_t = \frac{3600 \times \%CPU_t}{ST_b \times 10}$

3. Benchmark and Modeled Target Data

UserLoad	ResponseTime	ThruPut	CPUST_Web	CPUST_SOC	CPUST_DB	CPU_Web	CPU_SOC	CPU_DB	CPU_Total	EQ. 1 Queue Benchmark	EQ. 2 Queue Target	EQ. 3 Service Time Target
1	0.82	540	0.03	0.63		0.11	2.38		2.49	0.16	0.06	
6	0.60	3,260	0.02	0.51		0.40	11.44		11.84	0.07	0.08	
11	0.77	5,940	0.03	0.62		1.10	25.42		26.52	0.12	0.13	
16	0.75	8,600	0.02	0.62		1.38	37.16		38.54	0.11	0.21	
21	0.73	11,240	0.02	0.56		1.91	43.51		45.42	0.15	0.27	

3.4 Related Information

In addition to those items discussed abovem, the following related information may be included, if feasible, in a performance benchmark package:

- Data and related ArcMap document
- Test script and load test project (typically Visual Studio Test Team edition project)

4 Estimating Capacity Using Benchmark Data

Without relevant performance benchmarks, it might be challenging or impossible to conduct an effective capacity analysis. In many cases, "ball-parking" or "rationalizing" leads to increased potential error that can jeopardize the success of the project.

This section describes how to effectively utilize the performance benchmarks published in the Implementation Gallery site and how to model a different system.

4.1 Fundamental Laws, Formulas and Definitions

Although it is possible to effectively use the Capacity Calculator without a thorough understanding of the capacity planning formulas, being familiar with these principles will help estimating the potential error, validating the analysis or customizing the tool to specific needs.

Capacity planning expressions are as follows:

Equation 1: CPU Service Time

$$ST = \frac{\#CPU \times 3600 \times \%CPU}{TH \times 100}$$

Equation 2: Relative CPU Service Time

$$ST_t = ST_b \times \frac{SpecRatePerCPU_b}{SpecRatePerCPU_t}$$

Equation 3: Response Time as a function of CPU Service Time and Queue Time

$$RT = \sum ST + Q$$

Equation 4: CPU Sizing

$$\#CPU_t = \frac{ST_b \times TH_t \times 100}{3600 \times \%CPU_t} \times \frac{SpecRatePerCPU_b}{SpecRatePerCPU_t}$$

Equation 5: Throughput as a function of concurrent users and operation frequency (Response Time + Think Time)

$$TH_t = \frac{Users_t \times 3600}{RT_t + Think_t}$$

Equation 6: Throughput as a function of CPU service time and number of cores

$$TH_t = \frac{3600 \times \%CPU_t \times \#CPU_t}{ST_b \times 100} \times \frac{SpecRatePerCPU_t}{SpecRatePerCPU_b}$$

Equation 7: Network Sizing

$$Mbps_t = \frac{TH_t \times Mbitsptr_b}{3600}$$

Where,

- ST - Service time
- RT - Response time
- Q - Queue time

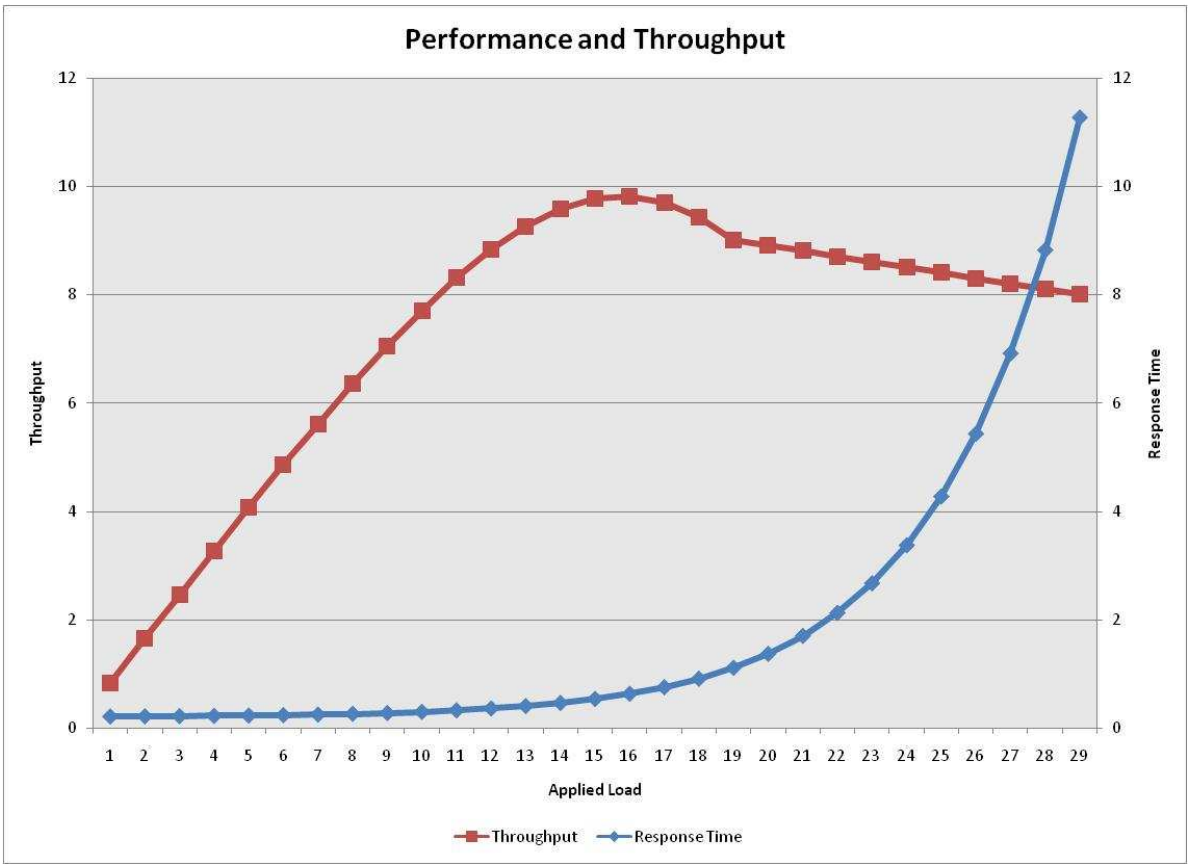
- #CPU - Number of CPU cores
- %CPU - Percentage of CPU utilization (Typically a target threshold is set between 85% and 95%.)
- TH - Maximum throughput
- Think - Assumed think time (sec.) between transactions
- Users - User load
- b (subscripted)—Benchmarked inputs
- t (subscripted)—Target outputs
- SpecRate—SpecRate CINT 2006 benchmarked system (See http://www.spec.org/cpu2006/results/cint2006.html.)

4.1.1 Understanding System Capacity

Typically, a graphical relationship may be established between user load or CPU utilization and response time and throughput. To understand this relationship and determine capacity of a system, the following graph may be composed:

- horizontal axis: user load or CPU Utilization
- vertical axis: Response Time and Throughput

As we can use, when the system has higher utilization (e.g. at 40-50%), the response time start increasing due to increasing queue time. It should be noted once the maximum throughput is reached, adding more load might actually degrade throughput.



