

Capacity Management and Sizing for Microsoft SharePoint Server 2010

This document is provided "as-is". Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2010 Microsoft Corporation. All rights reserved.

Kelley Vice  
Microsoft Corporation

April 2010

**Applies to: Microsoft SharePoint Server 2010**

Summary: This document contains extensive information about capacity planning and sizing for Microsoft SharePoint Server 2010.

**Note:** The capacity planning information in this document provides guidelines for you to use in your planning. It is based on testing performed at Microsoft, on live properties. However, your results are likely to vary based on the equipment you use and the features and functionality you implement for your sites.

Additional content is under development. Check back for new and updated content.

Contents

[Contents 2](#_Toc259784351)

[Capacity Management and Sizing Overview 1](#_Toc259784352)

[Glossary 2](#_Toc259784353)

[Who Should Read This Document 3](#_Toc259784354)

[Beginning to End 3](#_Toc259784355)

[Evaluating SharePoint Server 2010 4](#_Toc259784356)

[Upgrading from Office SharePoint Server 2007 4](#_Toc259784357)

[Tuning and Optimizing a Live SharePoint-based environment 5](#_Toc259784358)

[The Four Fundamentals of Performance 7](#_Toc259784359)

[Latency 7](#_Toc259784360)

[Throughput 8](#_Toc259784361)

[Data Scale 9](#_Toc259784362)

[Reliability 10](#_Toc259784363)

[Capacity Management versus Capacity Planning 11](#_Toc259784364)

[The SharePoint Server 2010 Capacity Management Model 11](#_Toc259784365)

[Over-Sizing versus Under-Sizing 13](#_Toc259784366)

[Limits and Boundaries 14](#_Toc259784367)

[How limits are established 15](#_Toc259784368)

[Key Differences: SharePoint Server 2010 versus Office SharePoint Server 2007 16](#_Toc259784369)

[Services and Features 17](#_Toc259784370)

[New Client Applications interactions with SharePoint Server 2010 22](#_Toc259784371)

[SharePoint Server 2010 Deployment Key Differentiators 24](#_Toc259784372)

[Specifications 24](#_Toc259784373)

[Hardware 24](#_Toc259784374)

[Topology 24](#_Toc259784375)

[Configuration 24](#_Toc259784376)

[Workload 25](#_Toc259784377)

[User Base 25](#_Toc259784378)

[Usage Characteristics 25](#_Toc259784379)

[Dataset 25](#_Toc259784380)

[Health and Performance 26](#_Toc259784381)

[Reference Architectures 27](#_Toc259784382)

[Single Server Deployment 27](#_Toc259784383)

[Small Farm Deployment 27](#_Toc259784384)

[Medium Farm Deployment 29](#_Toc259784385)

[Large Farm Deployment 29](#_Toc259784386)

[Right-Sizing SharePoint Server 2010 Deployments 31](#_Toc259784387)

[Step 1: Model 31](#_Toc259784388)

[Understand your expected workload and dataset 31](#_Toc259784389)

[Workload 32](#_Toc259784390)

[Estimating your production workload 34](#_Toc259784391)

[Analyzing your SharePoint Server 2010 IIS Logs 36](#_Toc259784392)

[Dataset 37](#_Toc259784393)

[Setting Farm Performance and Reliability Targets 38](#_Toc259784394)

[Step 2: Design 40](#_Toc259784395)

[Determine your starting point architecture 41](#_Toc259784396)

[SharePoint Server 2010 Technical Case Studies 41](#_Toc259784397)

[Select your hardware 41](#_Toc259784398)

[Hardware Selection Guidelines 42](#_Toc259784399)

[Choosing Processors 42](#_Toc259784400)

[Choosing Memory 42](#_Toc259784401)

[Choosing Networks 43](#_Toc259784402)

[Choosing Disks and Storage 43](#_Toc259784403)

[Step 3: Pilot, Test and Optimize 45](#_Toc259784404)

[Test 46](#_Toc259784405)

[Create a test plan 46](#_Toc259784406)

[Create the Test Environment 46](#_Toc259784407)

[Create Tests and Tools 48](#_Toc259784408)

[Optimize 52](#_Toc259784409)

[Deploy the pilot environment 52](#_Toc259784410)

[Step 4: Deploy 53](#_Toc259784411)

[Step 5: Maintain 54](#_Toc259784412)

[Monitor and Adjust 54](#_Toc259784413)

[Performance Counters 55](#_Toc259784414)

[System Counters 57](#_Toc259784415)

[SQL Server Counters 60](#_Toc259784416)

[Removing Bottlenecks 62](#_Toc259784417)

[Physical Bottleneck Resolution 63](#_Toc259784418)

# Capacity Management and Sizing Overview

Implementing a new environment based on Microsoft® SharePoint® Server 2010 demands that you understand and effectively plan and manage the capacity of that environment.

This document provides you with the information that will help you:

* Understand the concepts behind effective capacity management
* Define performance and capacity targets for your environment
* Select the appropriate data architecture
* Choose hardware to support the number of users, and the features you intend to deploy
* Test, validate and tune your environment to achieve your performance and capacity targets
* Monitor and adjust your environment to match demand

Capacity management is an ongoing process, because no implementation remains static in terms of content and usage. You need to plan for growth and change, so that your SharePoint-based environment can continue to deliver an effective business solution.

Capacity Planning is only one part of the Capacity Management cycle; it is the initial set of activities that brings the design architect to the point where he has an initial architecture that he believes will best serve his SharePoint Server 2010 deployment. The Capacity Management model includes additional steps to help you validate and tune the initial architecture, and provides a feedback loop for re-planning and optimizing the production environment until it can support design goals with optimal choices of hardware, topology, and configuration.

This white paper shows you how to maintain a good understanding of the capacity needs and capabilities of your deployment, by analysis of performance and volume data. It also reviews the major application impacts that affect capacity, including content characteristics and usage.

This document is divided into two major sections:

* [Capacity Management and Sizing Overview](#_Capacity_Management_and)  
  This section provides conceptual information and key capacity management considerations that will help you develop an effective capacity management strategy.
* [Right-Sizing SharePoint Server 2010 Deployments](#_Right-Sizing_SharePoint_Products)  
  This section provides step-by-step instructions on how to model, design, test, optimize, deploy and maintain your SharePoint-based environment.

In the overview section, we:

* Provide guidance on what content you should read for different deployment scenarios
* Describe the four fundamentals of performance
* Discuss over-sizing and under-sizing, and the drawbacks of each
* Discuss software limits and boundaries and how to use the boundaries guidance
* Describe the differences between SharePoint Server 2010 and Microsoft® Office SharePoint® Server 2007 as they relate to capacity management
* Describe the key differentiators between SharePoint Server 2010 deployments, and why they are important
* Describe the SharePoint solutions and how they relate to capacity management
* Describe the Capacity Management lifecycle and the steps required for proper sizing

## Glossary

There are some specialized terms you will encounter in SharePoint Server 2010 capacity management documentation. Here are a few key terms and their definitions.

* **RPS:** Requests per second. The number of requests received by a farm or server in one second. This is a common measurement of server and farm load.   
  Note that requests are different from page loads; each page contains several components, each of which creates one or more requests when the page is loaded. Therefore, one page load creates several requests. Typically, authentication handshakes and events consuming negligible resources are not counted in RPS measurements.
* **Peak hours:** The time or times during the day when load on the farm is at its maximum.
* **Peak load:** The average maximum daily load on the farm, measured in RPS.
* **Load spike:** Transient load peaks that fall outside normal peak hours. These may be caused by unplanned increases in user traffic, decreased farm throughput due to administrative operations, or combinations of such factors.

## Who Should Read This Document

### Beginning to End

*I want to know everything about SharePoint Server 2010 capacity management. Where do I start?*

You should begin by reading this document, which provides information about the general concepts behind capacity management, as well as links to additional documentation and resources. A site containing all available SharePoint Server 2010 capacity management documentation is available at <http://technet.microsoft.com/en-us/library/cc262971(Office.14).aspx>.

Once you have a good understanding of the concepts, you can read about the limits and boundaries of SharePoint Server 2010 here:

* [SharePoint Server 2010 Limits and Boundaries](http://office/14/teams/perf/server/cpwg/Shared%20Documents/White%20Paper%20Drafts/SharePoint%20Server%202010%20Limits%20and%20Boundaries) (http://technet.microsoft.com/en-us/library/cc262787(Office.14).aspx)

When you are ready to identify a starting point topology for your SharePoint-based environment, you can look through the library of available technical case studies to find the one that most closely matches your requirements. This library will grow as new case studies become available.

* [Performance and capacity technical case studies](http://technet.microsoft.com/en-us/library/cc261716(Office.14)aspx) (http://technet.microsoft.com/en-us/library/cc261716(Office.14)aspx).

Capacity management white papers are available for many specific SharePoint services and features. At the time of publication of this document, there are a limited number of such white papers available, but more will be added as they become available. To download these white papers, visit the following URL:

* <http://technet.microsoft.com/en-us/library/ff608068(office.14).aspx>

Read the following documents for information about database sizing and performance:

* [Storage and SQL Server capacity planning and configuration](http://technet.microsoft.com/en-us/library/%20a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx) (http://technet.microsoft.com/en-us/library/a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx)

Read the following documents for information about remote BLOB storage (RBS):

* [Plan for remote BLOB storage (RBS)](http://technet.microsoft.com/en-us/library/c1f83b4f-a507-42f7-bd82-fed5404ed1ad(Office.14).aspx) (http://technet.microsoft.com/en-us/library/c1f83b4f-a507-42f7-bd82-fed5404ed1ad(Office.14).aspx)

Read the following documents for information about health monitoring and troubleshooting using the health monitoring tools built into the Central Administration interface:

* [Health monitoring](http://technet.microsoft.com/en-us/library/ee681489(office.14).aspx)) (http://technet.microsoft.com/en-us/library/ee681489(office.14).aspx)
* [Solving problems and troubleshooting](http://technet.microsoft.com/en-us/library/ee748639(office.14).aspx) (http://technet.microsoft.com/en-us/library/ee748639(office.14).aspx)

The following documents address general performance tuning guidelines and a variety of specific performance and capacity subjects. The list of available documents will grow as new content is developed.

* [Use search administration reports (SharePoint Server 2010)](http://technet.microsoft.com/en-us/library/ee808861(office.14).aspx)

For more information about virtualizing SharePoint-based servers, see [Virtualization planning](http://technet.microsoft.com/en-us/library/71c203cd-7534-47b0-9122-657d72ff0080(Office.14).aspx) (http://technet.microsoft.com/en-us/library/71c203cd-7534-47b0-9122-657d72ff0080(Office.14).aspx).

### Evaluating SharePoint Server 2010

*I am an IT Pro or business decision maker, and I'm looking for a solution to specific business problems. SharePoint Server 2010 is an option for my deployment. Can it provide features and scalability that meet my specific requirements?*

There are several sections in this document you should read to provide you with general information about how SharePoint Server 2010 scales to meet the demands of specific solutions, and how to determine the hardware that will be required to support your requirements.

* [Key Differences: SharePoint Server 2010 versus Office SharePoint Server 2007](#_Key_Differences:_Office_1)
* [Limits and Boundaries](#_Key_Differences:_Office)

You should also read the following articles that will help you evaluate SharePoint Server 2010 for your specific business requirements.

* [Product evaluation for SharePoint Server 2010](http://technet.microsoft.com/en-us/library/cc261970(office.14).aspx) (http://technet.microsoft.com/en-us/library/cc261970(office.14).aspx)

* [SharePoint Server 2010 Limits and Boundaries](http://office/14/teams/perf/server/cpwg/Shared Documents/White Paper Drafts/SharePoint Server 2010 Limits and Boundaries) (http://technet.microsoft.com/en-us/library/cc262787(Office.14).aspx)

### Upgrading from Office SharePoint Server 2007

*I'm currently using Office SharePoint Server 2007. What has changed in SharePoint Server 2010, and what do I have to consider if I upgrade? What impact will the upgrade have on my topology's performance and scale?*

For information about how performance and capacity factors differ between Office SharePoint Server 2007 and SharePoint Server 2010, start by reading sections of this document that specify the key differences and their impact on performance and capacity.

* [Key Differences: SharePoint Server 2010 versus Office SharePoint Server 2007](#_Key_Differences:_Office_1)

Additionally, there are several articles and documents that discuss more general upgrade considerations, and provide guidance on how to plan and execute an upgrade from Office SharePoint Server 2007.

