Performance and Tuning Applied to the Five Major Resource Environments

Now let's look at how the performance and tuning process applies to each of the five major resource environments found within a typical infrastructure: servers, disk storage, databases, networks, and desktop computers. Since we are focusing first on people and processes ahead of technology, we will identify and examine issues associated with performance and tuning rather than talk about all of the technology products and tools used to actually do the tuning.

Server Environment

The first of the five infrastructure areas affected by performance and tuning covers all types and sizes of processor platforms, including mainframe computers, midrange computers, workstations, and servers. For simplicity, we refer to all of these platforms as servers. The following list details the major performance issues in a server environment:

1. Processors
2. Main memory
3. Cache memory
4. Number and size of buffers
5. Size of swap space
6. Number and type of channels

The number and power of **processors** influence the rate of work accomplished for processor-oriented transactions. Processors are the central components (also called the *central processing units*) of a digital computer that interpret instructions and process data. At its core, a processor adds and compares binary digits. All other mathematical and logical functions stem from these two activities.

For optimal performance, processor utilization rates should not exceed 80 percent. Tools are available to measure real-time utilizations of processors, **main memory**, and channels. **Cache memory** is available on most mainframe computers and on some models of servers, offering an additional means of tuning transaction processing. Cache memory differs from main memory in the following manner. Main memory is extremely fast circuitry (directly attached to the processor) that stores instructions, data, and most of the operating system software, all of which are likely to be used immediately by the processor. Cache memory is slightly slower memory (directly attached to main memory) that stores instructions and data about to be used. Cache is much faster (and more expensive) than secondary storage such as disks and tape.

Real Life Experience—Swapping Club Requires an Explanation

A CIO at a major defense contractor took an active interest in online response times for his customers and wanted to know exactly when and why performance might be degrading. It was a tough sell for systems analysts to explain to him why response times increased beyond service levels whenever only a few users were signed on yet seemed to improve with more users.

The cause of this performance paradox was the way main memory swap space was set up. When many online users were signed on, almost all the available memory would be used for online transactions and very little swapping out to disk storage occurred. When only a few users signed on, much of memory was used for batch jobs and the online users would frequently get swapped out, causing slower response times.

The **number and size of buffers** assigned for processing of I/O operations can trade off the amount of memory available for processor-oriented transactions. Buffers are high-speed registers of main memory that store data being staged for input or output.

The concept of virtual storage is used to temporarily store small, frequently used portions of large programs in part of main memory to reduce time-consuming I/O operations to secondary storage. The portion of main memory set aside for this is called *swap space* because the program segments get swapped in and out of main memory. The **size of swap space** can be adjusted to match the profiles of application and database processing.

The rate of processing I/O operations is also determined by the **number and speed of channels** connecting servers to external disk equipment. Channels are physical cables that connect the main memory to external I/O devices such as disk drives, tape drives, and printers.

Performance metrics commonly collected in a server environment include:

1. Processor utilization percentages
2. The frequency of swapping in and out of main memory

3. Percentage of hits to main memory cache
4. The length and duration of processing queues
5. The percentage of utilization of channels
6. The amount of processor overhead used to measure performance

Disk Storage Environment

The second type of infrastructure resources impacted by performance and tuning is disk-storage equipment. The following list indicates the major performance issues in a disk-storage environment:

1. Cache memory
2. Volume groups
3. Striping
4. Storage area networks
5. Network-attached storage
6. Extents
7. Fragmentation

Along with the configuration of the network and the design of databases, disk storage has a huge influence on the overall performance of an online system. If your disk environment is well-tuned, the resulting response times of your online applications are more likely to be acceptable. This is because it's a relative eternity in computer time to seek for a specific track on a data volume, to search for a specific piece of data on that track, to read or write it back out to the controller, to prepare it for transport along a channel, to transmit it down the channel, and then to finally have it arrive at its destination. Anything that can be done to shorten these steps and reduce this time significantly improves online response times.

One of the most effective ways to improve disk-storage performance is to utilize **cache memory** in RAID-type disk arrays. (RAID stands for redundant array of independent—originally, inexpensive—disks; see Chapter 12, "Storage Management," for a detailed description of the configurations and architectures of RAID disk arrays.) The reason cache is so effective at improving disk performance is that, for an overwhelming number of input or output transactions to and from disks, it eliminates the time-consuming steps of seeking and searching for data out on disk volumes. It accomplishes this through the use of ingenious pre-fetch algorithms that anticipate which data will be requested next and then preload it into the high-speed cache memory.