The capacity of the system may be defined as one of the following:

- Maximum throughput.
- Desired response time, e.g. as per Level of Service agreement
- Desired system utilization.
- At first error.

4.1.2 Understanding CPU Service time

CPU service time is a measure of the time required for the CPU to process a request or transaction, and is an important input for capacity planning. As a general rule, this service time should remain fairly constant, even as increased load is applied. Performance benchmarks typically include the CPU service time for each tier.

In order to calculate service time for a particular tier, its CPU utilization must be known. Sometimes a benchmark is conducted with each tier on a separate physical machine, allowing utilization to be easily found. However, many times a benchmark is conducted with all tiers on one machine (known as a "workgroup" configuration). In this case, each tier's utilization may be calculated by summing the CPU utilization of each associated process. The following are typical processes encountered:

- Web: w3wp.exe, lsass.exe
- ArcGIS Server: arcsom.exe, arcsoc.exe
- ArcGIS Image Server: ESRIImageServiceProvider.exe
- Database: oracle.exe, sqlserver.exe

Detailed process information is listed in each performance benchmark's appendix.

4.1.3 Understanding Queue Time

Queue time is a measure of the time a request or transaction spent waiting to be processed, and is an important input for capacity planning. As a general rule, queue time will increase as load and CPU utilization increase, although it is sometimes considered negligible a lower utilization levels (less than 50%).

There are several queuing theory models available, such as M/M/n. In many cases these generic models provide adequate values for capacity planning, especially when the system is primarily CPU bound. The advantage of performance benchmark data is that a custom model may be developed for each benchmark using regression analysis (a trend line). Typically, an exponential trend/regression provides the best fit, as determined by the coefficient of determination (R^2).

4.1.4 Helpful Resources

- Fundamental Laws
  http://www.cs.washington.edu/homes/lazowska/qsp/Images/Chap_03.pdf

- Little's Law
  http://en.wikipedia.org/wiki/Little_law#Use_in_performance_testing_of_computer_systems

4.2 Throughput and Concurrent Users Variance

Understanding throughput, average concurrent users, and peak concurrent users on an existing environment is important when beginning to plan for capacity. In many cases average concurrent users may be a very small fraction of the peak concurrent users; however it could just as easily be much higher. In many cases it is difficult to estimate these values, but a good place to start is by,

- Monitoring the existing system usage
- Analyzing proposed business processes and extrapolate average concurrent users from peak number of users.

For example, via web server logs an analysis could be conducted each hour to understand usage trends. However, if only a total monthly usage report is available, an average usage could be calculated by dividing by of the month's operational hours, (e.g. 8 hours a day, 5 days a week).

To calculate throughput knowing the concurrent number of users and frequency of requests or transactions, use the throughput equation from the previous section. For example, if users perform an operation approximately every 10 seconds (e.g. 1 sec operation and 9 seconds think time), this would equate to throughput 1*3600/10= 360 operations/hour.

Some mission critical applications may have high performance and availability requirements and should be designed for peak concurrent users, however this may increase hardware and license costs. For solutions that have short peak usage durations or if temporary performance degradation is acceptable, these costs may be reduced by designing for average concurrent usage.

If the throughput cannot be extrapolated from the existing usage, it can be estimated using the Throughput equation from the previous section. The key input values are concurrent users and frequency of operations (think time) as shown (highlighted in green) in the following figures for benchmark modeled data and the capacity calculator.

	Think Time	Spec_Per_CPU	Mbits_per_Tr	SOCCPUCount
Benchmark (B)	6	13.425	1.81	4
Target (T) or Modeled	6	**30**	**1.81**	**4**

Parameter	Value	Unit
RT_t	4.00	Sec
$Users_t$	1.00	users
Think	6.00	Sec
$ST_{ArcSOC\ Zoom\ b}$	0.60	Sec
$Mbits/tr_b$	1.81	Mbits/tr
$SpecRate_bPerCPU$	13.425	
$SpecRate_tPerCPU$	13.425	
%CPU	95	%
TH_t	360	tr/hr
$\#CPU_t$	**0.06**	**CPU cores**
Mbps	0.18	Mbps

4.3 Workflow Variance

Prior to applying the published ESRI benchmark results, it is important to understand differences between the current workflow and the performance benchmark so a proper model can be derived.

Determining all details of the workflow during the early planning phase may be difficult. However, for capacity planning purposes the focus should primarily be on:

- Transactions performed frequently with a relatively long response time
- Batch operations—transactions with a long response time

For example, a workflow may be composed of:

- Transaction 1:
  - Zoom against dynamic service.
  - Think 6 seconds.
- Transaction 2:
  - Zoom against dynamic service.
  - Think 6 seconds.
- Transaction 3
  - Query against small dataset.
  - Think 6 seconds.
- Transaction 4
  - Zoom 3 against dynamic service.
  - Think 6 seconds.
- Transaction 5
  - Print against the dynamic service.

For capacity planning purposes, transactions performed infrequently and have comparatively short response times may be excluded. In the above example, the query transaction against a small dataset might be just a fraction of a zoom operation against the dynamic service. Therefore, the focus should be on the zoom and print transactions only.

Once a workflow is defined for capacity planning, review the published benchmarks and select the relevant Performance Benchmark report (similar application and workflow) from the Implementation Gallery.

Even excluding insignificant transactions, it is possible the workflow will be more complex than one particular performance benchmark, and several may be required. Gathering the CPU service times for the various transactions from the respective benchmarks, the complex workflow may be pieced together. This is discussed in the CPU Service Time section.

4.4 Transaction "Size" Variance

As mentioned in the Workflow Variance section, transactions define the core units of work in a workflow. Each transaction will have a different size, which can impact how much network bandwidth is required to suitably transport
Transaction "size" typically is related to the following; each can have a significant impact:

- Data sources
- ArcMap documents
- Number of features and layers returned (e.g., zoom, search)
- Image compression
- Map sizes
- Caching

To extrapolate from published benchmark results, review and account for potential performance factor differences as described in Performance and Scalability section.

Once the final size has been understood, it may be adjusted within the modeled capacity result or as shown below in the sample Capacity Calculator.

RT_t	4.00	sec
$Users_t$	1.00	users
Think	6.00	sec
$ST_{ArcSOC\ Zoom\ b}$	0.60	sec
$Mbits/tr_b$	1.81	Mbits/tr
$SpecRate_bPerCPU$	13.425	
$SpecRate_tPerCPU$	13.425	
%CPU	95	%
TH_t	360	tr/hr
$\#CPU_t$	**0.06**	**CPU cores**
Mbps	0.18	Mbps

4.5 Adjusting for CPU Speed Relative Difference

This difference can be measured as a ratio of SpecRate CINT 2006 base per core as per http://www.spec.org/cpu2006/results/cint2006.html published benchmarks. It should be noted that under a light uder load,the CPU speed, not the number of cores, is the primary contributor to system performance (measured as a response time).