* [Upgrading to SharePoint Server 2010](http://technet.microsoft.com/en-us/library/cc303420(office.14).aspx) (http://technet.microsoft.com/en-us/library/cc303420(office.14).aspx)

### Tuning and Optimizing a Live SharePoint-based environment

*I have deployed SharePoint Server 2010, and I want to make sure I have the right hardware and topology in place. How do I validate my architecture and maintain it properly?*

Begin by reading the following sections of this document:

* [Step 3: Pilot, Test and Optimize](#_Step_3:_Pilot,)
* [Monitor and Adjust](#_Monitor_and_Adjust)

Read the following documents for information about health monitoring using the health monitoring tools built into the Central Administration interface:

* [Health monitoring](http://technet.microsoft.com/en-us/library/ee681489(office.14).aspx)) (http://technet.microsoft.com/en-us/library/ee681489(office.14).aspx)

*I have deployed SharePoint Server 2010, and I am experiencing performance issues. How do I troubleshoot and optimize my environment?*

Begin by reading the following sections of this document:

* [Step 5: Maintain](#_Step_5:_Maintain)

Read the following documents for information about troubleshooting using the health monitoring tools built into the Central Administration interface:

* [Solving problems and troubleshooting](http://technet.microsoft.com/en-us/library/ee748639(office.14).aspx) (<http://technet.microsoft.com/en-us/library/ee748639(office.14).aspx)>

Capacity management white papers are available for many specific SharePoint services and features. At the time of publication of this document, there are a limited number of such white papers available, but more will be added as they become available. To download these white papers, visit the following URL:

* <http://technet.microsoft.com/en-us/library/ff608068(office.14).aspx>

Read the following documents for information about database sizing and performance:

* [Storage and SQL Server capacity planning and configuration](http://technet.microsoft.com/en-us/library/%20a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx) (http://technet.microsoft.com/en-us/library/a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx)

Read the following documents for information about remote BLOB storage (RBS):

* [Plan for remote BLOB storage (RBS)](http://technet.microsoft.com/en-us/library/c1f83b4f-a507-42f7-bd82-fed5404ed1ad(Office.14).aspx) (http://technet.microsoft.com/en-us/library/c1f83b4f-a507-42f7-bd82-fed5404ed1ad(Office.14).aspx)

## The Four Fundamentals of Performance

Capacity management focuses on four major aspects of sizing your solution:

* **Latency** - For the purposes of capacity management, latency is defined as the duration between the time a user initiates an action, such as clicking on a hyperlink, and the time until the last byte is transmitted to the client application or Web browser.
* **Throughput** - Throughput is defined as the number of concurrent requests that a server or server farm is able to process.
* **Data scale** - Data scale is defined as the content size and data corpus that the system can host. The structure and distribution of the content databases has a significant impact on the time it takes the system to process requests (latency) and the number of concurrent requests it can serve (throughput).
* **Reliability** - Reliability is a measurement of the ability of the system to meet the targets set for the latency and throughput over time.

The main goal of managing your environment's capacity is to establish and maintain a system that meets your organization's latency, throughput, data scale, and reliability targets.

### Latency

Latency, also referred to as *end user perceived latency*, is composed of three major components:

* The time it takes the server to receive and process the request
* The time it takes the request and the server response to transfer over the network
* The time it takes the response to render on the client application

Different organizations define different latency goals based on business requirements and user expectations. Some organizations can afford latency of a few seconds, while others require very fast transactions. Optimizing for very fast transactions tends to be more costly, and usually requires more powerful clients and servers, more recent browser and client application versions, high-bandwidth network solutions, and possibly development investments and page tuning.

Some major factors that contribute to unacceptable end-user perceived latencies and examples of some common problems are listed below. These factors are especially relevant in scenarios where the clients are geographically distant from the server farm, or are accessing the farm across a low-bandwidth network connection.

* Features, services, or configuration parameters that are not optimized may delay the processing of requests and impact latency for both remote and local clients. See the Reliability and Throughput section of this document for more information.
* Web pages that generate unnecessary requests to the server in order to download required data and resources. Optimization would include downloading the minimum amount of resources to draw the page, reducing the sizes of images, storing the static resources in folders that enable anonymous access, clustering requests and enabling page interactivity while resources are downloaded asynchronously from the server. These optimizations are important for achieving an acceptable first time visit browse experience.
* Excessive volume of data being transmitted over the network contributes to latency and throughput degradation. For example, images and other binary objects on a page should use a compressed format such as .png or .jpg instead of bitmaps whenever possible.
* Web pages that are not optimized for second-access page loads. *Page Load Time* (PLT) improves for second-access page loads because some page resources are cached on the client, and the browser must only download dynamic uncached content. Unacceptable second-access page load latencies are often due to improper BLOB cache configuration or local browser caching being disabled on client computers. Optimizations would include proper caching of resources on the client.
* Web pages that have non-optimized custom JavaScript code. This may slow rendering of the page on the client. Optimization would defer JavaScript from being processed on the client until the rest of the page has loaded, and preferably calling scripts instead of adding JavaScript inline.

### Throughput

Throughput is described by the number of requests that a server farm is able to process in a unit of time, and is also often used to measure the scale of operations that the system is expected to sustain based on the size of the organization and its usage characteristics. Every operation has a specific cost in server farm resources. Understanding the demand and deploying a farm architecture that can consistently satisfy demand requires estimating the expected load, and testing the architecture under load to validate that latency does not fall below target when concurrency is high and the system is under stress.

Some common examples of low throughput conditions include:

* **Inadequate hardware resources** – When the farm receives more requests than it can process concurrently, some requests are queued, which cumulatively delays the processing of each subsequent request until demand is reduced enough for the queue to be cleared.  
  Some examples of optimizing a farm to sustain higher throughput include:
  + Ensure that the processors on farm servers are not over-utilized. For example, if CPU usage during peak hours or load spikes consistently exceeds 80%, add additional servers or redistribute services to other farm servers.
  + Ensure that there is adequate memory on application servers and Web servers to contain the entire cache. This will help to avoid calls to the database to serve requests for uncached content.
  + Ensure that database servers are free of bottlenecks. If total available disk IOPS are inadequate to support peak demand, add more disks or redistribute databases to underutilized disks. See [Removing Bottlenecks](#_Removing_Bottlenecks) later in this document for more information.
  + If adding resources to existing machines is insufficient to resolve throughput issues, add servers and redistribute affected features and services to the new servers.
* **Non-optimized custom Web pages** – Adding custom code to frequently used pages in a production environment is a common cause of throughput issues. Adding custom code may generate additional round trips to the database servers or Web services to service data requests. Customization of infrequently used pages may not significantly impact throughput, but even well-optimized code can decrease farm throughput if it is requested thousands of times a day.  
  SharePoint administrators can enable the Developer Dashboard to identify custom code that requires optimization.  
  Some examples of optimizing custom code include:
  + Minimize the number of Web service requests and SQL queries.
  + Fetch the minimum required data in each trip to the database server while minimizing the number of necessary round trips.
  + Avoid adding custom code to frequently used pages.
  + Use indexes when retrieving a filtered amount of data.
* **Untrusted solutions** – Deploying custom code in bin folders can cause slow server performance. Every time a page containing untrusted code is requested, SharePoint Server 2010 must perform security checks before the page can be loaded.  
  Unless there is a specific reason to deploy untrusted code, you should install custom assemblies in the GAC to avoid unnecessary security checking.

### Data Scale

Data scale is the corpus of data the server or server farm is able to store while meeting latency and throughput targets. Generally, the greater the data volume on the farm, the greater the impact on overall throughput and user experience. The way data is distributed across disks and database servers can also affect farm latency and throughput.

Database sizing, data architecture, and adequate database server hardware are all critical to an optimal database solution. In an ideal deployment, content databases are sized according to limits guidance and are distributed across physical disks such that requests are not queued due to disk overutilization, and database servers are able to support peak loads and unexpected spikes without exceeding resource utilization thresholds.

Also, certain operations can lock certain tables for the duration of the operation. An example of this is large site deletion, which can cause the related tables in the content database where the site resides to be locked until the delete operation is complete.

Some examples of optimizing a farm for data and storage performance include:

* Ensure that databases are properly distributed across the database servers, and that database server resources are sufficient to support the volume and distribution of data.
* Separate database volumes into unique Logical Units (LUNs), consisting of unique physical disk spindles. Use multiple disks with low seek time and appropriate RAID configurations to satisfy database server storage demands.
* Use RBS (Remote BLOB Storage) if your corpus contains a large number of BLOBs (Binary Large Objects). For more information, see [Plan for remote BLOB storage (RBS)](http://technet.microsoft.com/en-us/library/c1f83b4f-a507-42f7-bd82-fed5404ed1ad(Office.14).aspx) (<http://technet.microsoft.com/en-us/library/c1f83b4f-a507-42f7-bd82-fed5404ed1ad(Office.14).aspx)>.

For more details on how to plan data scale, see [Storage and SQL Server capacity planning and configuration](http://technet.microsoft.com/en-us/library/%20a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx) (http://technet.microsoft.com/en-us/library/a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx).

### Reliability

Reliability is the aggregate measurement of the server farm's capacity to meet established latency, throughput, and data capacity targets over time. A reliable farm is one for which uptime, responsiveness, failure rate, and frequency and amplitude of latency spikes are within established targets and operational requirements. A reliable farm can also consistently sustain latency and throughput targets during peak load and peak hours or when system operations like crawling or daily backups take place.

A major factor in sustaining reliability is the impact of common administrative operations on performance targets. During certain operations, such as rebuilding the database indexes, maintenance timer jobs, or deleting multiple sites with large volume of content, the system might not be able to process user requests as quickly. In this case, both latency and throughput of end user requests can be affected. The impact on the farm depends on the frequency and transaction cost of such less common operations, and whether they are run during normal operating hours.

Some examples of how to sustain a more reliable system include:

* Schedule resource-intensive timer jobs and administrative tasks during off-peak hours.
* Scale up hardware on existing farm servers, or scale out by adding Web servers, application servers or additional database servers.
* Distribute resource-intensive services and features to dedicated servers. You can also use a hardware load balancer to direct feature-specific traffic to a Web server dedicated to specific features or services.

## Capacity Management versus Capacity Planning

*Capacity management* extends the concept of capacity planning to express a cyclical approach in which the capacity of a SharePoint Server 2010 deployment is continually monitored and optimized to accommodate changing conditions and requirements.

SharePoint Server 2010 offers increased flexibility and can be configured to sustain usage scenarios in a wide variety of different scale points. There is no single deployment architecture, so system designers and administrators must understand the requirements for their specific environments.

### The SharePoint Server 2010 Capacity Management Model



* **Step 1: Model** – Modelling is the process by which you decide the key solutions you want your environment to support, and establish all important metrics and parameters. The output of the modelling exercise should be a list of all the key data you need to design your environment.
  + Understand your expected workload and dataset
  + Setting farm performance and reliability targets
  + Analyzing your SharePoint Server 2010 IIS logs
* **Step 2: Design** – Once you have gathered the data from Step 1, you can design your farm. Outputs are detailed data architecture and physical and logical topologies.
  + Determine your starting point architecture
  + Select your hardware
* **Step 3: Pilot, Test and Optimize** – If you have designed a new deployment, you need to deploy a pilot environment for testing against your workload and expected usage characteristics. For an existing farm, testing is advised when major changes are being made to the infrastructure, but regular optimization based on monitoring results may be necessary to maintain performance targets.   
  The output from this phase is analysis of test results against targets, and an optimized architecture able to sustain established performance and capacity targets.
  + Pilot - Deploy a pilot environment
  + Test - Test against latency and throughput targets
  + Optimize – Gather test results and start with substep 1 below, moving on to the next substep when indicated.
* **Step 4: Deploy** – This step describes implementing the farm, or deploying changes to an existing farm.   
  Output for a new design is a completed deployment to live production, including all content and user migrations.  
  Output for an existing farm is revised farm maps and updates to maintenance plans.
* **Step 5: Monitor and maintain** – This step describes how to set up monitoring, and how to predict and identify bottlenecks, as well as perform regular maintenance and bottleneck mitigation activities.

## 

## Over-Sizing versus Under-Sizing

*Over sizing* describes a deployment in which targets are achieved without full utilization of hardware, and the resources in the SharePoint farm are substantially and consistently underutilized. In an over-sized deployment, memory, CPU, and other indicators on the farm's resources show that it can well serve the demand with fewer resources. The downside of oversizing is increased hardware and maintenance expenditures and can impose greater power and space demands.

*Under sizing* describes a deployment in which performance and capacity targets are not achieved because hardware resources in the SharePoint farm are over-utilized. The downside is high latency leading to a poor user experience, low satisfaction, frequent escalations, high support costs, and unnecessary spending for troubleshooting and tuning the environment.

When you design your farm, it is important to ensure that your farm can meet established performance and capacity targets, both under regular peak load and unexpected spikes. Design, testing, and optimization will help you ensure that your farm has the correct hardware.

In order to maintain performance targets and accommodate growth, it is always more desirable to have more resources than you need to meet your targets. The cost of overinvestment in hardware is almost always far less than the cumulative expenses related to troubleshooting problems cause by under sizing.

You should always size a system to respond adequately during peak demand, which may be different for specific services at different times. To effectively estimate capacity requirements, you need to identify the worst case demand period for all resources. There may be increased load on various features and services at certain times of the day, such as first thing in the morning or after lunch.

The farm also needs to be able to support unplanned peaks, such as when organization-wide announcements are made and an unusually high number of user access a site at once as a result. During such periods of high demand, users will experience high latency or not get a response from the farm at all unless adequate farm resources are available to satisfy the increased load on the farm.

Farm capacity should also be revisited when additional users will be provisioned within the enterprise. Situations such as a merger or acquisition characterized by new employees or members accessing the farm as a result may have adverse effects on performance if not planned and estimated in advance.

## 

## Limits and Boundaries

In SharePoint Server 2010, there are certain limits that are by design and cannot be exceeded, and others that are set to default values that may be changed by the farm administrator. There are also certain limits that are not represented by a configurable value, such as the number of site collections per Web application.

* *Boundaries* are absolute limits that cannot be exceeded by design. It is important to understand these limits to ensure that you do not make incorrect assumptions when you design your farm.  
  An example of a boundary is the 2 GB document size limit; you cannot configure SharePoint Server 2010 to store documents that are larger than 2 GB. This is a built-in absolute value, and cannot be exceeded by design.
* *Thresholds* are those that have a default value that cannot be exceeded unless the value is modified. Thresholds can, in certain circumstances, be exceeded to accommodate variances in your farm design, but it is important to understand that doing so may impact the performance of the farm as well as the effective value of other limits.  
  The default value of certain thresholds can only be exceeded up to an absolute maximum value. A good example is the document size limit again. By default, the document size limit is set to 50MB, but can be changed to a maximum value of 2GB.
* *Supported limits* define the tested value for a given parameter. The default values for these limits were defined by testing, and represent the known limitations of the product. Exceeding supported limits may cause unexpected results, significant performance degradation, or other detrimental effects.   
  Some supported limits are configurable parameters that are set by default to the recommended value, while others relate to parameters that are not represented by a configurable value.  
  An example of a supported limit is the number of site collections per Web application. The supported limit is 500,000, which is the largest number of site collections per Web application that met performance benchmarks during testing.

