WRITTEN BY Eric Shupps

10 STEPS TO OPTIMIZE SHAREPOINT PERFORMANCE

INTRODUCTION Microsoft SharePoint is the fastest growing product in Microsoft history. Its adoption rate has been exponential with millions of documents being stored daily. With its growth critical documents and procedures are now being stored in SharePoint. It is fundamental that SharePoint maintains its healthy status, and application performance is a key component for a successful deployment and adoption of SharePoint.

This white paper was written by Eric Shupps, a renowned SharePoint MVP that specializes in SharePoint performance to help organizations better understand the different ways and places SharePoint performance can be improved. SharePoint deployments are a combination of applications, services and databases where delivering a scalable, dynamic and efficient portal implementation can be a challenging task for even the most experienced IT professionals. Improvements can be made within or outside of the SharePoint infrastructure.

idera™

STEP 1 Separate User and Database Traffic

A common misconception is that servers connected to a high-speed network segment (such as gigabit Ethernet or Fiber) will have plenty of bandwidth to perform all required operations. While it is unlikely that a high-speed link will ever be fully saturated at any given moment, it is quite possible that physical resource contention may have a direct impact on the performance of a specific operation. SharePoint places a tremendous amount of demand on SQL—each request for a page can result in numerous calls to the database, not to mention the additional overhead required by service jobs, search indexing, and other operations.

In order to mitigate the conflict between user and database traffic, connectivity between front-end servers and SQL should be isolated, either via separate physical networks or virtual LAN's. Typically, this requires at least two separate network interface cards in each front end web server with static routes configured to insure traffic is routed to the correct interface. The same configuration may also be applied to application and index server, although the perceived benefits will be less as these server roles typically do not involve much direct user traffic (environments which utilize InfoPath Forms Services and Excel Services will benefit the most).

The following is an example of a typical configuration optimized for data connectivity. In this scenario, Servers A functions as a Web Front End, while B performs index operations, C is an Active Directory domain controller and D hosts the SQL database:

Server A	→	**NIC 1**	→	**Network 172.16.x.x**	→	**Corporate Network/Internet**
	→	**NIC 2**	→	**Network 192.168.x.x**	→	**Active Directory (Server C)**
	→	**NIC 3**	→	**Network 10.x.x.x**	→	**SQL (Server D)**
Server B	→	**NIC 1**	→	**Network 172.16.x.x**	→	**Corporate Network/Internet**
	→	**NIC 2**	→	**Network 192.168.x.x**	→	**Active Directory (Server C)**
	→	**NIC 3**	→	**Network 10.x.x.x**	→	**SQL (Server D)**
Server D	→	**NIC 1**	→	**Network 192.168.x.x**	→	**Active Directory (Server C)**
	→	**NIC 2**	→	**Network 10.x.x.x**	→	**SharePoint (Servers A&B)**

An important issue that must be taken into account when considering the separation of WFE/Index servers and the SQL database is the use of firewalls. Network security policies often dictate the isolation of database servers behind a firewall which restricts communication to only a handful of necessary ports. Caution should be taken in this type of configuration to insure that the firewall ports operate at sufficient speed to insure optimal data transfer and that only the minimal amount of required filtering and packet analysis are being performed. SharePoint applications are very data intensive and the database connection can often be the most significant bottleneck; whenever this connection is impeded by a firewall, thorough load and scalability testing should be performed to assess the impact on portal operations under heavy use.

STEP 2 Isolate Search Indexing

A typical medium server farm consists of one or more web front end servers, a dedicated index or application server and a separate SQL database server. In this scenario, search traffic initiated by the index server must be processed by the same servers responsible for delivering end-user content. The crawl process can create a tremendous amount of network traffic as the Search service must make a number of HTTP requests for each page in the hierarchy. Failure to isolate this traffic can have a negative impact on performance as search requests conflict with user traffic competing for the same limited amount of physical bandwidth.

In order to prevent search and user traffic from conflicting with each other, an additional server may be added to the farm which is dedicated solely to servicing search queries (alternatively, in smaller environments, the index server may also serve this function). The farm administrator would then configure the Search service to perform crawls only against this dedicated server, thereby eliminating excessive load on the web front end servers during index operations and reducing the amount of potentially conflicting requests. Depending upon the scope and frequency of search crawls, this configuration can reduce traffic to the web front end servers by as much as 70% during index operations.