For example, a typical ESRI benchmark CPU is

- PowerEdge 1950 (Intel Xeon processor 5160, 3.00 GHz), SpecRate/Core = 53.7/4 = **13.425**
- PowerEdge 1950 III (Intel Xeon E5450, 3.00 GHz), SpecRate/Core = 109/8 = **13.625**

The following table is an excerpt from a published SpecRate CINT 2006 benchmarked system. See http://www.spec.org/cpu2006/results/cint2006.html.

Hardware Vendor	System	Result	Baseline	# Cores
Dell Inc.	PowerEdge 1950 (Intel Xeon 5140, 2.33 GHz)	0	45.9	4
Dell Inc.	PowerEdge 1950 (Intel Xeon 5150, 2.66 GHz)	0	49.8	4
Dell Inc.	PowerEdge 1950 (Intel Xeon 5160, 3.00 GHz)	0	53.7	4
Dell Inc.	PowerEdge 1950 (Intel Xeon E5335, 2.00 GHz)	0	69	8
Dell Inc.	PowerEdge 1950 III (Intel Xeon E5410, 2.33 GHz)	112	92.3	8
Dell Inc.	PowerEdge 1950 III (Intel Xeon E5420, 2.50 GHz)	112	98	8
Dell Inc.	PowerEdge 1950 III (Intel Xeon E5430, 2.66 GHz)	124	100	8
Dell Inc.	PowerEdge 1950 III (Intel Xeon E5450, 3.00 GHz)	125	109	8

If all other factors are constant, for a CPU-bound solution, the CPU service time measured from one system can be extrapolated using Relative CPU Service Time equation in the previous section.

In our case, the ratio is 13.425/13.625 = 0.985. It can be concluded that even though different server types are used, the relative CPU difference is marginal.

Users are often interested in relative performance (response time difference) between two systems. In case of low CPU utilization(less than 50%) queue time can be considered negligible. However, for a CPU- bound system response time (RT) can be estimated from a slower to a faster system using the following formula:

$$RT_t \approx RT_b \times \frac{SpecRatePerCPU_b}{SpecRatePerCPU_t}$$

4.6 Example 1: Estimating required number of CPU cores

A hosting company needs to design a system to support 50 concurrent users requesting maps. How many CPU cores are required to support this load?

Solution:

Step 1: Determine the SpecRate of the proposed server using http://www.spec.org/cpu2006/results/cint2006.html, as described in Adjusting for CPU Speed Relative Difference section.

IF there is no information about the type of CPU, assume the latest CPU is used. The spec rates were determined as **Spec_Per_CPU=30.**

Step 2: Adjust for throughput. Since there is no information available about frequency of transactions, throughput is estimated at 6 seconds think time and 1 second response time.

Step 3: Select the relevant Performance Benchmark report (similar application) from the Implementation Gallery. For input, use information in the Capacity Planning Model Input, including spec rate and CPU service time.

Step 4: Adjust for transaction size. IF there is no information about transaction type or size, assume the benchmarked CPU service time.

Step 5: Estimate capacity of the proposed system using Capacity Calculator

RT_t	1.00	Sec
$Users_t$	50.00	Users
Think	6.00	Sec
$ST_{ArcSOC\ Zoom\ b}$	0.60	Sec
$Mbits/tr_b$	1.81	Mbits/tr
$SpecRate_bPerCPU$	13.425	

$SpecRate_tPerCPU$	13.425	
%CPU	95	%
TH_t	25,714	tr/hr
$\#CPU_t$	**4.48**	**CPU cores**
Mbps	12.96	Mbps

Solution:
Approximately 5 CPU cores are required to support the given load. The current servers are sold typically with 4 or 8 cores. Therefore, to account for future growth it is recommended to use 8-core server.

4.7 Example 2: Using faster CPU

A hosting company charges $ 0.01 per map request with the performance level of service less than 1 second. The company has more potential customers, however the current system based on 4 core 3 GHz CPU is reaching the capacity and users report a degraded performance, with a map response time frequently above 2 seconds. An IT manager considers upgrading the existing server to a newer one. He estimates the total cost of upgrading hardware is $30,000 How should he estimate the return on investment (ROI)?

Solution:

Step 1: Determine the SpecRate of the existing and proposed (modeled) environment using http://www.spec.org/cpu2006/results/cint2006.html as described in Adjusting for CPU Speed Relative Difference section.

The spec rates were determined as **Spec_Per_CPU = 13.425** for the existing system and **Spec_Per_CPU = 30** for the proposed system.

Step 2: Select relevant Performance Benchmark report and data from Implementation Gallery. Ensure the benchmark and design applications are similar.

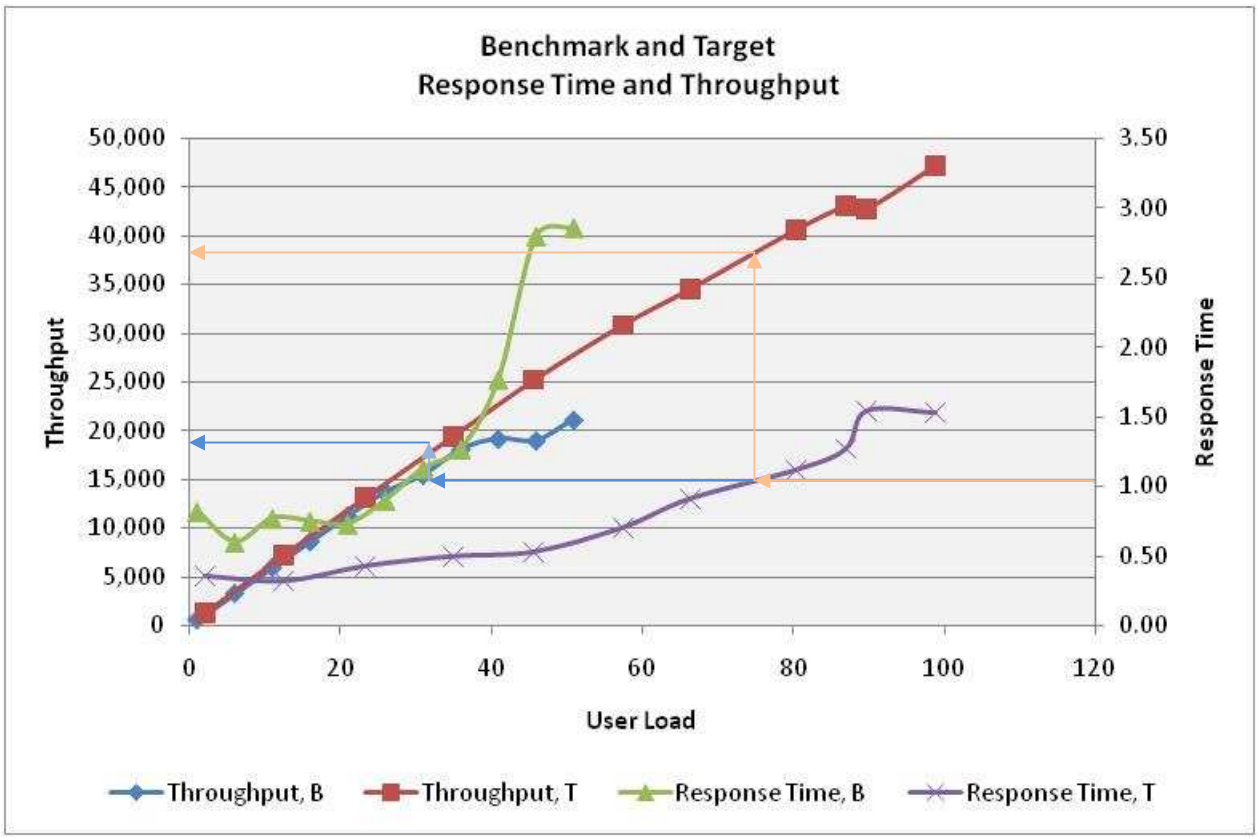
Step 3: Model the existing and proposed system using benchmark data. Determine the maximum throughput corresponding to contracted level of service (response time=1 sec).