It is important to note that many of the limit values provided in this document represent a point in a curve that describes an increasing resource load and concomitant performance degradation as the value increases. Therefore, exceeding certain limits, such as the number of site collections per Web application, may only result in a fractional decrease in farm performance. However, in most cases, operating at or near an established limit is not a best practice, as acceptable performance and reliability targets are best achieved when a farm's design provides for a reasonable balance of limits values.

Thresholds and supported limits guidelines are determined by performance. In other words, you can exceed the default values of the limits, but as you increase the limit value, farm performance and the effective value of other limits may be affected. Many limits in SharePoint Server 2010 can be changed, but it is important to understand how changing a given limit affects other parts of the farm.

### How limits are established

In SharePoint Server 2010, thresholds and supported limits are established through testing and observation of farm behavior under increasing loads up to the point where farm services and operations reach their effective operational limits. Some farm services and components can support a higher load than others, so in some cases it is necessary to assign a limit value based on an average of several factors.

For example, observations of farm behavior under load when site collections are added indicate that certain features exhibit unacceptably high latency while other features are still operating within acceptable parameters. Therefore, the maximum value assigned to the number of site collections is not absolute, but is calculated based on an expected set of usage characteristics in which overall farm performance would be acceptable at the given limit under most circumstances.

If other services are operating under parameters that are higher than those used for limits testing, the maximum effective limits of other services will be reduced. It is therefore important to execute rigorous capacity management and scale testing exercises for specific deployments in order to establish effective limits for that environment.

For more information on boundaries and limits and how they affect the capacity management process, see [SharePoint Server 2010 Limits and Boundaries](http://technet.microsoft.com/en-us/library/cc262787(Office.14).aspx) (http://technet.microsoft.com/en-us/library/cc262787(Office.14).aspx).

## Key Differences: SharePoint Server 2010 versus Office SharePoint Server 2007

SharePoint Server 2010 offers a richer set of features and a more flexible topology model than earlier versions. Before you employ this more complex architecture to deliver more powerful features and functionality to your users, you must carefully consider the impact upon your farm's capacity and performance.

In Office SharePoint Server 2007, there were four major services you could enable in SSPs (Shared Service Providers): Search Service, Excel Calculation Service, User Profile Service, and the Business Data Catalog (BDC) Service. Additionally, there was a relatively smaller set of clients that could directly interface with Office SharePoint Server 2007.

In SharePoint Server 2010, there are more available services, known as SSAs (SharePoint service applications), and SharePoint Server 2010 offers a much broader range of client applications that can interact with the farm, including several new Office applications, mobile devices, designer tools and browsers. Some examples of how expanded client interactions impact the capacity considerations include:

* SharePoint Server 2010 includes social applications that integrate with Outlook, allowing Outlook 2010 clients to display information about e-mail recipients that is pulled from the SharePoint farm when emails are viewed in the Outlook client. This introduces a new set of traffic patterns and server load that should be accounted for.
* Some new Microsoft® Office 2010 client capabilities automatically refresh data against the SharePoint farm, even when the client applications are open but are not actively being used. Such clients as SharePoint Workspace and OneNote will also introduce some new traffic patterns and server load that should be accounted for.
* SharePoint Server 2010 new Web interactivity capabilities, such as Office Web Apps that enable editing office files directly from the browser, using AJAX calls that introduce some new traffic patterns and server load that should be accounted for.

In Office SharePoint Server 2007, the primary client used to interact with the server was the Web browser. Given the richer feature set in SharePoint Server 2010, the overall requests per second (RPS) is expected to grow. Further, the percent of requests coming from the browser is expected to be smaller than in Office SharePoint Server 2007, making room for the growing percent of new traffic coming from other clients as they are broadly adopted throughout the organization.

Additionally, SharePoint Server 2010 introduces new functionality such as native embedded video support which can add stress to the farm. Some functionality has also been expanded to support a larger scale than previous versions.

The section below describes these client interactions, services and features and their overall performance and capacity implications on the system that you should consider when designing your solution.

For more information about upgrading to SharePoint Server 2010, see [Upgrading to SharePoint Server 2010](http://technet.microsoft.com/en-us/library/cc303420(office.14).aspx) (http://technet.microsoft.com/en-us/library/cc303420(office.14).aspx).

### Services and Features

The table below provides a simplified high level description of the resource requirements for the different services on each tier. Blank cells indicate that the service does not run on or impact that tier.

– Indicates minimal or negligible cost on the resource. The service can share this resource with other services.

 – Indicates medium cost on the resource. The service might be able to share this resource with other services that have minimal impact.

 – Indicates high cost on the resource. The service should generally not share this resource with other services.

For more details on how to plan SQL Server databases, see [Storage and SQL Server capacity planning and configuration](http://technet.microsoft.com/en-us/library/%20a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx) (http://technet.microsoft.com/en-us/library/a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx).

Capacity management white papers are available for many specific SharePoint services and features. At the time of publication of this document, there are a limited number of such white papers available, but more will be added as they become available. To download these white papers, visit the following URL: <http://technet.microsoft.com/en-us/library/ff608068(office.14).aspx>

| **Service Application** | **Web server** | | **App Server** | | **SQL Server** | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **CPU** | **RAM** | **CPU** | **RAM** | **CPU** | **IOps** | **Storage** |
| SharePoint Foundation Service | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 |  |  | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 |
| Central Admin service |  |  | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: * | Description: * | Description: * |
| Logging Service \* | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 |  |  | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 |
| SharePoint Search Service | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 |
| Word Viewing Service Application \* | Description: * | Description: * | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 |  |  |  |
| PowerPoint Service \* | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 |  |  |  |
| Excel Calculation Service | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: * | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 |  |  |  |
| Visio Service \* | Description: * | Description: * | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: * | Description: * | Description: * |
| Access Service \* | Description: * | Description: * | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: * | Description: * | Description: * |
| User Profile Service | Description: * | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 |
| Managed Metadata Service \* | Description: * | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: * | Description: * | Description: cid:image010.png@01CABC66.FE8C63E0 |
| Web Analytics Service \* | Description: * | Description: * |  |  | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 |
| Business Connection Service \* | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 |  |  |  |
| InfoPath Forms Service | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: * | Description: * | Description: * |
| Word Conversion Service | Description: * | Description: * | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: * | Description: * | Description: * |
| PerformancePoint Service Application \* | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: * | Description: * | Description: * |
| Project Service \* | Description: * | Description: * | Description: * | Description: * | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: Description: cid:image010.png@01CABC66.FE8C63E0 |
| Sandboxed Solutions \* | Description: * | Description: * | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 |  |  |  |
| Workflow capabilities \* | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 |  |  |  |  |  |
| Timer Service | Description: cid:image010.png@01CABC66.FE8C63E0 | Description: Description: cid:image010.png@01CABC66.FE8C63E0 | Description: Description: cid:image010.png@01CABC66.FE8C63E0 | Description: Description: cid:image010.png@01CABC66.FE8C63E0 |  |  |  |
| PowerPivot \* | Description: * | Description: * | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 | Description: Description: cid:image010.png@01CABC66.FE8C63E0 | Description: Description: cid:image010.png@01CABC66.FE8C63E0 | Description: cid:image011.png@01CABC66.FE8C63E0 |

Note: An asterisk (\*) indicates a new service in SharePoint Server 2010.

* **SharePoint Foundation Service –** The core SharePoint service for content collaboration. In large SharePoint deployments, it is recommended to allocate redundant Web servers based on expected traffic load, properly size the SQL Server-based computers that service the content database(s), and properly allocate storage based on the size of the farm.
* **Central Admin Service –** The administration service. This service has relatively small capacity requirements. It is recommended that you enable this service on multiple farm servers to ensure redundancy.
* **Logging Service –** The service to record usage and health indicators for monitoring purposes. This is a write-intensive service, and can require relatively large disk space depending on the number of indicators and the frequency at which they are logged. In large SharePoint Server 2010 deployments, it is recommended that you isolate the usage database from the content databases on different SQL Server-based computers.
* **SharePoint Search Service Application** – The shared service application that provides indexing and querying capabilities. Generally this is a relatively resource intensive service, that can scale to serve very large content deployments. In large SharePoint deployments where enterprise search is key, it is recommended that you use a separate "service farm" to host search service applications, with dedicated database resources, use multiple application servers servicing specific search functions (crawl or query), and dedicated target Web server(s) on the content farms to ensure proper throughput for crawling and querying.   
  You can also enable the FAST Service Applications as your Search Service Application. Choose to create one or more FAST Search Connectors for indexing content with FAST Search Server 2010 and create another FAST Search Query (SSA) for querying content that is crawled by the FAST Search Connector(s).
* **Word Viewing Service Application** – Enabling this service lets you view Word documents directly from the browser, this service is added once you install Office Web Apps on top of SharePoint Server 2010. This service requires an application server to prepare the original files for browser viewing. In large SharePoint deployments, it is recommended that you scale out the service to multiple application servers for redundancy and throughput.  
  **Note:** Word and OneNote editing in the browser are also added once you install Office Web Apps on top of SharePoint Server 2010, but these capabilities do not leverage any service applications, and run on the Web servers.
* **PowerPoint Service Application** – This service displays and allows users to edit PowerPoint files directly in the browser, also enables you to broadcast and share live PowerPoint presentations. This service is added once you install Office Web Apps on top of SharePoint Server 2010. This service requires an application server to prepare the original files for browser viewing. In large SharePoint deployments where this becomes a frequently used capability, it is recommended that you deploy multiple application servers to ensure proper redundancy and throughput, and add more Web servers when PowerPoint Broadcast is frequently used as well.
* **Excel Calculation Service Application –** This service displays Excel worksheets directly in the browser and performs Excel calculations on the server. It also enables editing of worksheets directly from the browser once you install Office Web Apps on top of SharePoint Server 2010. In large SharePoint deployments where this becomes a frequently used capability, it is recommended that you allocate sufficient application servers with adequate RAM to ensure proper performance and throughput.
* **PowerPivot for SharePoint –** The service to display PowerPivot enabled Excel worksheets directly from the browser. In large SharePoint deployments where this becomes a frequently used capability, it is recommended that you allocate sufficient application servers with adequate RAM and CPU to ensure proper performance and throughput. See [Hardware and Software Requirements (PowerPivot for SharePoint)](http://msdn.microsoft.com/en-us/library/ee210640(SQL.105).aspx) for more details.
* **Visio Service Application –** The service to display dynamic Visio diagrams directly in the browser. This service has a dependency on the Session State Service Application, which requires a relatively small SQL Server database. The Visio service requires an application server to prepare the original Visio files for browser viewing. In large SharePoint deployments where this becomes a frequently used capability, it is recommended that you scale out the service to multiple application servers with adequate CPU and RAM to ensure proper performance and throughput.
* **Access Service Application –** The service to host Access solutions inside SharePoint Server 2010. In large SharePoint deployments where this becomes a frequently used capability, it is recommended that you scale out to multiple application servers with adequate RAM for proper performance and throughput. The Access service utilizes SQL Reporting Services, which will require a SQL Server database that can be co-located with other databases.
* **User Profile Service Application** – The service that lights up the social scenarios in SharePoint Server 2010 and enables My Sites, Tagging, Notes, Profile sync with directories and other social capabilities. The profile service requires three relatively resource intensive databases - the synchronization, the Profile and the Social Tagging databases, this service is dependent on the Managed Metadata Service Application.In large SharePoint deployments you should consider distributing this service to a federated services farm, and properly sizing the SQL Server resources to ensure proper performance of the common transactions and of the directory synchronization jobs.
* **Managed Metadata Service Application** – The service that serves the central metadata store across the enterprise and allows the syndication of content types across the enterprise. The service can be federated to a dedicated services farm. It requires a database that can be co-located with other databases.
* **Web Analytics Service Application** – The service that aggregates and stores statistics on the usage characteristics of the farm. This service has relatively high SQL Server resource and storage demands. The service can be federated to a dedicated services farm. In large SharePoint deployments, it is recommended that you properly size the SQL Server resources serving the Web Analytics databases and isolate these databases from other critical or resource intensive databases on different database servers.
* **Business Connection Service Application –** The service that enables the integration of various other organizational line-of-business applications with SharePoint Server 2010. This service requires an application service to maintain the connection with the various external resources. In large SharePoint deployments where this is a frequently used capability, it is recommended that you allocate sufficient application servers with adequate RAM for proper performance.
* **InfoPath Forms Service Application –** The service that enables sophisticated forms in SharePoint Server 2010 and the integration with the InfoPath client application for form creation. This service requires an application server and has a dependency on the Session State Service Application, which requires a relatively small SQL Server database. The service can be co-located with other services and has relatively small capacity requirements that can grow depending on the frequency of use of this capability.
* **Word Automation Service Application –** The service that enables conversion of Word files from one format, such as DOC, to another format, such as DOCX or PDF. This service requires an application server. In large SharePoint deployments where this becomes a frequently used capability, it is recommended that you scale out the service to multiple application servers with adequate CPU resources to achieve proper conversion throughput. This service also requires a relatively small SQL Server database to maintain the queue of conversion jobs.
* **PerformancePoint Service Application** – The service that enables the PerformancePoint BI capabilities in SharePoint Server 2010, allowing you to create analytic visualizations. This service requires an application server and a database. In large SharePoint deployments where this becomes a frequently used capability, it is recommended that you allocate adequate RAM to the application servers for proper performance and throughput.
* **Project Service Application** – The service that enables all the Microsoft® Project Server 2010 planning and tracking capabilities on top of SharePoint Server 2010.This service requires an application server and a relatively resource intensive database. In large SharePoint deployments where this is a frequently used capability, you should dedicate a server running SQL Server for the Project Server database and even consider a dedicated SharePoint farm for the Project Server management solutions.
* **Timer Service –** The process responsible of executing the various scheduled tasks on the different servers in the farm. There are various timer jobs that the system executes, some run on all the servers, and some run only on specific servers depending on the server's role, some of the timer jobs are resource intensive and have the potential to load both the server they execute on and the SQL Server databases, depending on their activity and the amount of content they are operation on. In large SharePoint deployments where timer jobs have the potential to impact end user interactions, it is recommended that you dedicate a server to isolate the execution of the more resource intensive jobs.
* **Workflow –** The capability that enables integrated workflows in SharePoint Server 2010, workflows are handled on the Web server and the load on the system is dependent on the complexity of the workflows and the amount of total events they handle. In large SharePoint deployments where this is a frequently used capability, you should consider adding web servers or isolating a server to handle only the workflow timer service to ensure end user traffic is not impacted and that the workflow throughput is not falling behind schedule.
* **Sandboxed Solutions –** The service that enables isolation of custom code to dedicated farm resources. In large SharePoint deployments where this becomes a frequently used capability, you should consider dedicating additional web servers once the custom code becomes a bottleneck.