STEP 3 Adjust SQL Parameters

SQL performance tuning is a discipline unto itself but there are some simple things that SharePoint farm administrators can do to improve performance without requiring the services of an experienced DBA. To begin with, the implementation of SQL should be planned at least as carefully as the SharePoint farm itself; perhaps more, considering the level of detail that can be involved. Physical hardware resources, network connectivity, disk size and speed, location of data files, configuration of shared storage—all aspects must be taken into consideration based on the size of the farm and the projected amount of data.

One quick way to avoid future headaches is to provision the major SharePoint databases onto separate physical disks (or LUNs if a SAN is involved). This means one set of disks for search databases, one for temporary databases, and still another for content databases (depending upon the size of the individual content databases these may require further separation). SharePoint is both read and write intensive, so separating the I/O operations onto separate disks prevents excessive thrashing and cache hits. Additional consideration should be given to isolating the log files (*.ldf); although these do not incur the same level of I/O as other files, they do play a primary role in backup and recovery and, because they can grow to several times the size of the master database files, can consume a great deal of disk space.

Another simple optimization technique is to proactively manage the size and growth of individual databases. By default, SQL grows database files in small increments, either 1MB at a time or as a fixed percentage of database size (usually 10%). These settings can cause SQL to waste cycles constantly expanding databases, especially in larger environments which utilize a great deal of storage space, and prevents further data from being written while the databases are expanding. An alternative approach is to first pre-size the databases up to the maximum recommended size (100GB) if space is available and set autogrowth to a fixed size (e.g. 10MB or 20MB). This will prevent SQL from expanding databases unnecessarily and insure that growth happens in a manageable fashion.

Finally, autogrowth, and its corollary, autoshrink, are prone to producing excessive fragmentation, especially when done in small increments. Even on fast disks fragmentation can have a substantially negative impact on performance, a situation which may be compounded by complex RAID configurations and distributed SAN storage. Disk defragmentation should be scheduled on a frequent basis to insure that SQL is using resources effectively.

STEP 4 Defragment Database Indexes

SQL Server maintains its own set of indexes for data stored in various databases in order to improve query efficiency and read operations. Just as with files stored on disk, these indexes can become fragmented over time as new INSERT, DELETE and UPDATE operations are performed. This can have a particular effect on SharePoint as many of the behind-the-scenes query operations are not optimized for large datasets (such as list views encompassing hundreds of thousands of items).

Index fragmentation is a complex topic best left to SQL DBA's but it is important for administrators to plan for regular maintenance operations which include index defragmentation. Special care should be taken to schedule these types of operations in a maintenance window as they are both resource-intensive and, in many cases, blocking tasks which prevent data from being written to or read from the indexes as they are being rebuilt. Service Pack 2 for SharePoint Server 2007 and WSS v3.0 contains updates to the database index management routines functionality and scheduling but enterprise administrators who manage large farms may wish to implement their own index defragmentation maintenance tasks and schedules.

STEP 5 Distribute User Data Across Multiple Content Databases

Most SharePoint data is stored in lists: Tasks, Announcements, Document Libraries, Issues, Picture Libraries, and so forth. A great deal of this data is actually stored in a single table in the content database associated with the site collection. Regardless of how many sites and subsites are created within the SharePoint hierarchy, each Site Collection has only one associated Content Database. This means that a Site Collection with thousands of subsites is storing the bulk of the user data from every list in every site in a single table in SQL.

This can lead to a delay in the time it takes to insert items, render views, and navigate folders in any given list, as SQL must recursively execute the various queries over one potentially very large dataset. One way to reduce the workload is to manage the mapping of site collections to content databases. Administrators can use the Central Administration interface to pre-stage content databases to insure that site collections are associated with a single database or grouped logically based on size or priority. By adjusting the Maximum Number of Sites setting or changing database status to 'offline', administrations can also control which content database is used when new site collections are created.

STEP 6 Minimize Page Size

The SharePoint user interface is intended to make it easy for information workers to manage content and find resources. For users connected to the portal over the corporate LAN this doesn't pose much of a problem but for disconnected users on slower WAN links or public facing websites the heavyweight nature of the typical SharePoint page can prove to be a real performance-killer. Fortunately, there are a number of ways to reduce the bloat and trim those pages down to a more manageable size.