> Note, we are not evaluating the number of concurrent users, but the throughput. For this analysis, we can assume any think time and number of concurrent users. For the purpose of analysis we will use the benchmark think time of 6 seconds.

Also, the benchmark and the existing system are the same. If not both system would need to be modeled.

We input Spec_Per_CPU=30 in this worksheet and analyze the result.

	ThinkingTime	Spec_Per_CPU	Mbits_per_Tr	SOCCPUCount
Benchmark (B)	6	13.425	1.81	4
Target (T) or Modeled	6	**30**		**4**



We determine the existing throughput corresponding to the required response time by drawing a horizontal line (blue) from 1 sec Response Time and intersecting with the Throughput curve (blue). We can estimate the existing throughput at **19,000 map/hour**.

We repeat the same steps (pink line) and estimate the new throughput at **38,000 map/hour** at an average response time less than 1 second for the targeted server

Note, for more precision, we can use *Equation 6: Throughput as a function of CPU service time and number of cores* or benchmark data to calculate the throughput. Since the system has to deliver 1 sec response time, we should first determine the corresponding CPU utilization from benchmark data as shown below (we can assume 60%).

3. Benchmark and Modeled Target Data

UserLoad	ResponseTime	ThruPut	CPUST_Web	CPUST_SOC	CPUST_DB	CPU_Web	CPU_SOC	CPU_DB	CPU_Total
1	0.82	540	0.03	0.63		0.11	2.38		2.49
6	0.60	3,260	0.02	0.51		0.40	11.44		11.84
11	0.77	5,940	0.03	0.62		1.10	25.42		26.52
16	0.75	8,600	0.02	0.62		1.38	37.16		38.54
21	0.73	11,240	0.02	0.56		1.91	43.51		45.42
26	0.90	13,700	0.02	0.59		2.22	56.25		58.47
31	1.13	15,460	0.02	0.62		2.57	66.16		68.73
36	1.27	18,060	0.02	0.59		3.04	74.02		77.06
41	1.77	19,160	0.02	0.59		3.23	78.41		81.64
46	2.80	18,980	0.02	0.64		3.20	84.62		87.82
51	2.86	21,120	0.02	0.58		3.41	84.69		88.10
			0.02	**0.60**	**#DIV/0!**				

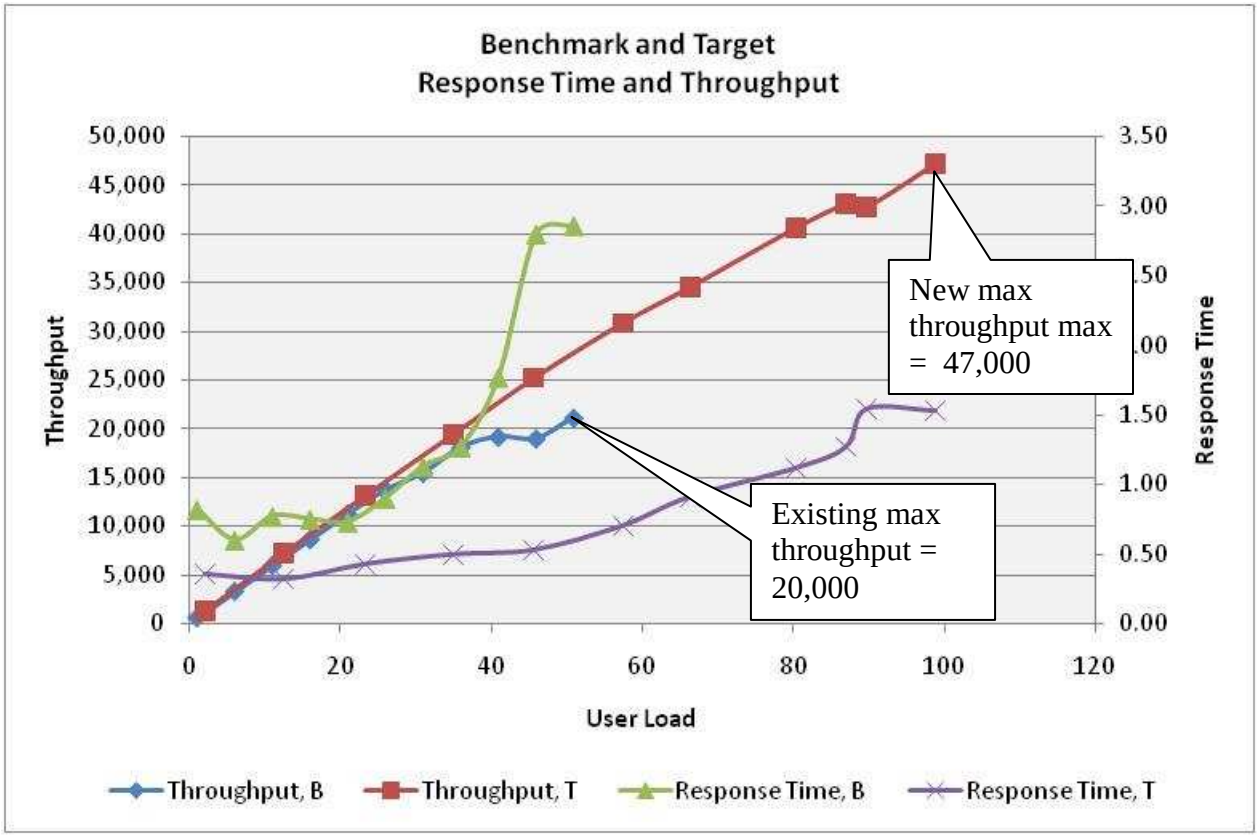
Step 4: Conduct Cost/Benefit Analysis
The new system can support an additional 19,000 map/hour; the estimated profit is $190/hour.
Assuming the system will be consistently utilized at 38,000 maps/hour for 10 hours a day, 5 day a week (10*5*4=200 hr/month), this investment would yield 10*5*4*190=$38,000 potential profit per month.

4.8 Example 3: Using faster CPU to maximize throughput

The same company from the previous Example considers renegotiating the existing level of service agreement. The management wants to evaluate what would be an additional monthly profit if the company did not have to guarantee the performance level of service (response time under 1 sec)?