### New Client Applications interactions with SharePoint Server 2010

This section describes some of the new client-server interactions that SharePoint Server 2010 supports and their capacity planning implications.

The table below provides a simplified high level description of the typical load that these new capabilities introduce on the system:

– Indicates minimal load on the system's resources  
 – Indicates medium load on the system's resources  
 – Indicates high load on the system's resources

|  |  |  |
| --- | --- | --- |
| **Client** | **Traffic** | **Payload** |
| Office Web Apps | Description: Description: Description: Description: cid:image011.png@01CABC66.FE8C63E0 | Description: Description: Description: Description: cid:image010.png@01CABC66.FE8C63E0 |
| PowerPoint Broadcast | Description: Description: Description: Description: cid:image011.png@01CABC66.FE8C63E0 | Description: Description: Description: Description: * |
| Word and PowerPoint 2010 client application | Description: Description: Description: Description: cid:image010.png@01CABC66.FE8C63E0 | Description: Description: Description: Description: * |
| OneNote client application | Description: Description: Description: Description: cid:image011.png@01CABC66.FE8C63E0 | Description: Description: Description: Description: cid:image011.png@01CABC66.FE8C63E0 |
| Outlook Social Connector | Description: Description: Description: Description: cid:image010.png@01CABC66.FE8C63E0 | Description: Description: Description: Description: cid:image010.png@01CABC66.FE8C63E0 |
| SharePoint Workspace | Description: Description: Description: Description: cid:image011.png@01CABC66.FE8C63E0 | Description: Description: Description: Description: cid:image010.png@01CABC66.FE8C63E0 |

* **Office Web Apps** – Web viewing and editing of Word, PowerPoint, Excel and OneNote files is a sub set of browser requests, with slightly different traffic characteristics, this type of interaction introduces a relatively high load of traffic necessary for enabling capabilities like co-authoring. In large SharePoint deployments where these capabilities are enabled, you should expect additional load on the Web servers.
* **PowerPoint Broadcast** – The set of requests associated with viewing live PowerPoint presentation in the Web browser is another sub set of browser requests. During live PowerPoint broadcast sessions, the participating clients pull changes from the server that is running SharePoint Server 2010. In large SharePoint deployments where this is a frequently used capability, you should expect additional load on the Web servers.
* **Word and PowerPoint 2010 client applications** – The Word and PowerPoint 2010 clients have new features that take advantage of the SharePoint farm. One example is the document co-authoring capability, in which all client applications participating in a co-authoring session frequently upload and download updates against the server. In large SharePoint deployments where this is a frequently used capability, you should expect additional load on the Web servers.
* **OneNote 2010 client application** – The OneNote 2010 client interacts with the SharePoint farm in a similar fashion to the previous OneNote version, using SharePoint Server 2010 to share and allow co-authoring of OneNote notebooks. This scenario introduces load on SharePoint Server 2010 even when the client is open but not actively in use. In large SharePoint deployments where this is a frequently used capability, you should expect additional load on the Web servers.
* **Outlook client** – Outlook 2010 has a new capability that takes advantage of the SharePoint farm named Outlook social connector (this component can be added to previous versions of Outlook as well), this capability enables you to view social activity pulled from the SharePoint farm directly in e-mails. In large SharePoint deployments where this capability is enabled, you should expect additional load on the Web servers.
* **SharePoint Workspace –** SharePoint Workspace 2010clients has new features that take advantage of the SharePoint farm and enable you to sync web sites, lists and document libraries to you client for offline use. These scenarios introduce load on SharePoint Server 2010 also when the client is left open. In large SharePoint deployments where this is a frequently used capability, you should expect additional load on the Web servers.

## SharePoint Server 2010 Deployment Key Differentiators

Each SharePoint Server 2010 deployment has a key set of characteristics that will make it unique and different from other farms. These key differentiators can be described by these four major categories:

* **Specification** describes the farm's hardware, and the farm topology and configuration.
* **Workload** describes the demand on the farm, including the number of users, and the usage characteristics.
* **Dataset** describes contents sizes and distribution.
* **Health and performance** describes the farm's performance against latency and throughput targets.

### Specifications

#### Hardware

Hardware is the computer's physical resources such as processors, memory, and hard disks. Hardware also includes physical network components such as NICs (Network Interface Cards), cables, switches, routers and hardware load balancers. Many performance and capacity issues can be resolved by ensuring that the right hardware is being used. Conversely, a single misapplication of a hardware resource, such as inadequate memory on a server, can affect performance across the entire farm.

#### Topology

Topology is the distribution and interrelationships of farm hardware and components. There are two kinds of topology:

* Logical topology, which is the map of software components such as services and features in a farm.
* Physical topology, which is the map of servers and physical resources.

Typically, the number of users and usage characteristics determine the physical topology of a farm, and business requirements such as the need to support specific features for expected load drives the logical topology.

#### Configuration

We use the term configuration to describe software settings and how parameters are set. Also, configuration refers to caching, RBS (Remote BLOB Storage), how configurable limits are set, and any part of the software environment that can be set or modified to meet specific requirements.

### Workload

Workload defines the key operational characteristics of the farm, including the user base, concurrency, features in use, and the user agents or client applications that are used to connect with the farm.

Different SharePoint features have different associated costs on the farm's resources. Popularity of more costly features has the potential to significantly impact the performance and the health of the system. Understanding your expected demand and usage characteristics will enable you to properly size your implementation, and reduce the risk of constantly running your system in an unhealthy condition.

#### User Base

The user base of a SharePoint-based application is a combination of the total number of users and how they are geographically distributed. Also, within the total user base, there are subgroups of users who may use given features or services more heavily than other groups. Concurrency of users is defined as the total percentage of users actively using the system at a given time. Indicators that define the user base include the number of total unique users, number of concurrent users and others.

#### Usage Characteristics

A farm's performance can be affected not only by the number of users interacting with the system, but also by their usage characteristics. Two organizations with the same number of users may have dramatically different requirements based on how often their users access farm resources, and whether resource-intensive features and services are enabled on the farm. Indicators that describe the usage characteristics include the frequency of unique operations, the overall operational mix (the ratio of read and write operations and administrative operations), and the usage patterns and load against new features that are enabled on the farm (such as My Sites, Search, Workflows, Office Web Apps).

### Dataset

The volume of content stored in the system and the characteristics of the architecture in which it is stored can have a significant impact on the overall health and performance of the system. Understanding the size, access frequency, and distribution of data will enable you to properly size the storage in your system and prevent it from becoming the bottleneck that slows down user interactions with farm services and affects the end user experience.

In order to properly estimate and design the storage architecture of a SharePoint-based solution, you need to know the volume of data you will store on the system, and how many users are requesting data from different data sources. The volume of the content is an important element of sizing disk capacity, because it may influence the performance of other features, and may also potentially impact network latency and available bandwidth. Indicators that define the dataset include total size of content, total number of documents, total number of site collections, average and maximum sizes of site collection and others.

### Health and Performance

SharePoint farm health is essentially a simplified measurement or score that reflects the reliability, stability, and performance of the system. How well the farm performs against targets is basically dependent on the first three differentiators. The health and performance score can be tracked and described by a distillation of a set of indicators; these are described in more detail in the next chapter. These indicators include the system's uptime, end user perceived latency, page failure rates, resource utilization indicators (CPU, RAM) and others.

Any significant change in hardware, topology, configuration, workload, or dataset can vary the reliability and responsiveness of the system considerably. The health score can be used to track performance over time and to assess how changing operating conditions or system modifications affect farm reliability.

## Reference Architectures

SharePoint Server 2010 is a complex and powerful product, and there is no one-size-fits-all architecture solution. Each SharePoint deployment is unique, and is defined by its usage and data characteristics. Every organization needs to perform a thorough capacity management process and effectively leverage the flexibility that the SharePoint Server 2010 system offers to tailor a properly sized solution that best satisfies the organizational needs.

The concept of reference architectures is meant to describe and illustrate the different major categories of SharePoint deployments, and not to provide a recipe for architects to use to design their solutions. This section focuses on describing the vectors on which SharePoint deployments tend to scale.

The architectures listed here are provided as a useful way to understand the general differentiators between these generic categories, and to distinguish them by general cost factors and scale of effort.

### Single Server Deployment

The single server deployment architecture consists of one server running SharePoint Server 2010 and a supported version of SQL Server. This architecture may be appropriate for evaluation purposes, developers or for an isolated non-mission critical departmental implementation with only a few users, but we do not recommend its use for a production environment.



### Small Farm Deployment

A small farm deployment consists of a single database server or cluster and one or two SharePoint Server 2010-based computers. The major architecture characteristics include limited redundancy and failover, and a minimal set of SharePoint capabilities enabled.

A small farm is useful to serve only limited deployments, with a minimal set of service applications enabled, a relatively small user base, a relatively low usage load (a few request per min up to very few requests per second), and a relatively small volume of data (10s of gigabytes).



### 

### Medium Farm Deployment

This architecture introduces the breakdown of the topology into three tier – dedicated Web servers, dedicated application servers and one or more database servers or clusters. Separating the front end server tier from the application server tier allows greater flexibility in service isolation and helps balancing the load across the system.

This is the most common architecture, and encompasses a wide spectrum of service topologies and farm sizes. A medium farm deployment is useful to serve environments with:

* Several service applications distributed across multiple servers. A typical set of features might include Office Web Apps Service, User Profile Service, Managed Metadata Service, Excel Calculation Service, and others.
* A user base of a few ten thousands of users and a load of a few tens of requests per second
* A data store of one or two terabytes



### Large Farm Deployment

Large farm deployments introduce the breakdown of services and solutions across multiple farms, and further scaling out the tiers on a single farm. Several SharePoint services can be setup on a dedicated services farm that serves requests from multiple consumption farms. In these large architectures, there are typically Web servers; multiple application servers, depending on the usage characteristic of each of the local (non-federated) services; and multiple SQL Server-based servers or SQL Server clusters, depending on the content size and the application services databases that are enabled on the farm.  
Such architecture is expected to serve deployments with:

* Several service applications federated and consumed from dedicated services farm, typically the User Profile Service, Search, Managed Metadata service, and Web Analytics.
* Most other service applications are enabled locally
* A user base in the range of hundreds of thousands of users
* A usage load in the range of hundreds of requests per second
* A data set in the range of tens of terabytes



# Right-Sizing SharePoint Server 2010 Deployments

When you have a good appreciation and understanding of capacity planning and management, you can apply your knowledge to system sizing. Sizing is the term used to describe the selection and configuration of appropriate data architecture, logical and physical topology, and hardware for a solution platform. There is a range of capacity management and usage considerations that affect how you should determine the most appropriate hardware and configuration options.

In this section, we describe the steps you should take to undertake effective capacity management for your environment. Each step requires certain information for successful execution, and has a set of deliverables that you will use in the subsequent step. For each step, these requirements and deliverables are outlined in tables.

## Step 1: Model

Modeling your SharePoint-based environment begins with analyzing your existing solutions and estimating the expected demand and targets for the deployment you are planning to set up. You start by gathering information about your user base, data requirements, latency and throughput targets, and document the SharePoint features you want to deploy. Use this section to understand what data you should collect, how to collect it, and how it can be used in subsequent steps.

### Understand your expected workload and dataset

Proper sizing of a SharePoint Server 2010 implementation requires that you study and understand the demand characteristics that your solution is expected to handle. Understanding the demand requires that you be able to describe both the workload characteristics such as number of users and the most frequently used operations, and dataset characteristics such as content size and content distribution.

This section can help you understand some specific metrics and parameters you should collect and mechanisms by which they can be collected.

#### 

#### Workload

Workload describes the demand that the system will need to sustain, the user base and usage characteristics. The following table provides some key metrics that are helpful in determining your workload. You can use this table to record these metrics as you collect them.

|  |  |  |
| --- | --- | --- |
| **Workload Characteristics** | **Value** | |
| Average daily RPS |  | |
| Average RPS at peak time |  | |
| Total number of unique users per day |  | |
| Average daily concurrent users |  | |
| Peak concurrent users at peak time |  | |
| Total number of requests per day |  | |
| **Expected workload distribution** | **No. of Requests per day** | **%** |
| Web Browser - Search Crawl |  |  |
| Web Browser - General Collaboration Interaction |  |  |
| Web Browser - Social Interaction |  |  |
| Web Browser - General Interaction |  |  |
| Web Browser - Office Web Apps |  |  |
| Office Clients |  |  |
| OneNote Client |  |  |
| SharePoint Workspace |  |  |
| Outlook RSS Sync |  |  |
| Outlook Social Connector |  |  |
| Other interactions (Custom Applications/Web services) |  |  |

* **Concurrent users** – It is most common to measure the concurrency of operations executed on the server farm as the number of distinct users generating requests in a given time frame. The key metrics are the daily average and the concurrent users at peak load.
* **Requests per second (RPS)** – RPS is a commonly used indicator used to describe the demand on the server farm expressed in the number of requests processed by the farm per second, but with no differentiation between the type or size of requests.   
  Every organization's user base generates system load at a rate that is dependent on the organization's unique usage characteristics.