Pre-fetch algorithms—algorithms that analyze patterns of fetch activity to more intelligently anticipate requests—have become much more sophisticated in recent years. For example, a particular application may be accessing a database in such a way that every third record is being read. After a few of these requests are handled, the pre-fetch algorithm pre-loads every third record into the high-speed cache. The number and the complexity of pre-fetch algorithms vary from supplier to supplier, so you should thoroughly research what is available and what best suits your particular environment prior to investing large capital costs into an expensive asset.

Pre-Fetch Algorithm

A pre-fetch algorithm is a set of software instructions that analyzes the patterns of disk storage read-request activity to anticipate and pre-load, or pre-fetch, expected records of data.

The reason pre-fetch algorithms are so important is that they directly affect the likelihood of finding, or hitting, the desired record in cache. The percentage of these hits to the total number of requests—called *hit ratio*—is tracked very closely and is one of the best indicators of how well-tuned a cache disk array is to a particular application. Highly tuned cache memories can have hit ratios exceeding 90 percent, which is a remarkable percentage given the millions of requests coming into the array. Some high-performance disk systems always read the cache first to check for the desired record. Having the data pre-loaded in the cache drastically reduces the time necessary to retrieve and process the data because the relatively long amount of time needed to seek and search for data out on disk is eliminated. Some manufacturers reduce this time even further by reading the cache first on every operation.

Mapping out **volume groups** is another effective way to improve performance. Volume groups are a logical grouping of physical disk drives. The intent is to improve performance with reduced seek-and-search times by combining frequently used groups of physical volumes into one logical group. Suppose, for example, a large customer database spans across dozens of physical disk drives and you want to optimize performance for the most frequently used parts of the database. Analysis determines that customer spending habits and payment history are the two most frequently used tables in the database. Database administrators could re-configure the database so that these tables are spread across several physical drives instead of all of them being on a single drive. More of the desired data can be accessed quicker because of multiple paths to the multiple physical drives. This also facilitates pre-loading of anticipated data into cache.

Striping is a performance-improvement technique in which long blocks of data that are to be read or written sequentially are stored across multiple drives, usually within a logical volume group. This is done to increase the number of data paths and transfer rates and to allow for the simultaneous reading and writing to multiple disks.

Tuning the blocksizes of the striping to the blocksizes of the databases and the application can also improve performance. For example, a common blocksize for relational databases is 8 kilobytes; for a typical disk storage array, it is 32 kilobytes; for disk-volume utilities used for striping, it is 64 kilobytes. These three sizes accommodate each other efficiently because they are all even multiples of each other. But if some application were to use an uneven multiple of blocksize, the blocksizes would need some tuning to prevent spending costly and time-consuming overhead to reconcile the differences.

A **storage area network** (SAN) is a configuration enhancement that places a high-speed fiber-optic switch between servers and disk arrays (see Figure 8-1). The two primary advantages are speed and flexibility. The fiber channel can transmit data between the servers and the arrays at speeds of up to 1 or 2 terabytes per second (as opposed to 6 to 10 megabytes per second on standard channels). This greatly improves data-transfer rates and online response times. Parameters within the switch can help to improve performance by controlling buffers, contention, and load-balancing. The switch also allows a greater number of input paths to the switch than what might be available on the array. This ratio of server paths to array paths is known as the *fan-in ratio*. A properly tuned, two-to-one fan-in ratio can result in twice as much data being transmitted between the server and the arrays as would be possible with a one-to-one conventional ratio. The major downside to SAN is that it is expensive, primarily due to the fiber channels and its associated switch.

Storage Area Network (SAN)

A storage area network (SAN) consists of a high-speed subnetwork of shared storage devices. The configuration of a SAN enables all storage devices to connect to all servers within a local or wide area network (WAN).