Solution:

Follow Step 1 -2 from the previous example.
Step 3: Model the existing and proposed system. Determine the maximum throughput.



We can summarize the throughput and response time for the existing and upgraded system in the following table:

	Max Throughput (tr/hr)	Max User load
Before	20,000	51
After	47,000	99
% Improvement	**130%**	**94%**

It can be estimated that by using faster CPU (30/13.425=2.2 times faster) we improve both performance and scalability:

- Throughput (scalability) increased by 100% ; the curve extended
- User load (scalability) increased by 94%

Step 4: Conduct Cost/Benefit Analysis
With the new system we can support additional 47,000-21,000=26,000 map/hour, with the estimated profit of $260/hour. Assuming the system will be used 10 hours a day, 5 day a week (10*5*4=200 hr/month), this investment would yield 10*5*4*260=$52,000 profit per month.

4.9 Example 4: Using more CPU cores

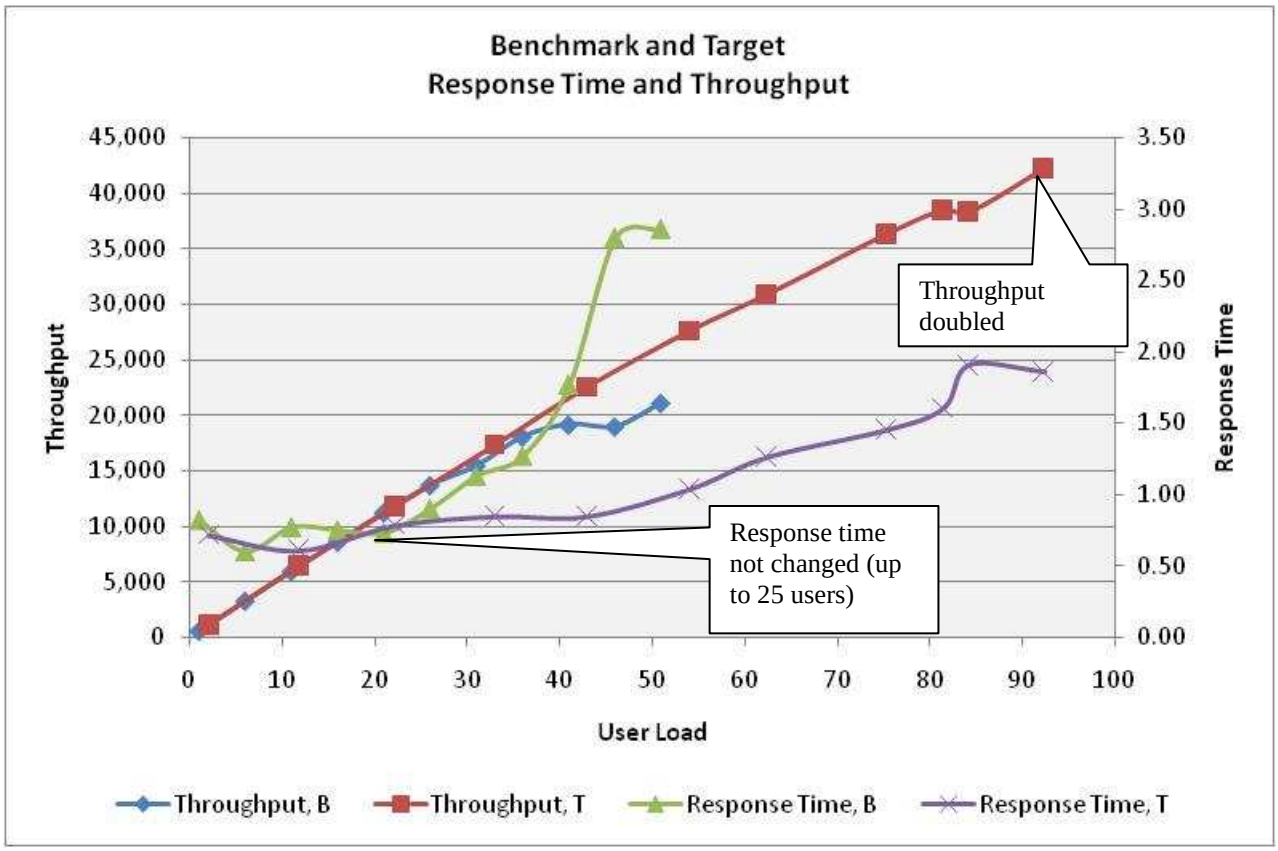
The same hosting company from the previous Example considers adding (doubling) CPU cores to the existing system. How will this improve performance and scalability of the system? Howis return on investment estiamted?

Solution:
Follow Step 1-2 from previous Example.

Step 3: Model the existing and proposed system. Determine the maximum throughput.

In this case, we increase the CPU Count to 8 in this worksheet and analyze the data.

	ThinkingTime	Spec_Per_CPU	Mbits_per_Tr	SOCCPUCount
Benchmark (B)	6	13.425	1.81	4
Target (T) or Modeled	6	**13.425**		8



The following table summarizes the key statistics.

Response Time (sec)	Max Throughput	Max User load
(at 23 users)	(tr/hr)	

Before	0.73	21,000	51
After	0.73	43,000	92
% Improvement	**0%**	**100%**	**94%**

It can be estimated that adding more (doubling) CPU cores from 4 to 8:

- Response time (performance) remains the same for the moderate user load (less than 25 users); the curve shifted to the right.
- Throughput (scalability) increased by 100%; the curve extended.
- User load (scalability) increased by 94%

Step 4: Conduct Cost/Benefit Analysis
With the new system we can support additional 43,000-21,000=22,000 map/hour, with the estimated profit of $220/hour. Assuming the system will be used 10 hours a day, 5 day a week (10*5*4=200 hr/month), this investment would yield 10*5*4*220=$44,000 potential profit per month.

4.10 Example 5: Estimating the value of tuning

The same hosting company from the previous Example hired a vendor to tune their existing system at the cost of $30,000. As a result it was report performance improved 50% (response time 0.5 times). What is the return on investment for the tuning engagement?

Solution:
Follow Step 1-2 from previous Example.

Step 3: Model the before and after system. Determine the maximum throughput.
Based on the response time reduction information, we can assume CPU service time was reduced to 0.3 seconds.
From the benchmark, we determined the original system CPU service time was equal to 0.6 seconds. We can then calculate the corresponding capacity of the system (4 core, 95% CPU utilization) using *Equation 6: Throughput as a function of CPU service time and number of cores:*

$$TH_t = \frac{3600 \times \%CPU_T \times \#CPU_T}{ST_b \times 100} \times \frac{SpecRatePerCPU_T}{SpecRatePerCPU_b}$$

TH before tuning =(3600*95*4)/(0.6*100)* 13.425/13.425= 22,800 tr/hr
TH after tuning =(3600*95*4)/(0.3*100)* 13.425/13.425= 45,600 tr/hr

As expected, given all other factors as constant, the reduction of CPU service time by 50% yields double the throughput.