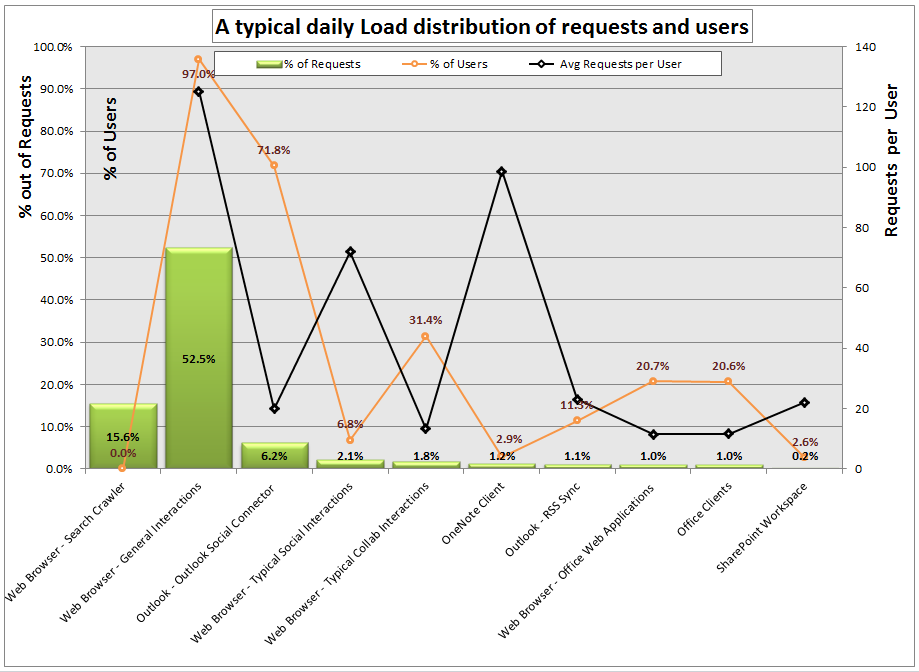
Since not all requests have identical transaction costs on the system, it is common to filter out certain types of requests that can bias this indicator, it is most common to measure RPS, using all request in the IIS log but the cheap Authentication handshake requests (401HTTP status), the values reported in the SharePoint Server 2010 Capacity documents are based on this method. The interesting measures to indicate would be the daily average and the daily max RPS.

* **Total daily requests** – Total daily requests is a good indicator of the overall load the system will need to handle. It is most common to measure all requests except authentication handshake requests (HTTP status 401) in a 24 hours period.
* **Total daily users** - Total users is another key indicator of the overall load the system will need to handle. This measurement is the actual number of unique users in a 24 hours period, not the total number of employees in the organization.

**Note** – the Total daily users can indicate the growth potential of the load on the farm, for example if the number of potential user base is 100k employees, 15k daily users can indicate that the load may significantly grow over time as user adoption increases.

* **Workload Distribution** – Understanding the distribution of the requests based on the clients applications that are hitting the farm will help predict the expected trend and load changes after switching to SharePoint Server 2010. As users transition to newer clients, like Outlook 2010 or Office 2010, and start using the new capabilities new load patterns, the RPS and total request are expected to grow. For each client we can describe the number of distinct users using it in a time frame of a day, and the amount of total requests that the client or feature generate on the server.

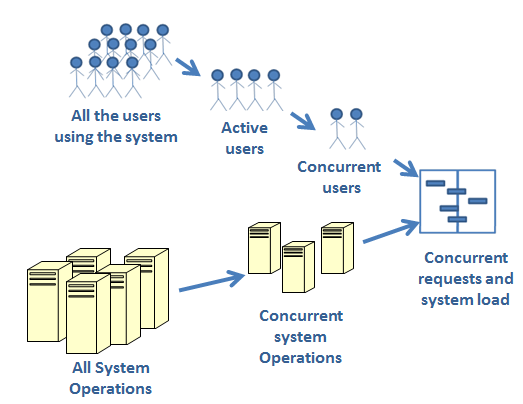
For example, the chart below shows a snapshot of a live Microsoft environment serving a typical social solution, where you can see that the majority of the load is generated by the search crawler and typical end user web browsing. You can also observe that there is significant load introduced by the new Outlook Social Connector feature (6.2 percent of the requests).



##### Estimating your production workload

In estimating the required throughput your farm needs to be able to sustain, begin with estimating the mix of transactions that will be used in your farm. Focus on analyzing the most frequently used transactions the system will serve, understanding how frequently they will be used and by how many users. That will help you validate later whether the farm can sustain such load in pre-production testing.

The following diagram describes the relationship of the workload and load on the system:



To estimate your expected workload, collect the following information:

* **Identify user interactions** such as typical web page browses, file downloads and uploads, Office Web Application views and edits in the browser, co-authoring interactions, SharePoint Workspace site syncs, Outlook Social Connections,RSS sync (in Outlook or other viewers), PowerPoint Broadcasts, OneNote shared notebooks, Excel Service shared workbooks, Access Service shared applicationsand others. See [Services and Features](#_Services_and_Features_1) earlier in this document for more information.  
  Focus on the identifying the interactions that may be unique to your deployment, and recognize the expected impact of such load, examples can be significant use of InfoPath Forms, Excel Service Calculations and similar dedicated solutions.
* **Identify system operations** such as Search incremental crawls, daily backups, profile sync timer jobs, web analytics processing, logging timer jobs and others.
* **Estimate the total number of users per day** that are expected to utilize each capability, derive the estimated concurrent users and high level Requests per second, there are some assumptions you will be making such as present concurrency and the factor of RPS per concurrent users that is different across capabilities, you should use [the workload table](#_Workload) earlier in this section for your estimates. It is important to focus on peak hours, rather than average throughput. Planning for peak activity, you are able to proper size your SharePoint-based solution.

If you have an existing Office SharePoint Server 2007 solution, you can mine the IIS log files or look to other Web monitoring tools you have to better understand some of the expected behaviors from the existing solution or [see the instructions below](#_Analyzing_your_SharePoint) for more details. If you are not migrating an existing solution, you should fill out the table using rough estimates. In later steps you will need to validate your assumptions and tune the system.

##### Analyzing your SharePoint Server 2010 IIS Logs

To discover key metrics about an existing SharePoint Server 2010 deployment, such as how many users are active, how heavily they are using the system, what kind of requests are coming in, and from what kind of clients they originate, it is necessary to extract data from ULS and IIS logs. One of the easiest ways to acquire this data is to use Log Parser, a powerful tool available free for download from Microsoft. Log Parser can read and write to a number of textual and binary formats, including all the IIS formats.

For detailed information about how to analyze SharePoint Server 2010 usage using Log Parser, read [Analyzing Microsoft SharePoint Products and Technologies Usage](http://www.microsoft.com/downloads/details.aspx?familyid=f159af68-c3a3-413c-a3f7-2e0be6d5532e&displaylang=en&tm) (http://www.microsoft.com/downloads/details.aspx?familyid=f159af68-c3a3-413c-a3f7-2e0be6d5532e&displaylang=en&tm).

You can download Log Parser 2.2 at <http://www.microsoft.com/downloads/details.aspx?FamilyID=890CD06B-ABF8-4C25-91B2-F8D975CF8C07&displaylang=en>.

#### Dataset

Dataset describes the volume of content stored in the system and how it can be distributed in the data store. The following table provides some key metrics that are helpful in determining your dataset. You can use this table to record these metrics as you collect them.

|  |  |
| --- | --- |
| **Object** | **Value** |
| DB size (in GB) |  |
| Number of Content DBs |  |
| Number of site collections |  |
| Number of web apps |  |
| Number of sites |  |
| Search index size (# of items) |  |
| Number of docs |  |
| Number of lists |  |
| Average size of sites |  |
| Largest site size |  |
| Number of user profiles |  |

* **Content size** – Understanding the size of the content that you expect to store in the SharePoint Server 2010 system is important for planning and architecting the system storage, and also for properly sizing the Search solution that will crawl and index this content. The content size is described in total disk space. If you are migrating content from an existing deployment you might find it simple to identify the total size that you will move; while planning you should leave room for growth over time based on the predicted trend.
* **Total number of documents** – Other than the data corpus size, it is important to track the overall number of items. The system reacts differently if 100 GB of data is composed of 50 files of 2 GB each versus 100,000 files of 1 KB each. In large deployments, the less stress there is on a single item, document or area of documents, the better performance will be. Widely distributed content like multiple smaller files across many sites and site collection is easier to serve then a single large document library with very large files.
* **Maximum site collection size** – It is important to identify what is the biggest unit of content that you will store in SharePoint Server 2010; usually it is an organizational need that prevents you from splitting that unit of content. Average size of all site collections and the estimated total number of site collections are additional indicators that will help you identify your preferred data architecture.
* **Service applications data characteristics** – In addition to analysing the storage needs for the content store, you should analyse and estimate the sizes of other SharePoint Server 2010 stores, including:
  + Total size of the Search index
  + The profile database total size based on the number of user in the profile store
  + The social database total size based on the expected number of tags, colleagues and activities
  + The metadata store size
  + The size of the usage database
  + The size of the Web Analytics data base

For more information on how to estimate database sizes, see [Storage and SQL Server capacity planning and configuration](http://technet.microsoft.com/en-us/library/%20a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx) (http://technet.microsoft.com/en-us/library/a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx).

### Setting Farm Performance and Reliability Targets

One of the deliverables of [Step 1](#_Step_1:_Model) is a good understanding of the performance and reliability targets that best fit the needs of your organization. A properly designed SharePoint solution should be able to achieve "four nines" (99.99%) of uptime with sub-second server responsiveness.

The indicators used to describe the performance and reliability of the farm can include:

* **Server availability** – Usually described by the percent of overall uptime of the system. You should track any unexpected downtime and compare the overall availability to the organizational target you set. The targets are commonly described by a number of nines (i.e. 99%, 99.9%, 99.99%)
* **Server responsiveness** – The time it takes the farm to serve requests is a good indicator to track the health of the farm. This indicator is usually named server side latency, and it is common to use the average or median (the 50th percentile) latency of the daily requests being served. The targets are commonly described in sub seconds or seconds. Note that if your organization has a target to serve pages from SharePoint Server 2010 in less than two seconds, then the server side goal needs to be sub seconds to leave time for the page to reach the client over the network and time to render in the browser. Also in general longer server response times are an indication of an unhealthy farm, as this usually as an impact on throughput and rarely can RPS keep up if you spend more than a second on the server on most requests
* **Server spikiness** – Another good server side latency indicator worth tracking is the behaviour of the slowest 5% of all requests. Slower requests are usually the requests that hit the system when it is under higher load or even more commonly, requests that are impacted by less frequent activity that occur while users interact with the system; a healthy system is one that has the slowest requests under control as well. The target here is similar to Server Responsiveness, but to achieve sub-second response on server spikiness, you will need to build the system with a lot of spare resources to handle the spikes in load.
* **System Resource utilization** – Other common indicators used to track the health of the system are a collection of system counters that indicate the health of each server in the farm topology. The most frequently used indicators to track are % CPU utilization and Available Memory; however, there are several additional counters that can help identify a non-healthy system; more details can be found in [Step 5: Maintain](#_Step_5:_Maintain).

## 

## Step 2: Design

Now that you have finished collecting some facts or estimates on the solution you need to deliver, you are ready to start the next step of designing a proposed architecture that you predict will be able to sustain the expected demand.

By the end of this step you should have a design for your physical topology and a layout for your logical topology, so you should be able to go ahead with any necessary purchase orders.

The hardware specifications and the number of machines you layout are tightly related, to handle a specific load there are several solutions you can choose to deploy. It is common to either use a small set of strong machines (scale up) or a larger set of smaller machines (scale out); each solution has its advantages and disadvantages when it comes to capacity, redundancy, power, cost, space, and other considerations.

We recommend that you begin this step by determining your architecture and topology. Define how you plan to layout the different farms and the different services in each farm, and then pick the hardware specifications for each of the individual servers in your design. You can also execute this process by identifying the hardware specifications you are expected to deploy (many organizations are constrained to a certain company standard) and then define your architecture and topology.

Use the following table to record your design parameters. The data included is sample data, and should not be used to size your farm. It is intended to demonstrate how to use this table for your own data.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Role** | **Type (Standard or virtual)** | **# of machines** | **Procs** | **RAM** | **IOPS need** | **Disk size OS+Log** | **Data drive** |
| Web servers | Virtual | 4 | 4 cores | 8 | N/A | 400 GB | N/A |
| Content database server  **Example** | Standard | 1 cluster | 4 quad-core 2.33 (GHz) | 48 | 2k | 400 GB | 20 disks of 300GB  @ 15K RPM |
| Application servers | Virtual | 4 | 4 cores | 16 | N/A | 400 GB | N/A |
| Search Crawl Target  Web server | Virtual | 1 | 4 cores | 8 | N/A | 400 GB | N/A |
| Search Query server | Standard | 2 | 2 quad-core 2.33 (GHz) | 32 | N/A | 400 GB | 500 GB |
| Search Crawler server | Standard | 2 | 2 quad-core 2.33 (GHz) | 16 | 400 | 400 GB | N/A |
| Search Crawl database server | Standard | 1 cluster | 4 quad-core 2.33 (GHz) | 48 | 4k (tuned for read) | 100 GB | 16 disks of 150GB @ 15K RPM |
| Search Property Store database + Administration database server | Standard | 1 cluster | 4 quad-core 2.33 (GHz) | 48 | 2k (tuned for write) | 100 GB | 16 disks of 150GB @ 15K RPM |

### Determine your starting point architecture

This section describes how to select a starting point architecture.