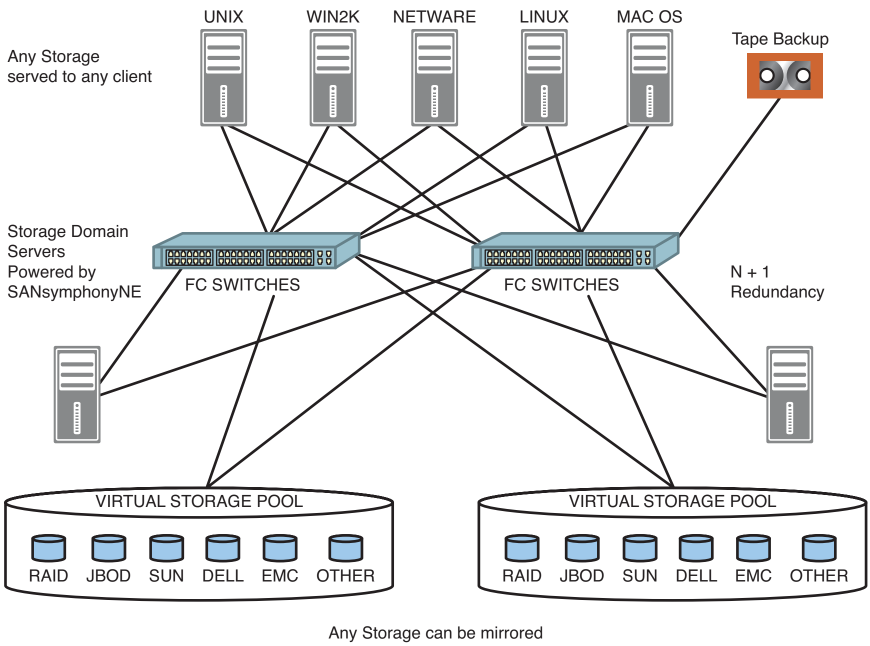

Figure 8-1 Diagram of a Storage Area Network (SAN) (Diagram courtesy of Nikita Kozlovskij)

Another storage configuration enhancement is **network-attached storage** (NAS), in which the disk array, along with the servers and the clients, is attached directly to the network. A disk-array server connects to the network with special-purpose interface devices or by using multipurpose operating systems running on multipurpose servers. The result is that data can travel from a NAS device over the network directly to the server or the client requesting the data. One common application of NAS is to use it for the network drives in a PC client network. The one drawback of using NAS rather than SAN in this way is that NAS tends to run slower. The speed of data transfer in a NAS environment is limited by the speed of the network. This normally runs between 10 and 100 megabitsper second. Gigabit Ethernet is now widely available with speeds between 10 to 100 gigabits per second.

Network Attached Storage (NAS)

A network attached storage (NAS) device is a server that is dedicated solely to file sharing. This enables other servers to be dedicated to processing activities while the NAS device is dedicated to storage activities.

The final two performance issues in a disk-storage environment involve the utilization of individual disk drives. **Extents** occur when the amount of data needing to be written exceeds the amount of original contiguous disk space allocated. Rather than abnormally terminating the request, the operating system looks for additional available space on the disk volume and extends the original contiguous space. Up to 16 extents are allowed per file, but the larger the number of extents, the longer it takes to seek and search for the data due to the fragmented nature of the data. **Fragmentation** can occur by other means as well, with the similar result of longer disk-access times and slower response times. Reallocating files and compressing them to reduce extents and fragmentation are good tuning methods to improve response.

Database Environment

The physical layout of a database can greatly influence the eventual response times of inquiries or updates. Databases in large companies can grow to hundreds of columns and millions of rows, and e-commerce databases can grow even larger. The placement of critical files such as system, data, log, index, and rollback files within the database can spell the difference between acceptable responses and outright lockups.

Eight of the most common performance issues associated with a database environment are shown in the following list:

1. Placement of table files
2. Initialization parameters
3. Placement of data files
4. Indexes and keys
5. Locks
6. Balance of system resources
7. Access patterns
8. Database fragmentation

First on the list is the **placement of table files**. A table file, which describes the various tables that reside within the database, is a critical navigational map of a database that serves as its data dictionary. Most database requests are first channeled through the table file. The placement of this file is extremely important to delivering acceptable online performance. The optimal location for this file is in the main memory of the server where fetch times are the quickest. However, the size of this file often prevents its being located there. In that case, as much of it as possible should be loaded into the cache memory of the disk array when space is available. When accessing data on disk drives, proper placement on the volume—as well as the volume's placement within a volume group—influences the amount of time it takes to complete database-access requests; this, in turn, affects online response times.

Initialization parameters for database management software have a direct impact on performance. Depending on the software being used, there can be dozens of these parameters from which to select and specify values. Some of the more common of these are database blocksize, the shared pool size for locating the data dictionary in memory, the system global area (SGA) for selected shared table files, log buffers, and the database block buffers for data files. The **placement of the data files** is crucial for minimizing access times and optimizing online response, just as it is for table files.

The judicious use of **indexes and keys** is another performance and tuning aid that can drastically reduce table lookup times. **Keys** are used to identify records in certain sequences by designating one or more unique fields in a table. The **indexes** are tree structures that hold the keys along with a pointer to the remainder of the data. Searching for a particular record is typically much quicker using an index, key, and pointer scheme because it eliminates the need to search the entire database.

Keys

Keys are a designated field within a record of a file or a database that enables the quick and secure searching or sorting of records within a table.

The keys offer another benefit as well. Since multiple keys are allowed for a given table, that table can, in effect, be sorted in multiple orders. This results in complex queries with multiple constraints that can be retrieved, in most cases, much more quickly with the use of keys. In cases where the resulting set is extremely large, it may actually be quicker to perform a single pass through the entire table. Keys can also be used to chain together groups of transactions that are normally executed sequentially. This improves transaction response times and user productivity by reducing keystrokes.