First, when designing a new look and feel, it is helpful to start with a minimal master page. The master page contains all the chrome and navigation elements typically found on the top and left of a standard SharePoint page—things like the search box, global navigation, Quick Launch, Site Actions menu and other visual element. A minimal master page, as the name implies, removes unnecessary elements and allows designers to start with a clean slate that only contains the base functionality required for the page to render correctly.

Second, most SharePoint pages contain links to supporting files, including javascript and style sheets, which require additional time to retrieve and execute. Designers can alter the way in which SharePoint pages retrieve these files through a technique called "delayed loading", which essentially loads the linked files in the background while the rest of the page is rendering, allowing users to view content without waiting for all the back-end processing to take place.

Finally, there are programmatic ways to reduce page sizes in SharePoint which developers can implement relatively easily. One technique is the elimination of whitespace—dead weight which makes the page more readable to developers and designers but which has no impact on the ability of a client browser to render the page. Another method is to remove specific javascript and authoring content entirely based upon a user's permission level. Developers may also wish to disable ViewState persistence for specific page objects and implement custom server controls which leverage caching mechanisms that are more efficient than the default cache objects in SharePoint.

STEP 7 Configure IIS Compression

SharePoint content consists of two primary sources—static files resident in the SharePoint root directories (C:\Program Files\Common Files\Microsoft Shared\12 for 2007 and \14 for 2010) and dynamic data stored in the content database (web parts, publishing field controls, navigation controls, etc.). At runtime, SharePoint merges the page contents from both sources prior to transmitting them inside an HTTP response to the requesting user. Internet Information Server versions 6 (Windows Server 2003) and 7 (Windows Server 2008) both contain various mechanisms for reducing the payload of HTTP responses prior to transmitting them across the network. Adjusting these settings can reduce the size of the data transmitted to the client, resulting in shorter load times and faster page rendering. It is important to note that doing so can have an impact on CPU load and memory utilization – care should be taken to insure that sufficient hardware resources exist before tuning IIS compression settings.

IIS compression settings can be modified from a base value of 0 (no compression) to a maximum value of 10 (full compression). Adjusting this setting determines how aggressive IIS should be in executing the compression algorithms. Setting the value to 4 can result in a static linked file such as core.js being reduced in size from 79k to 60k, resulting in a payload reduction of 24%. Turning it up to 9 will reduce the size even further to 57k (it is generally recommended to avoid using the max setting of 10 as this can place excessive demands on the CPU). Environments which employ extensive custom branding and customization, and those with a large proportion of remote users on slower connections, will benefit the most from adjusting IIS compression levels.

STEP 8 Take Advantage of Caching

SharePoint serves up a lot of data but not all of it is retrieved in real-time from the database. Much of the content requested by users can be cached in memory, including list items, documents, query results and web parts. How, when and by what rules SharePoint caches content is governed by cache profiles which can be configured for a single site or an entire site collection. Each cache profile is comprised of settings which tell the system which content to cache, who to cache it for, when to release it and when to check for updated content.

Site administrators can configure their own cache profiles to meet different user needs. Anonymous users, for example, can be assigned one set of cache policies while authenticated users are assigned another, allowing content editors to get a more recent view of content changes than general readers. Cache profiles can also be configured by page type, so publishing pages and layout pages behave differently, and administrators have the option to specify caching on the server, the client, or both.

In addition, the SharePoint Object Cache can significantly improve the execution time for resource-intensive components (such as the Content Query Web Part). By determining in advance whether or not to check for new query results on each request or to use a set of previous results stored in memory, queries which cross site boundaries can be prevented from making excess demands on the database. Large objects which are requested frequently, such as images and files, can also be cached on disk for each web application to improve page delivery times.

STEP 9 Manage Page Customizations

SharePoint Designer is a useful tool for administrators and power users but it has a side-effect which can be harmful to overall performance: page customizations (or, in legacy parlance, 'unghosting'). A customized page is one that has been opened in Designer, modified and saved. This has the effect of breaking the coupling between the pages on disk in the SharePoint Root folders and data in the content database. When customization occurs, the entire page content, including the markup and inline code, is stored in the database and must be retrieved each time the page is requested. In most cases, this introduces relatively little additional overhead on a page-by-page basis, but in larger environments with hundreds or even thousands of pages, all that back-and-forth to the database can add up to significant performance degradation.