When you deploy SharePoint Server 2010, you can choose from a range of topologies to implement your solution; you may deploy a single server or scale out many servers to a SharePoint farm with clustered or mirrored database servers and discreet application servers for various services. Later you will select the hardware configurations based on the requirements of each of the roles, based on your capacity, availability, and redundancy needs.

Start by reviewing the different reference architectures and figure out your farm structure, decide if you should split your solution across multiple farms, or federate some services, such as search, on a dedicated farm. See the [Reference Architectures](#_Reference_Standard_Architectures) section in this for more information.

#### SharePoint Server 2010 Technical Case Studies

Capacity management guidance for SharePoint Server 2010 includes a number of technical case studies of existing production environments that present a detailed description of existing SharePoint-based production environments. Additional technical case studies will be published over time; these can serve as a reference on how to design a SharePoint-based environment for specific purposes.

You can use these case studies as a reference while designing the architecture of your SharePoint solutions especially if you find the description of these deployment specific key differentiators similar to the demands and targets of the solution you are architecting.

These documents describe the following information for each documented case study:

* **Specifications**, such as hardware, farm topology and configuration;
* **Workload** including the user base, and the usage characteristics;
* **Dataset**, including contents sizes, content characteristics and content distribution
* **Health and performance** including a set of recorded indicators describing the farm's reliability and performance characteristics

For more information, download relevant documents from the [Performance and capacity technical case studies](http://technet.microsoft.com/en-us/library/cc261716(Office.14)aspx) page (<http://technet.microsoft.com/en-us/library/cc261716(Office.14)aspx>).

### Select your hardware

Selecting the right specifications for the machines in your farm is a crucial step to ensure proper reliability and performance of your deployment, one key concept to keep in mind is that you should plan for peak load and peak hours; in other words, when your farm is operating under average load conditions, there should be enough resources available to handle the greatest expected demand while still hitting latency and throughput targets.

The core capacity and performance hardware features of servers reflect four main categories: processing power, disk performance, network capacity, and memory capabilities of a system.

Another thing to consider is using virtualized machines. A SharePoint farm can be deployed using virtual machines. Although it has not been found to add any performance benefits, it does provide manageability benefits. Virtualizing SQL Server-based computers is generally not recommended, but there may be certain benefits to virtualizing the Web server and application server tiers. For more information, see [Virtualization planning](http://technet.microsoft.com/en-us/library/71c203cd-7534-47b0-9122-657d72ff0080(Office.14).aspx) (http://technet.microsoft.com/en-us/library/71c203cd-7534-47b0-9122-657d72ff0080(Office.14).aspx).

#### Hardware Selection Guidelines

##### Choosing Processors

SharePoint Server 2010 is only available for 64-bit processors. In general, more processors will enable you to serve higher demand.

In SharePoint Server 2010, individual Web servers will scale as you add more cores (we have tested up to 24 cores); the more cores the server has the more load it can sustain, all else being equal. In large SharePoint deployments, it is recommended to allocate either multiple 4-core Web servers (which can be virtualized), or fewer stronger (8-/16-/24-cores) Web servers.

Application servers' processor capacity requirements differ depending on the role of the server and the services it is running. Some SharePoint features demand greater processing power than others. For example, the Search Service is highly dependent on the processing power of the application server. For more information on the resource requirements of SharePoint features and services, see [Services and Features](#_Services_and_Features_1) earlier in this document.

The processor capacity requirements for SQL Server also depend on the service databases that a SQL Server-based computer is hosting. For more information on the typical behavior and requirements of each database, see [Storage and SQL Server capacity planning and configuration](http://technet.microsoft.com/en-us/library/%20a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx) (http://technet.microsoft.com/en-us/library/a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx).

##### Choosing Memory

Your servers will require varying amounts of memory, depending on server function and role. For example, servers running Search crawl components will process data more quickly if they have a large amount of memory because documents are read into memory for processing. Web servers that leverage many of the caching features of SharePoint Server 2010 may require more memory as well.

In general, Web server memory requirements are highly dependent on the number of application pools enabled in the farm and the number of concurrent requests being served. In most production SharePoint deployments, it is recommended to allocate at least 8 GB RAM on each Web server, with 16 GB recommended for servers with higher traffic or deployments with multiple application pools set up for isolation.

Application servers' memory requirements differ as well; some SharePoint features have greater memory needs on the application tier then others. In most production SharePoint deployments it is recommended to allocate at least 8 GB RAM on each application server; 16 GB, 32 GB and 64 GB application servers are common when many application services are enabled on the same server, or when services that are highly dependent on memory, such as the Excel Calculation Service and SharePoint Search Service, are enabled.

The memory requirements of database servers are tightly dependent on the database sizes. For more information on choosing memory for your SQL Server-based computers, see [Storage and SQL Server capacity planning and configuration](http://technet.microsoft.com/en-us/library/%20a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx) (http://technet.microsoft.com/en-us/library/a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx).

##### Choosing Networks

In addition to the benefit offered to users if clients have fast data access through the network, a distributed farm must have fast access for inter-server communication. This is particularly true when you distribute services across multiple servers or federate some of the services to other farms. There is significant traffic in a farm across the Web server tier, the application server tier, and the database server tier, and network can easily become a bottleneck under certain conditions like dealing with very large files or very high loads.

Web servers and application servers should be configured with at least two network interface cards (NICs): one NIC to handle end-user traffic and the other to handle the inter-server communication. Network latency between servers can have a significant impact on performance, so it is important to maintain less than 1 millisecond of network latency between the Web server and the SQL Server-based computers hosting the content databases. The SQL Server-based computers that host each service application database should be as close as possible to the consuming application server as well. The network between farm servers should have at least 1 Gbps of bandwidth.

##### Choosing Disks and Storage

Disk management is not simply a function of providing adequate space for your data. You must assess the on-going demand and growth, and ensure that the storage architecture is not slowing the system down. You should always ensure that you have at least 30 percent additional capacity on each disk, above your highest data requirement estimate, to leave room for future growth. Additionally, in most production environments, disk speed (IOps) is crucial to providing sufficient throughput to satisfy the servers' storage demands. You must estimate the amount of traffic (IOps) the major databases will require in your deployment and allocate enough disks to satisfy that traffic.

For more information on how to choose disks for database servers, see [Storage and SQL Server capacity planning and configuration](http://technet.microsoft.com/en-us/library/%20a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx) (http://technet.microsoft.com/en-us/library/a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx).

The Web and application servers have storage requirements as well. In most production environments, it is recommended to allocate at least 200 GB disk space for OS and temp and 150 GB of disk space for logs**.**

## Step 3: Pilot, Test and Optimize

The testing and optimization stage is a critical component of effective capacity management. You should test new architectures before you deploy them to production and you should conduct acceptance testing in conjunction with following monitoring best practices in order to ensure the architectures you design achieve the performance and capacity targets. This allows you to identify and optimize potential bottlenecks before they impact users in a live deployment. If you are upgrading from an Office SharePoint Server 2007 environment and plan to make architectural changes, or are estimating user load of the new SharePoint features, then testing particularly important to make sure your new SharePoint-based environment will meet performance and capacity targets.

Once you have tested your environment, you can analyze the test results to determine what changes need to be made in order to achieve the performance and capacity targets you established in Step 1.

These are the recommended sub steps you should follow for pre-production:

* Create the test environment that mimics the initial architecture you've designed.
* Populate the storage with the dataset or part of the dataset that you've identified in Step 1.
* Stress the system with synthetic load that represents the workload you've identified in Step 1.
* Run tests, analyze results, and optimize your architecture.
* Deploy your optimized architecture in your data center, and roll out a pilot with a smaller set of users.
* Analyze the pilot results, identify potential bottlenecks, and optimize the architecture. Retest if needed.
* Go live with full production.

### Test

#### Create a test plan

Verify that your plan includes:

* Hardware that is designed to operate at expected production performance targets. Always measure the performance of test systems conservatively.
* If you have custom code or custom component, it is important that you test the performance of those components in isolation first to validate their performance and stability. After they are stable, you should test the system with those components installed and compare performance to the farm *without* them installed. Custom components are often a major culprit of performance and reliability problems in production systems.
* Know the goal of your testing. Understand ahead of time what your testing objectives are. Is it to validate the performance of some new custom components that were developed for the farm? Is it to see how long it will take to crawl and index a set of content? Is it to determine how many requests per second your farm can support? There can be many different objectives during a test, and the first step in developing a good test plan is deciding what your objectives are.
* Understand how to measure for your testing goal. If you are interested in measuring the throughput capacity of your farm for example, you will want to measure the RPS and page latency. If you are measuring for search performance then you will want to measure crawl time and document indexing rates. If your testing objective is well understood, that will help you clearly define what key performance indicators you need to validate in order to complete your tests.

#### Create the Test Environment

Once your test objectives have been decided, your measurements have been defined, and you have determined what the capacity requirements are for your farm (from steps 1 and 2 of this process), the next objective will be to design and create the test environment. The effort to create a test environment is often underestimated. It should duplicate the production environment as closely as possible. Some of the features and functionality you should consider when designing your test environment include:

* **Authentication** – Decide whether the farm will use Active Directory Domain Services (AD DS), forms-based authentication (and if so with what directory), claims-based authentication, etc. Regardless of which directory you are using, how many users do you need in your test environment and how are you going to create them? How many groups or roles are you going to need and how will you create and populate them? You also need to ensure that you have enough resources allocated to your authentication services that they don't become a bottleneck during testing.
* **DNS** – Know what the namespaces are that you will need during your testing. Identify which servers will be responding to those requests and make sure you've included a plan that has what IP addresses will be used by which servers, and what DNS entries you will need to create.
* **Load balancing** – Assuming you are using more than one server (which you normally would or you likely wouldn't have enough load to warrant load testing), you will need some kind of load balancer solution. That could be a hardware load balancing device, or you could use software load balancing like Windows NLB. Figure out what you will use and write down all of the configuration information you will need to get it set up quickly and efficiently. Another thing to remember is that load test agents typically try and resolve the address to a URL only once every 30 minutes. That means that you should not use a local hosts file or round robin DNS for load balancing because the test agents will likely end up going to the same server for every single request, instead of balancing around all available servers.
* **Test servers** – When you plan your test environment, you not only need to plan for the servers for the SharePoint farm, you also need to plan for the machines needed to execute the tests. Typically that will include 3 servers at a minimum; more may be necessary. If you are using Visual Studio Team System (Team Test Load Agent) to do the testing, one machine will be used as the load test controller. There are generally 2 or more machines that are used as load test agents.   
  The agents are the machines that take the instructions from the test controller about what to test and issue the requests to the SharePoint farm. The test results themselves are stored on a SQL Server-based computer. You should **not** use the same SQL Server-based computer that is used for the SharePoint farm, because writing the test data will skew the available SQL Server resources for the SharePoint farm.  
  You also need to monitor your test servers when running your tests, the same way as you would monitor the servers in the SharePoint farm, or domain controllers, etc. to make sure that the test results are representative of the farm you're setting up. Sometimes the load agents or controller can become the bottleneck themselves. If that happens then the throughput you see in your test is typically not the maximum the farm can support.
* **SQL Server** –In your test environment, follow the guidance in the sections "Configure SQL Server" and "Validate and monitor storage and SQL Server performance" in the article [Storage and SQL Server capacity planning and configuration](http://technet.microsoft.com/en-us/library/%20a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx) (http://technet.microsoft.com/en-us/library/a96075c6-d315-40a8-a739-49b91c61978f(Office.14).aspx).
* **Dataset validation** – As you decide what content you are going to run tests against, remember that in the best case scenario you will use data from an existing production system. For example, you can back up your content databases from a production farm and restore them into your test environment, then attach the databases to bring the content into the farm. Anytime you run tests against made up or sample data, you run the risk of having your results skewed because of differences in your content corpus.

If you do have to create sample data, there are a few considerations to keep in mind as you build out that content:

* + All pages should be published; nothing should be checked out
  + Navigation should be realistic; don't build beyond what you would reasonably expect to use in production.
  + You should have an idea of the customizations the production site will be using. For example, master pages, style sheets, JavaScript, etc. should all be implemented in the test environment as closely as possible to the production environment.
  + Determine how many SharePoint groups and/or permission levels you are going to need, and how you are going to associate users with them.
  + Figure out whether you'll need to do profile imports, and how long that will take.
  + Determine whether you'll need Audiences, and how you'll create and populate them.
  + Determine whether you need additional search content sources, and what you will need to create them. If you won't need to create them, determine whether you'll have network access to be able to crawl them.
  + Determine whether you have enough sample data – documents, lists, list items, etc. If not, create a plan for how you will create this content.
  + Have a plan for enough unique content to adequately test search. A common mistake is to upload the same document – maybe hundreds or even thousands of times – to different document libraries with different names. That can impact search performance because the query processor will spend an ordinate amount of time doing duplicate detection that it wouldn't otherwise have to in a production environment with real content.

#### Create Tests and Tools

After the test environment is functional, it is time to create and fine-tune the tests that will be used to measure the performance capacity of the farm. This section will at times make references specifically to Visual Studio Team System (Team Test Load Agent), but many of the concepts are applicable irrespective of which load test tool you use. For more information about Visual Studio Team System, see [Visual Studio Team System](http://msdn.microsoft.com/en-us/library/fda2bad5.aspx) at MSDN (<http://msdn.microsoft.com/en-us/library/fda2bad5.aspx>).