Step 4: Conduct Cost/Benefit Analysis
With the new system we can support additional 45,600-22,800=22,800 map/hour, with the estimated profit of $228/hour. Assuming the system will be used 10 hours a day, 5 day a week (10*5*4=200 hr/month), this investment would yield 10*5*4*228=$45,600 potential profit in one month.

4.11 Example 6: Changing frequency of user operations

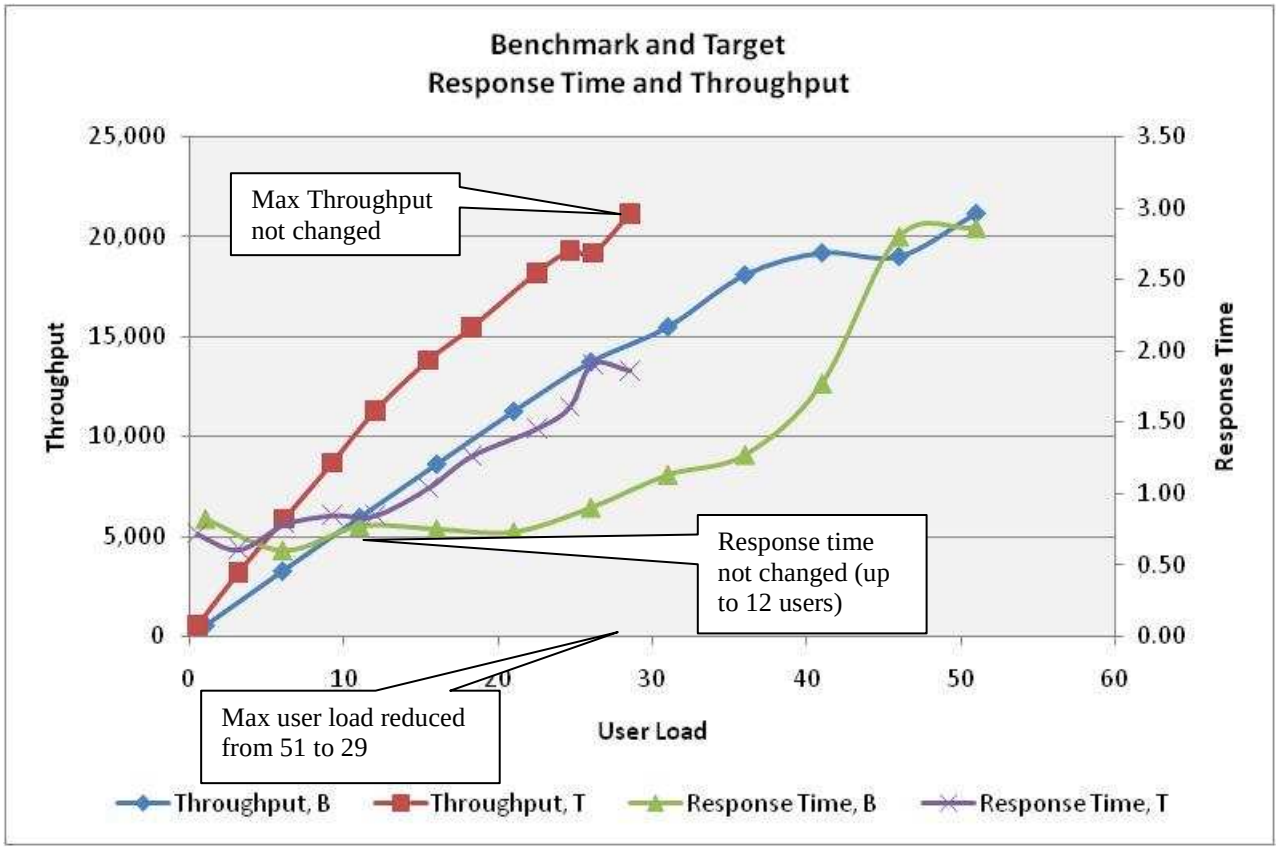
The same hosting company from the previous Example observed that the system was running at capacity (95% average CPU utilization) and users reported performance degradation. The IT manager analyzed the previous user workflow and anticipated that , users zoomed to an area of interest then analyzed the data for 6 seconds before repeating the operation. He now estimates the new and improved analytical tools allow users to reduce the analysis time from 6 to 3 seconds. As a result, users request a new map every 3 seconds. The same hosting company is negotiating new level of service agreement based on the guaranteed maximum number of concurrent users. What is the maximum number of users the company should agree to?

Solution:

Follow Step 1 -2 from the previous example.
Step 3: Model the existing and proposed system. Determine the maximum throughput.

In this case, reduce Think Time from 6 seconds to 3 seconds.

	ThinkingTime	Spec_Per_CPU	Mbits_per_Tr	SOCCPUCount
Benchmark (B)	6	13.425	1.81	4
Target (T) or Modeled	3	**13.425**		4



The following table summarizes the key statistics.

	Response Time (sec) (at 23 users)	Max Throughput (tr/hr)	Max User load
Before	0.77	21120	51
After	0.77	21144	29
% Improvement	**0%**	**0%**	**<43%>**

It can be estimated that by increasing the frequency of requesting maps (shortening think time):

- Response time (performance) remains the same for the moderate user load (less than 12 users).
- Maximum throughput (scalability) not changed.
- User load decreased 43%, from 51 to29 users.

Step 4: Conduct Cost/Benefit Analysis
With the new user workflow, the system can support a maximum of 29 users.

4.12 Example 7: Network sizing - changing the size of the map display

The same hosting company from the previous Example hosts an application with the map display size set at 800 x 600. Users have been requesting a larger map display size to improve usability, e.g. 1200 x 1000. Currently, the company pays $1000 a month per 1 Mbps of bandwidth. How much extra should the company charge per request?

Solution:
Follow Step 1 -2 from the previous example.

Step 3: Model the existing and proposed system. We analyzed the published benchmarks for a similar application. In addition to throughput, we found that a map display size of 1200 x 1000 requires 1.81 Mbits/tr while an 800 x 600 size requires 0.6 Mbits/tr.

We can use the Capacity Calculator to conduct the analysis. The following table lists the current requirements:

RT_t		sec
$Users_t$		users
Think		sec
$ST_{ArcSOC\ Zoom\ b}$	0.60	sec
$Mbits/tr_b$	0.6	Mbits/tr
$SpecRate_bPerCPU$		
$SpecRate_tPerCPU$		
%CPU		%
TH_t	21,000	tr/hr
$\#CPU_t$		**CPU cores**
Mbps	3.50	Mbps

To calculate the new requirements, we input Mbits/tr=1.81.

RT_t		sec
$Users_t$		users
Think		sec
$ST_{ArcSOC\ Zoom\ b}$	0.60	sec
$Mbits/tr_b$	1.81	Mbits/tr
$SpecRate_bPerCPU$		
$SpecRate_tPerCPU$		
%CPU		%
TH_t	21,000	tr/hr
$\#CPU_t$		**CPU cores**

Mbps	10.56	Mbps

Step 4: Conduct the Cost/Benefit Analysis
Assuming the system will be used 10 hours a day, 5 day a week, we estimate the number of requests per month is 10*5*4 * 21,000 = 4,200,000 request/month. The additional network cost per month is (10.56-3.50)*$1000= $7,060. The company should charge extra 7060/4,200,000=$0.0017 per request.