To prevent this problem, administrators should implement a policy which restricts page customizations to only those situations where it is absolutely necessary. It may be fine, for example, for select contributors to customize user-authored layout pages in the /pages system library, while master and default layout pages are strictly off-limits. Also, it may be necessary to set up a 'sandbox' environment where users can create custom web parts which can then be imported into production without having to open controlled pages directly in SharePoint Designer. Site Collection and Farm administrators also have the option of completely disabling the use of Designer or, when necessary, using the "Reset to Site Definition" option to undo changes and revert back to the original content.

STEP 10 Limit Navigation Depth

One of the most significant design elements on any portal site is the global, drop-down, flyout menu at the top of each page. It seems like a handy way to navigate through all the various sites and pages—until it becomes so deep and cluttered that all ability to navigate beyond the first few levels is lost completely. Even worse, fetching all the data to populate the navigation menus can be resource-intensive on sites with deep hierarchies. SharePoint designers have the ability to customize the depth and level of each navigation menu by modifying the parameters for the various navigation controls within the master page. When using dynamic navigation, SharePoint builds a site hierarchy in the background and caches this information, helping to reduce the amount of data retrieved from the database, and stores it in an XML file. The size of this file, cached or otherwise, can impact the ability of pages to render in a timely manner—the controls must still be populated with all of the nodes in the hierarchy. Administrators should limit the depth of navigation to a manageable level which is both usable and constrained to a number of sites that does not impact performance. Furthermore, if the Pages library contains a large number of documents and the Pages option is selected in navigation settings, rendering the current navigation elements can also have a negative impact on performance; either exclude Pages from navigation or limit the amount of documents in this system library.

CONCLUSION SharePoint offers a tremendous array of options for improving performance and efficiency of collaborative portal applications. Understanding the inner workings of the framework and its dependencies upon key infrastructure components, including servers, network connections, services, operating system components and databases, is vital to the success of any optimization strategy. Implementing basic performance enhancements as outlined in this paper can provide tremendous improvements to the operation and performance of SharePoint within the enterprise.

HOW CAN IDERA HELP? Idera's **SharePoint performance manager** is a powerful and intuitive tool that enables developers and administrators to ensure the performance of their SharePoint applications. It increases user productivity and satisfaction, and also saves IT time and money by dramatically reducing the time to discover, analyze and fix SharePoint performance bottlenecks.

Try SharePoint performance manager for free!

Download a free 14-day trial: www.idera.com/Products/SharePoint/SharePoint-perf-manager/

ABOUT THE AUTHOR Eric Shupps is the founder and President of BinaryWave, a premier SharePoint consultancy headquartered in Irving, Texas, and Director of BinaryWave Limited, based in Bristol, England. Eric has worked with SharePoint Products and Technologies since 2001 as a consultant, administrator, architect, developer and trainer. He is an advisory committee member of the Dallas/Ft. Worth SharePoint Community group, participating member of the UK SharePoint User Group and Regional Evangelist for the International SharePoint Professionals Association. Eric has authored numerous articles on SharePoint, speaks at user group meeting and conferences around the world, and publishes a popular SharePoint blog at http://www.sharepointcowboy.com

---

For additional information or to download a 14-day evaluation of any Idera product, please visit: www.idera.com.

ABOUT IDERA

Idera provides tools for Microsoft SQL Server, SharePoint and PowerShell management and administration. Our products provide solutions for performance monitoring, backup and recovery, security and auditing and PowerShell scripting. Headquartered in Houston, Texas, Idera is a Microsoft Gold Partner and has over 5,000 customers worldwide. For more information, or to download a free 14-day full-functional evaluation copy of any of Idera's tools for SQL Server, SharePoint or PowerShell, please visit www.idera.com.

Idera | BBS Technologies | 802 Lovett Blvd Houston, Texas 77006
PHONE 713.523.4433 | TOLL FREE 877.GO.IDERA (464.3372)
FAX 713.862.5210 | WEB www.idera.com