A key criterion to the success of the tests is to be able to effectively simulate a realistic workload by generating requests across a wide range of the test site data, just as users would access a wide range of content in a production SharePoint farm. In order to do that, you will typically need to construct your tests such that they are data driven. Rather than creating hundreds of individual tests that are hard-coded to access a specific page, you should use just a few tests that use data sources containing the URLs for those items to dynamically access that set of pages.

In Visual Studio Team System (Team Test Load Agent), a data source can come in a variety of formats, but a CSV file format is often easiest to manage and transport between development and test environments. Keep in mind that creating CSV files with that content might require the creation of custom tools to enumerate the SharePoint-based environment and record the various URLs being used.

You may need to use tools for tasks like:

* Creating users and groups in Active Directory or other authentication store if you're using forms based authentication
* Enumerating URLs for sites, lists and libraries, list items, documents, etc. and putting them into CSV files for load tests
* Uploading sample documents across a range of document libraries and sites
* Creating site collections, webs, lists, libraries, folders and list items
* Creating My Sites
* Creating CSV files with usernames and passwords for test users; these are the user accounts that the load tests will execute as. There should be multiple files so that, for example, some contain only administrator users, some contain other users with elevated privileges (like author / contributor, hierarchy manager, etc.), and others are only readers, etc.
* Creating a list of sample search keywords and phrases
* Populating SharePoint groups and permission levels with users and Active Directory groups (or roles if you are using forms based authentication)

When creating the web tests, there are other best practices that you should observe and implement. They include:

* Record simple web tests as a starting point. Those tests will have hard-coded values in them for parameters like URL, ID's, etc. Replace those hard-coded values with links from your CSV files. Data binding those values in Visual Studio Team System (Team Test Load Agent) is extremely easy.
* Always have validation rules for your test. For example, when requesting a page, if an error occurs you will often get the error.aspx page in response. From a web test perspective it appears as just another positive response, because you get an HTTP status code of 200 (successful) in the load test results. Obviously an error has occurred though so that should be tracked differently. Creating one or more validation rules allows you to trap when certain text is sent as a response so that the validation fails and the request is marked as a failure. For example, in Visual Studio Team System (Team Test Load Agent) a simple validation rule might be a ResponseUrl validation – it records a failure if the page that is rendered after redirects is not the same response page that was recorded in the test. You could also add a FindText rule that will record a failure if it finds the word "access denied", for example, in the response.
* Use multiple users in different roles for tests. Certain behaviors such as output caching work differently depending on the rights of the current user. For example, a site collection administrator or an authenticated user with approval or authoring rights will not get cached results because we always want them to see the most current version of content. Anonymous users, however, will get the cached content. You need to make sure that your test users are in a mix of these roles that approximately matches the mix of users in the production environment. For example, in production there are probably only two or three site collection administrators, so you should not create tests where 10% of the page requests are made by user accounts that are site collection administrators over the test content.
* Parsing dependent requests is an attribute of a Visual Studio Team System (Team Test Load Agent) that determines whether the test agent should attempt to retrieve just the page, or the page and all associated requests that are part of the page, such as images, stylesheets, scripts, etc. When load testing, we usually ignore these items for a few reasons:
  + After a user hits a site the first time these items are often cached by the local browser
  + These items don't typically come from SQL Server in a SharePoint-based environment. With BLOB caching turned on, they are instead served by the Web servers so they don't generate SQL Server load.

If you regularly have a high percentage of first time users to your site, or you have disabled browser caching, or for some reason you don't intend to use the blob cache, then it may make sense to enable parsing dependent requests in your tests. However this is really the exception and not the rule of thumb for most implementations. Be aware that if you do turn this on it can significantly inflate the RPS numbers reported by the test controller. These requests are served so quickly it may mislead you into thinking that there is more capacity available in the farm than there actually is.

* Remember to model client application activity as well. Client applications, such as Microsoft Word, PowerPoint, Excel and Outlook generate requests to SharePoint farms as well. They add load to the environment by sending the server requests such as retrieving RSS feeds, acquiring social information, requesting details on site and list structure, synchronizing data, etc. These types of requests should be included and modelled if you have those clients in your implementation.
* In most cases a web test should only contain a single request. It's easier to fine-tune and troubleshoot your testing harness and individual requests if the test only contains a single request. Web tests will typically need to contain multiple requests if the operation it is simulating is composed of multiple requests. For example, to test this set of actions you will need a test with multiple step: checking out a document, editing it, checking it in and publishing it. It also requires reserving state between the steps – for example, the same user account should be used to check it out, make the edits, and check it back in. Those multi-step operations that require state to be carried forward between each step are best served by multiple requests in a single web test.
* Test each web test individually. Make sure that each test is able to complete successfully before running it in a larger load test. Confirm that all of the names for web applications resolve, and that the user accounts used in the test have sufficient rights to execute the test.

Web tests comprise the requests for individual pages, uploading documents, view list items, etc. All of these are pulled together in load tests. A load test is where you plug in all of the different web tests that are going to be executed. Each web test can be given a percentage of time that it will execute – for example, if you find that 10% of requests in a production farm are search queries, then in the load test you would configure a query web test to run 10% of the time. In Visual Studio Team System (Team Test Load Agent), load tests are also how you configure things like the browser mix, network mix, load patterns, and run settings. There are some additional best practices that should be observed and implemented for load tests:

* Use a reasonable read/write ratio in your tests. Overloading the number of writes in a test can significantly impact the overall throughput of a test. Even on collaboration farms, the read/write ratios tend to have many more reads than writes. For more information, see the [Performance and capacity technical case studies](http://technet.microsoft.com/en-us/library/cc261716(Office.14)aspx) page (<http://technet.microsoft.com/en-us/library/cc261716(Office.14)aspx)>.
* Consider the impact of other resource intensive operations and decide whether they should be occurring during the load test. For example, operations like backup and restore are not generally done during a load test. A full search crawl may not be usually run during a load test, whereas an incremental crawl may be normal. You need to consider how those tasks will be scheduled in production – will they be running at peak load times? If not, then they should probably be excluded during load testing, when you are trying to determine the maximum steady state load you can support for peak traffic.
* Don't use think times. Think times are a feature of Visual Studio Team System (Team Test Load Agent) that allow you to simulate the time that users pause between clicks on a page. For example a typical user might load a page, spend three minutes reading it, then click a link on the page to visit another site. Trying to model this in a test environment is nearly impossible to do correctly, and effectively doesn't add value to the test results. It's difficult to model because most organizations don't have a way to monitor different users and the time they spend between clicks on different types of SharePoint sites (like publishing versus search versus collaboration, etc.). It also doesn't really add value because even though a user may pause between page requests, the SharePoint-based servers do not. They just get a steady stream of requests that may have peaks and valleys over time, but they are not waiting idly as each user pauses between clicking links on a page.
* Understand the difference between users and requests. Visual Studio Team System (Team Test Load Agent) has load pattern where it asks you to enter the number of users to simulate. This doesn't have anything to do with application users, it's really just how many threads are going to be used on the load test agents to generate requests. A common mistake is thinking that if the deployment will have 5,000 users for example, then 5,000 is the number that should be used in Visual Studio Team System (Team Test Load Agent) – it is not! That's one of the many reasons why when estimating capacity planning requirements, the usage requirements should be based on number of requests per second and not number of users. In a Visual Studio Team System (Team Test Load Agent) load test, you will find that you can often generate hundreds of requests per second using only 50 to 75 load test "users".
* Use a constant load pattern for the most reliable and reproducible test results. In Visual Studio Team System (Team Test Load Agent) you have the option of basing load on a constant number of users (threads, as explained in the previous point), a stepped up load pattern of users, or a goal based usage test. A stepped load pattern is when you start with a lower number of users and then "step up" adding additional users every few minutes. A goal based usage test is when you establish a threshold for a certain diagnostic counter, like CPU utilization, and test attempts to drive the load to keep that counter between a minimum and maximum threshold that you define for it. However, if you are just trying to determine the maximum throughput your SharePoint farm can sustain during peak load, it is more effective and accurate to just pick a constant load pattern. That allows you to more easily identify how much load the system can take before starting to regularly exceed the thresholds that should be maintained in a healthy farm.

Each time you run a load test remember that it is changing data in the database. Whether that's uploading documents, editing list items, or just recording activity in the usage database, there will be data that is written to SQL Server. To ensure a consistent and legitimate set of test results from each load test, you should have a backup available before you run the first load test. After each load test is complete the backup should be used to restore the content back to the way it was before the test was started.

### Optimize

If you cannot meet your capacity and performance targets by scaling your farm hardware or making changes to the topology, you may need to consider revising your solution. For example, if your initial requirements were for a single farm for collaboration, Search and Social, you may need to federate some of the services such as search to a dedicated services farm, or split the workload across more farms, maybe use a dedicated farm for social and another for team collaboration.

### Deploy the pilot environment

Before you deploy SharePoint Server 2010 to a production environment, it is important that you first deploy a pilot environment and thoroughly test the farm to ensure that it can meet capacity and performance targets for your expected peak load. We recommend that the pilot environment is first tested with synthetic load particularly for large deployments, and then stressed by a small set of live users and live content. The benefit of analyzing a pilot environment with a small set of live users is the opportunity to validate some of the assumptions you made about the usage characteristics and the content growth before you go fully into production.

## Step 4: Deploy

Once you have executed your final round of tests and confirmed that the architecture you have selected can achieve the performance and capacity targets you established in Step 1, you can deploy your SharePoint-based environment to production.

The appropriate rollout strategy will vary depending upon the environment and situation. While SharePoint deployment in general is outside the scope of this document, there are certain suggested activities that may come out of the capacity planning exercise. Here are some examples:

* **Deploying a new SharePoint farm:** The capacity planning exercise should have guided and confirmed plans for a design and deployment of SharePoint Server 2010. In this case, the rollout will be the first broad deployment of SharePoint Server 2010. It will require moving or rebuilding the servers and services that were used during the capacity planning exercises into production. This is the most straight-forward scenario because there aren't any upgrades or modifications needed to an existing farm.
* **Upgrading an Office SharePoint Server 2007 farm to SharePoint Server 2010:** The capacity planning exercise should have validated the design for a farm that can meet existing demands and scale up to meet increased demand and usage of a SharePoint 2010 farm. Part of the capacity planning exercise should have included test migrations to validate how long the upgrade process will take, whether any custom code needs to be modified or replaced, whether any third-party tools need updating, etc. At the conclusion of capacity planning you should have a validated design, and understanding of how much time it will take to upgrade, and a plan for how best to work through the upgrade process – for example, an in-place upgrade, or migrating content databases into a new farm. If you're doing an in-place upgrade then during capacity planning you may have found that additional or upgraded hardware will be needed, and considerations for downtime. Part of the output from the planning exercise should be a list of the hardware changes that are needed and a detailed plan for deploying the hardware changes to the farm first. Once the hardware platform that was validated during capacity planning is in place, you can move forward with the process of upgrading to SharePoint Server 2010.
* **Improving the performance of an existing SharePoint 2010 farm:** The capacity planning exercise should have helped you to identify the bottlenecks in your current implementation, devise ways to reduce or eliminate those bottlenecks, and validate an improved implementation that meets your business requirements for SharePoint services. There are different ways in which performance issues could have been resolved, from something as simple as reallocating services across existing hardware, upgrading existing hardware, or adding additional hardware and adding additional services to it. The different approaches should be tested and validated during the capacity planning exercise, and then a deployment plan formulated depending on the results of that testing.

## Step 5: Maintain

### Monitor and Adjust

To maintain system performance, you must monitor your server to identify potential bottlenecks. Before you can monitor effectively, you must understand the key indicators that will tell you if a specific part of your farm requires attention, and know how to interpret these indicators. If you find that your farm is operating outside the targets you have defined, you can adjust your farm by adding or removing hardware resources, modifying your topology, or changing how data is stored.