4.13 Example 8: Network sizing – increasing throughput

The same hosting company from the previous Example upgraded its server hardware and now supports double the number of requests from 21,000 tr/hr to 42,000 tr/hr. The company currently pays $1000 a month per 1 Mbps. What is the additional monthly network fee?

Solution:
Follow Step 1 -2 from the previous example.

Step 3: Model the existing and proposed system. We analyzed the published benchmarks for a similar application. In addition to throughput, we found that a map display size of 1200 x 1000 requires 1.81 Mbits/tr.

We can use Capacity Calculator to conduct analysis. The following table lists the current and new network requirements:

RT_t		sec
$Users_t$		users
Think		sec
$ST_{ArcSOC\ Zoom\ b}$	0.60	sec
$Mbits/tr_b$	1.81	Mbits/tr
$SpecRate_bPerCPU$		
$SpecRate_tPerCPU$		
%CPU		%
TH_t	**21,000**	tr/hr
$\#CPU_t$		**CPU cores**
Mbps	10.5	Mbps

RT_t		sec
$Users_t$		users
Think		sec
$ST_{ArcSOC\ Zoom\ b}$	0.60	sec
$Mbits/tr_b$	1.81	Mbits/tr
$SpecRate_bPerCPU$		
$SpecRate_tPerCPU$		

%CPU		%
TH_t	**42,000**	tr/hr
$\#CPU_t$		**CPU cores**
Mbps	21.12	Mbps

Step 4: Conduct Cost/Benefit Analysis
The company will have to pay (21.12-10.56)*$1000=$10,560 per month.

5 Walk-through of Benchmark Result Sections

The following sections describe the information presented within a typical benchmark report.

5.1 Application Architecture

This section provides a brief overview of the tested application. For practical information on this topic, see ESRI Application Architectures.

5.2 Hardware and Software Configuration

This section lists the tested hardware and software configuration. For more information, see ESRI Application Architectures. The following is an example of a test configuration.

Figure 2: Test Configuration Example

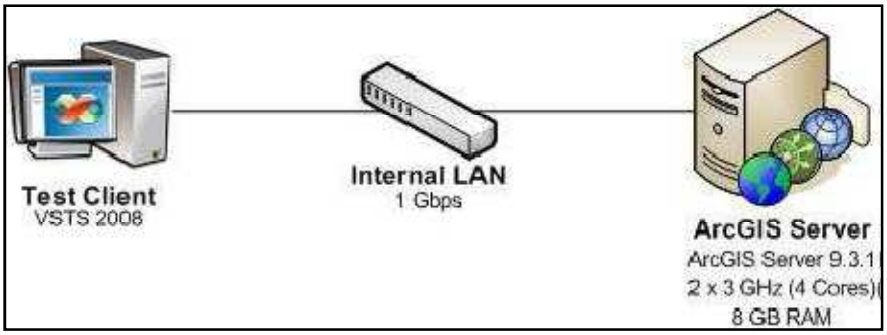


Diagram Key				
Web	SOM	SOC	RDBMS	File

5.3 Benchmark Results

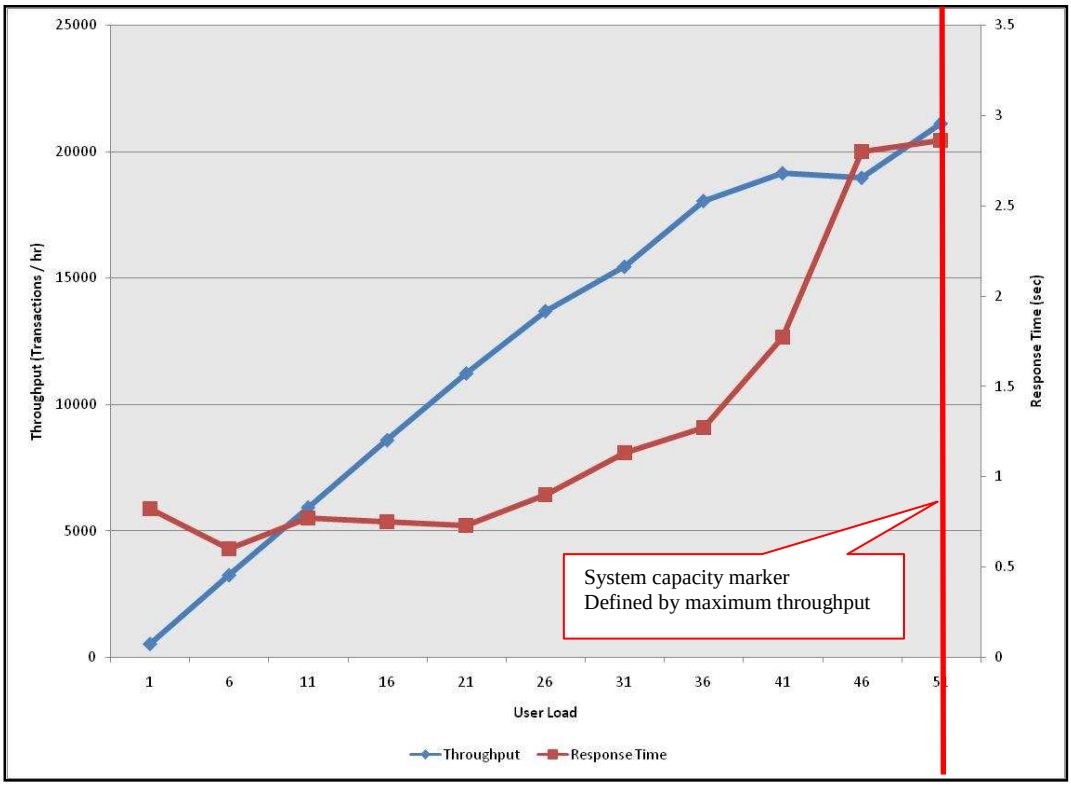
This is a key section of the performance benchmark document. It will report performance and scalability information.

5.3.1 Performance and Scalability

This section contains key test results and summary information derived from the raw test values.

The figure below shows an example of a key test results report.

Figure 3: Key Test Results



Listed under this chart you will find a summary of the system capacity. Capacity of the system can be determined as user load corresponding to one of the following criteria:

- at first error
- maximum throughput
- desired response time

In ESRI benchmarks, *capacity* is referred to as user load corresponding to **maximum throughput.**

Below is an example of how capacity information is reported:

Maximum Throughput (Transactions/Hour)	At User Load	Average Response Time (Seconds)
21,120	51	2.9

For more information on how to apply this information to your solution, see Estimating Capacity Using Benchmark Data.