The information in this section is intended to help administrators manually configure performance counters and other settings. For more information about health monitoring and troubleshooting using the health monitoring tools built into the SharePoint Central Administration interface, read the following:

* [Health monitoring](http://technet.microsoft.com/en-us/library/ee681489(office.14).aspx)) (http://technet.microsoft.com/en-us/library/ee681489(office.14).aspx)
* [Solving problems and troubleshooting](http://technet.microsoft.com/en-us/library/ee748639(office.14).aspx) (http://technet.microsoft.com/en-us/library/ee748639(office.14).aspx)

Below is a list of the settings that you can modify to monitor your environment in its early stages, which will help you determine if any changes are needed. Keep in mind that increasing your monitoring capabilities will affect the amount of disk space that your usage database will require. Once the environment is stable and this detailed monitoring is no longer required, you may want to reverse the settings below to their defaults.

| Setting | Value | Notes |
| --- | --- | --- |
| Event Log Flooding Protection | Disabled | The default is Enabled. It can be disabled to collect as much monitoring data as possible. For normal operations, it should be enabled. |
| **Timer Job Schedule** |  |  |
| Microsoft SharePoint Foundation Usage Data Import | 5 minutes | The default is 30 minutes. Lowering this setting imports the data into the usage database more frequently, and is particularly useful when troubleshooting. For normal operations, it should be 30 minutes. |
| **Diagnostic Providers** |  |  |
| Enable all diagnostic providers | Enabled | The default is Disabled except for the "*Search Health Monitoring - Trace Events*" provider. These providers collect health data for various features and components. For normal operations, you may want to revert to the default. |
| Set "*job-diagnostics-performance-counter-wfe-provide*r" and "*job-diagnostics-performance-counter-sql-provider*" Schedule Intervals | 1 minute | The default is 5 minutes. Lowering this setting can poll data more frequently, and is particularly useful when troubleshooting. For normal operations, it should be 5 minutes. |
| **Miscellaneous** |  |  |
| Enable stack tracing for content requests | Enabled | The default is Disabled. Enabling this setting allows diagnosis of content requests failures using the process stack trace. For normal operations, it should be disabled. |
| Enable the Developer Dashboard | Enabled | The default is Disabled. Enabling this setting allows diagnosis of slow pages, or other problems by using the Developer Dashboard. For normal operations, and once troubleshooting is no longer necessary, it should be disabled. |
| **Usage Data Collection** |  |  |
| * Content Import Usage * Content Export Usage * Page Requests * Feature Use * Search Query Use * Site Inventory Usage * Timer Jobs * Rating Usage | Enabled | Enabling the logging of this set of counters allows you to collect more usage data across the environment and to better understand the traffic patterns in the environment. |

#### Performance Counters

If you are making use of the usage database, then you can add the performance counters that assist you in monitoring and evaluating your farm's performance to the usage database, such that they are logged automatically at a specific interval (30 minutes by default). Given that, you can query the usage database to retrieve these counters and graph the results over time. Here's an example of using the Add-SPDiagnosticsPerformanceCounter cmdlet to add the % Processor Time counter to the usage database. This only needs to be run on one of the Web servers:

Add-SPDiagnosticsPerformanceCounter -Category "Processor" -Counter "% Processor Time" -Instance "\_Total" -WebFrontEnd

There are a number of generic performance counters that you should monitor for any server system. The following table outlines these performance counters.

| Performance Counter | Description |
| --- | --- |
| Processor | You should monitor processor performance to ensure that all processor usage does not remain consistently high (over 80 percent) as this indicates that the system would not be able to handle any sudden surges of activity. And that in the common state, you will not see a domino effect if one component failure will bring the remaining components to a malfunctioning state, example – if you have 3 Web servers, you should make sure the average CPU across all servers is under 60%, this way if one fails there is still room for the other two to pick up the extra load. |
| Network Interface | Monitor the rate at which data is sent and received via the network interface card. This should remain below 50 percent of network capacity. |
| Disks and Cache | There are a number of logical disk options that you should monitor regularly. The available disk space is essential in any capacity study, but you should also review the time that the disk is idle. Dependent on the types of applications or services you are running on your servers, you may review disk read and write times. Extended queuing for write or read function will affect performance. The cache has a major effect on read and write operations. You must monitor for increased cache failures. |
| Memory and Paging File | Monitor the amount of physical memory available for allocation. Insufficient memory will lead to excessive use of the page file and an increase in the number of page faults per second. |

#### System Counters

The following table provides information on system objects and counters that you could add to the set of counters monitored in the usage database using the SPDiagnosticPerformanceCounter on a web server.

| Objects and Counters | Description |
| --- | --- |
| **Processor** |  |
| % Processor Time | This shows processor usage over a period of time. If this is consistently too high, you may find performance is adversely affected. Remember to count "Total" in multiprocessor systems. You can measure the utilization on each processor as well, to ensure balanced performance between cores. |
| **Disk** |  |
| - Avg. Disk Queue Length | This shows the average number of both read and write requests that were queued for the selected disk during the sample interval. A bigger disk queue length may not be a problem as long as disk reads/writes are not suffering and the system is working in a steady state without expanding queuing. |
| Avg. Disk Read Queue Length | The average number of read requests that are queued. |
| Avg. Disk Write Queue Length | The average number of write requests that are queued. |
| Disk Reads/sec | The number of reads to disk per second. |
| Disk Writes/sec | The number of writes to disk per second. |
| **Memory** |  |
| - Available Mbytes | This shows the amount of physical memory available for allocation. Insufficient memory will lead to excessive use of the page file and an increase in the number of page faults per second. |
| - Cache Faults/sec | This counter shows the rate at which faults occur when a page is sought in the file system cache and is not found. This may be a soft fault, when the page is found in memory, or a hard fault, when the page is on disk.  The effective use of the cache for read and write operations can have a significant effect on server performance. You must monitor for increased cache failures, indicated by a reduction in the **Async Fast Reads/sec** or **Read Aheads/sec**. |
| - Pages/sec | This counter shows the rate at which pages are read from or written to disk to resolve hard page faults. If this rises, it indicates system-wide performance problems. |
| **Paging File** |  |
| - % Used and % Used Peak | The server paging file, sometimes called the *swapfile*, holds "virtual" memory addresses on disk. Page faults occur when a process has to stop and wait while required "virtual" resources are retrieved from disk into memory. These will be more frequent if the physical memory is inadequate. |
| **NIC** |  |
| - Total Bytes/sec | This is the rate at which data is sent and received via the network interface card. You may need to investigate further if this rate is over 40-50 percent network capacity. To fine-tune your investigation, monitor **Bytes received/sec** and **Bytes Sent/sec**. |
| **Process** |  |
| - Working Set | This counter indicates the current size (in bytes) of the working set for a given process. This memory is reserved for the process, even if it is not in use. |
| - % Processor Time | This counter indicates the percentage of processor time that is used by a given process. |
| Thread Count (\_Total) | The current number of threads. |
| **ASP.NET** |  |
| Requests Total | The total number of requests since the service was started. |
| Requests Queued | Microsoft® SharePoint® Foundation 2010 provides the building blocks for HTML pages that are rendered in the user browser over HTTP. This counter shows the number of requests waiting to be processed. |
| Request Wait Time | The number of milliseconds that the most recent request waited in the queue for processing. As the number of wait events increases, users will experience degraded page-rendering performance. |
| Requests Rejected | The total number of requests not executed because of insufficient server resources to process them. This counter represents the number of requests that return a 503 HTTP status code, indicating that the server is too busy. |
| Requests Executing (\_Total) | The number of requests currently executing. |
| Requests/Sec (\_Total) | The number of requests executed per second. This represents the current throughput of the application. Under constant load, this number should remain within a certain range, barring other server work (such as garbage collection, cache cleanup thread, external server tools, and so on). |
| **.NET CLR Memory** |  |
| # Gen 0 Collections | Displays the number of times the generation 0 objects (that is, the youngest, most recently allocated objects) are garbage collected since the application started. This number is useful as a ratio of #Gen 0: #Gen 1: #Gen 2 to make sure that the number of Gen 2 collections does not greatly exceed Gen 0 collections, optimally by a factor of 2. |
| # Gen 1 Collections | Displays the number of times the generation 1 objects are garbage collected since the application started. |
| # Gen 2 Collections | Displays the number of times the generation 2 objects are garbage collected since the application started. The counter is incremented at the end of a generation 2 garbage collection (also called a full garbage collection). |
| % Time in GC | Displays the percentage of elapsed time that was spent performing a garbage collection since the last garbage collection cycle. This counter usually indicates the work done by the garbage collector to collect and compact memory on behalf of the application. This counter is updated only at the end of every garbage collection. This counter is not an average; its value reflects the last observed value. This counter should be under 5% in normal operation. |

#### SQL Server Counters

The following table provides information on SQL Server Counters objects and counters.

| Objects and Counters | Description |
| --- | --- |
| **General Statistics** | This object provides counters to monitor general server-wide activity, such as the number of current connections and the number of users connecting and disconnecting per second from computers running an instance of SQL Server. |
| User Connections | This counter shows the amount of user connections on your instance of SQL Server. If you see this number rise by 500 percent from your baseline, you may see a performance reduction. |
| **Databases** | This object provides counters to monitor bulk copy operations, backup and restore throughput, and transaction log activities. Monitor transactions and the transaction log to determine how much user activity is occurring in the database and how full the transaction log is becoming. The amount of user activity can determine the performance of the database and affect log size, locking, and replication. Monitoring low-level log activity to gauge user activity and resource usage can help you to identify performance bottlenecks. |
| Transactions/sec | This counter shows the amount of transactions on a given database or on the entire SQL Server instance per second. This number is to help you create a baseline and to help you troubleshoot issues. |
| **Locks** | This object provides information about SQL Server locks on individual resource types. |
| Number of Deadlocks/sec | This counter shows the number of deadlocks on the SQL Server per second. This should be 0. |
| Average Wait Time (ms) | This counter shows the average amount of wait time for each lock request that resulted in a wait. |
| Lock Wait Time (ms) | This counter shows the total wait time for locks in the last second. |
| Lock Waits/sec | This counter shows the number of locks per second that could not be satisfied immediately and had to wait for resources. |
| **Latches** | This object provides counters to monitor internal SQL Server resource locks called latches. Monitoring the latches to determine user activity and resource usage can help you to identify performance bottlenecks. |
| Average Latch Wait Time (ms) | This counter shows the average latch wait time for latch requests that had to wait. |
| Latch Waits/sec | This counter shows the number of latch requests per second that could not be granted immediately. |
| **SQL Statistics** | This object provides counters to monitor compilation and the type of requests sent to an instance of SQL Server. Monitoring the number of query compilations and recompilations and the number of batches received by an instance of SQL Server gives you an indication of how quickly SQL Server is processing user queries and how effectively the query optimizer is processing the queries. |
| SQL Compilations/sec | This counter indicates the number of times the compile code path is entered per second. |
| SQL Re-Compilations/sec | This counter indicates the number of times statement recompiles are triggered per second. |
| **Plan Cache** | This object provides counters to monitor how SQL Server uses memory to store objects such as stored procedures, ad hoc and prepared Transact-SQL statements, and triggers. |
| Cache Hit Ratio | This counter indicates the ratio between cache hits and lookups for plans. |
| **Buffer Cache** | This object provides counters to monitor how SQL Server uses memory to store data pages, internal data structures, and the procedure cache, as well as counters to monitor the physical I/O as SQL Server reads and writes database pages. |
| Buffer Cache Hit Ratio | This counter shows the percentage of pages found in the buffer cache without having to read from disk. The ratio is the total number of cache hits divided by the total number of cache lookups since an instance of SQL Server was started. |

### Removing Bottlenecks

System bottlenecks represent a point of contention where there are insufficient resources to service user transaction requests. These may be physical hardware, operating environment, or application-based. Often, the reason for the bottleneck will be inefficient custom code or 3rd party solutions, and a review of those could yield better results than adding hardware. Another common cause for bottlenecks is a misconfiguration of the farm, or an inefficient solution implementation that structures data in a way that requires more resources than necessary. For a system administrator, it is essential to manage bottlenecks by constantly monitoring performance. When you identify a performance issue, you must assess the best resolution for removing the bottleneck. The Performance counters and other performance monitoring applications, such as System Center Operations Manager (SCOM), are the key tools in tracking and analyzing problems, so that you can develop a solution.

#### Physical Bottleneck Resolution

Physical bottlenecks are based on processor, disk, memory, and network contention: too many requests are contending for too few physical resources. The objects and counters described in the *Monitoring Performance* topic indicate where the performance problem is located, for example, hardware processor or ASP.NET. Bottleneck resolution requires that you identify the issue and then make a change or changes that mitigate the performance problem.

Problems seldom happen instantaneously; there is usually a gradual performance degradation that you can track if you monitor regularly, using your Performance tool or a more sophisticated system, such as SCOM. For both of these options, to varying degrees, you can embed solutions within an alert, in the form of advisory text or scripted commands.

You may have to resolve bottleneck issues by making changes to hardware or system configurations, once you have determined that they are not caused by a misconfiguration, inefficient custom code or third party solutions, or inefficient solution implementation. The following tables identify problem threshold and possible resolution options. Some of the options suggest hardware upgrades or modifications.

| Objects and Counters | Problem | Resolution Options |
| --- | --- | --- |
| Processor -  % Processor Time | Over 75-85% | Upgrade processor  Increase number of processors  Add additional server(s) |
| **Disk** |  |  |
| Avg. Disk Queue Length | Gradually increasing, system not in a steady state and queue is backing up | Increase number or speed of disks  Change array configuration to stripe  Move some data to an alternative server |
| % Idle Time | Greater than 90% | Increase number of disks  Move data to an alternative disk or server |
| % Free Space | Less than 30% | Increase number of disks  Move data to an alternative disk or server |
| **Memory** |  |  |
| Available Mbytes | Less than 2GB on a Web server. | Add memory.  *Note: SQL server available memory will be low, by design, and does not always indicate a problem.* |
| Cache Faults/sec | Greater than 1 | Add memory  Increase cache speed or size if possible  Move data to an alternative disk or server |
| Pages/sec | Greater than 10 | Add memory |
| **Paging File** |  |  |
| % Used and % Used Peak | The server paging file, sometimes called the *swapfile*, holds "virtual" memory addresses on disk. Page faults occur when a process has to stop and wait while required "virtual" resources are retrieved from disk into memory. These will be more frequent if the physical memory is inadequate. | Add memory |
| **NIC** |  |  |
| Total Bytes/sec | Over 40-50% of network capacity. This is the rate at which data is sent and received via the network interface card. | Investigate further by monitoring **Bytes received/sec** and **Bytes Sent/sec**.  Reassess network interface card speed  Check number, size, and usage of memory buffers |
| **Process** |  |  |
| Working Set | Greater than 80% of total memory | Add memory |
| % Processor Time | Over 75-85%. | Increase number of processors  Redistribute workload to additional servers |
| **ASP.NET** |  |  |
| Application Pool Recycles | Several per day, causing intermittent slowness. | Make sure that you have not implemented settings that automatically recycle the application pool unnecessarily throughout the day. |
| Requests Queued | Hundreds or thousands of requests queued. | Implement additional Web servers  The default maximum for this counter is 5,000, and you can change this setting in the Machine.config file |
| Request Wait Time | As the number of wait events increases, users will experience degraded page rendering performance. | Implement additional Web servers |
| Requests Rejected | Greater than 0 | Implement additional Web servers |