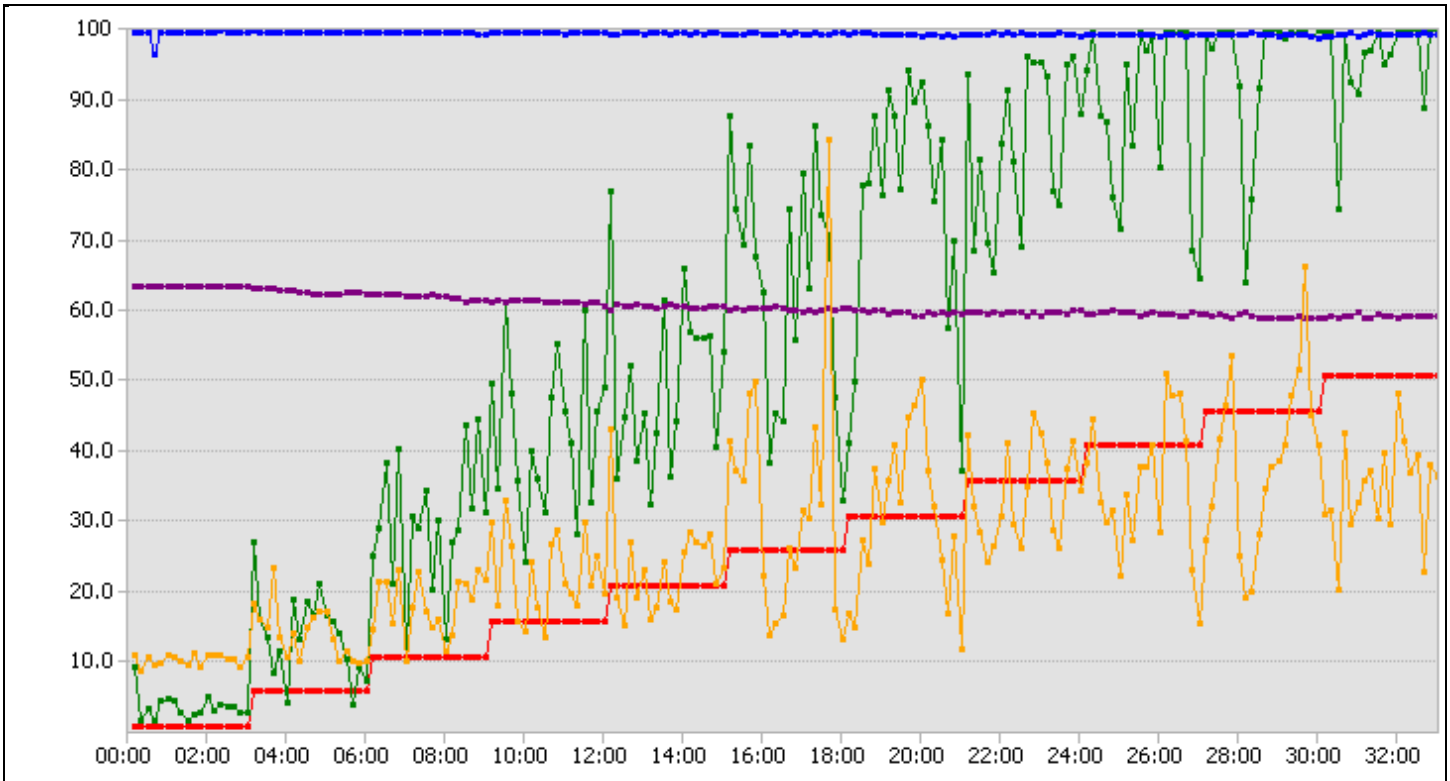
For details on how to analyze load test results, see

- MSDN Library, Analyzing Load Test Runs
- DS2009: ArcGIS Server Performance and Scalability—Testing Methodologies

5.3.2 Resource Utilization

This section includes key resource utilization that can be used to validate test results and identify solution bottlenecks. For example, in the case as shown below, the green line represents CPU utilization and demonstrates the system is CPU bound. It can be concluded that by adding more CPU resources, the capacity of this solution could be expanded.

Figure 4: Key Resource Utilization



Counter	Instance	Category	Computer	Color	Range	Min.	Max.	Avg.
User Load	_Total	Load Test: Scenario	ArcGIS Server		100	1	51	26
% Processor Time	_Total	Processor	ArcGIS Server		100	1.76	100	58.7
% Idle Time	0 C:	Physical Disk	ArcGIS Server		100	96.8	100	99.7
Available MB	-	Memory	ArcGIS Server		10,000	5,909	6,378	6,093
Bytes Received/Sec.	Broadcom BCM5708C NetXtreme II GigE	Network Interface	ArcGIS Server		100,000	8,989	84,503	26,905

For details on how to analyze load test results, see

- MSDN Library, Analyzing Load Test Runs
- DS2009: ArcGIS Server Performance and Scalability—Testing Methodologies

5.4 Capacity Planning

This section provides the input for the capacity planning model, typically found as an Excel spreadsheet within the benchmark package.

5.4.1 CPU SpecRate

CPU SpecRate
SpecRate is a standardized metric allowing various systems to be compared. See http://spec.org for specific information and results.

The following is an example of a reported Spec results:

- SpecRate/CPU = **13.425**
- Total CPU Cores = 4

5.4.2 CPU Service Time

CPU service time is a measure of how many seconds, on average, were needed for the CPU processor to process one request or transaction. It is used as an input for sizing models.

The following is an example of a service time reported result:

Web	ArcGIS SOC/SOM	Database
.03	0.62	N/A: RDBMS not utilized

5.4.3 Transaction Size

The following is an example of network bandwidth (?) reported results:

Average map size: **244,275 bytes**

6 Helpful Resources

- DS2009: ArcGIS Server Performance and Scalability—Testing Methodologies
  http://resources.esri.com/arcgisserver/apis/javascript/arcgis/index.cfm?fa=mediaGalleryDetails&mediaID=6D73B2DB-1422-2418-344143680A5154BA

  http://proceedings.esri.com/library/userconf/devsummit09/papers/performancetesting2009_devsummit_ppt_v1.pdf

- Patterns & practices: Performance Testing Guidance for Web Applications
  http://perftestingguide.codeplex.com/Release/ProjectReleases.aspx?ReleaseId=6690#DownloadId=17955

- SpecRate CINT 2006
  http://www.spec.org/cpu2006/results/cint2006.html

- MSDN, Chapter 2—Performance Modeling
  http://msdn.microsoft.com/en-us/library/ms998537.aspx

- Performance Testing Guidance for Web Applications
  http://msdn.microsoft.com/en-us/library/bb924375.aspx

- MSDN, Chapter 16—Testing .NET Application Performance
  http://msdn.microsoft.com/en-us/library/ms998581.aspx

- Getting Started with Team System Testing Tools
  http://msdn.microsoft.com/en-us/library/ms243146.aspx

- Fiddler
  http://www.fiddlertool.com/fiddler/version.asp

- Visual Studio 2008 Professional Edition (90-day trial)
  http://www.microsoft.com/downloads/details.aspx?familyid=83C3A1EC-ED72-4A79-8961-25635DB0192B&displaylang=en

- MSDN, Chapter 17—Load-Testing Web Applications
  http://msdn.microsoft.com/en-us/library/bb924372.aspx

- Creating a Web Test
  http://msdn.microsoft.com/en-us/library/ms182538.aspx