Indexes

Indexes are lists of keys within a database that enable the fast searching of a particular record or groups of records. Indexes also allow for multiple sorting of records within groups of tables or keys.

The use of indexes and keys must be balanced with the expense of additional storage, the overhead of processing and maintaining the indexes, and the updating and merging of keys. The greater the number of indexes, the greater the overhead. Performance specialists need to consider the trade-off between the benefit of fast retrievals and the cost of overhead for index maintenance.

Locks

Locks protect the integrity of a record of data by making it inaccessible to unauthorized users. Locks often are used with multi-user files and databases to ensure multiple updates are not made simultaneously.

Locks are sometimes used to protect the integrity of data records and to ensure single, sequential updates. Similar to keys, their implementation needs to be carefully planned since the overuse of locks can extend both disk space and access times. **Balancing system resources** of processors, memory, channel, and disk is necessary to ensure no single resource becomes saturated or grossly under-utilized in the quest for good performance. Monitoring the usage of these various resources to ensure none becomes under- or over-utilized is a proactive way to ensure balance among the devices. For example, once a server approaches 80 percent utilization, action should be taken to add capacity or, if possible, to re-distribute the workload to under-utilized servers.

Enterprise Resource Planning (ERP)

Enterprise resource planning refers to a set of applications that offers use and benefits across an entire corporation, rather than to just isolated departments. Finance, payroll, and personnel systems are examples of ERP systems.

Understanding the **access patterns** into a database is one of the best ways to optimize an application's performance. Anticipating what the read and write sequences are likely to be, and to which parts of the database they will likely occur, allows database administrators to map out the physical database in an optimal configuration. At the same time, performance specialists can configure disk-volume groups to align as closely as possible to the expected disk-access patterns. Measuring end-to-end response times is the best to verify the validity of these adjustments.

Extensive testing of a new database layout should be performed prior to production deployment to ensure cache hits are maximized, contention bottlenecks are minimized, and overall response times are optimized. Major hardware and software suppliers of enterprise resource planning (ERP) platforms and applications have comprehensive performance centers that can and should be utilized for testing prior to putting these systems into production.

Real Life Experience—A Chain With No Weak Links

An aerospace company instituted an extensive program to measure online response of transactions in a critical engineering application with multiple databases. The customers insisted that 90 percent of all transactions complete in less than a second. During the first few months this service-level objective was met, but even more improvements were sought.

A database consultant was brought in to suggest ways to optimize performance even more. He suggested that several of the commands frequently used sequentially could be chained together internally to save on keystrokes. The response times would slightly exceed the one-second objective since multiple commands would be processed. It took some considerable marketing efforts to convince some of the engineering customers that processing up to 10 commands in just under four seconds was more productive than entering each command manually, getting a response back in 0.9 seconds, then taking another second or two to enter the next command.

Over time, database records are rewritten in locations that are different from their optimal starting points. This causes two problems:

1. The data is no longer optimally placed for its anticipated access pattern.
2. The data records are now **fragmented**, causing less than efficient use of the disk space.

Periodically defragmenting the database with software utilities corrects both problems. Performance reports and trending statistics are available to help predict when it is best to schedule the running of defragmenting utilities.

Network Environment

There are a variety of factors that influence the performance of a network, beginning with its overall design and topology. In this section we discuss some of the more common issues:

1. Bandwidth
2. Line speed
3. Protocols
4. Single sign-ons
5. Number of retries
6. Nonstandard interfaces
7. Broadcast storms

Primary among these is network **bandwidth**. Calculating sufficient bandwidth is often more of an art than it is a science, but there are several planning aids available to help quantify these estimates. A robust capacity planning forecast is a good place to start. It should include a reasonable forecast of the number of concurrent users, the peak transaction traffic loads, and an estimate of transaction arrival patterns and types. This information should help determine the appropriate amount of bandwidth required for a given application. Network workload simulators can be used to test transaction response times under varying loads with different amounts of bandwidth.

Line speed is another key parameter affecting network performance. Obviously, the faster the speed, the greater the performance. Just as in a server and disk environment, resources need to be balanced and cost-justified, so simply upgrading lines from a T1 (1.544 megabits per second) to a T3 (43 megabits per second) may not be the best solution possible. It certainly may not be the most cost-effective solution. If lines must be upgraded for performance reasons, it pays to compare suppliers, shop around, and negotiate for terms such as maintenance, tariffs, and add-ons.

Network **protocols** are usually dictated by business, programming, and application requirements. In instances where alternatives are possible, give consideration to potential performance impacts, especially when running diagnostic programs such as traces and sniffers. Many shops struggle with the issue of **single sign-ons**—there is a trade-off between performance and security. The convenience and performance savings of logging on only once instead of multiple times for access to the network, operating system, database management system, and a particular application must be weighed against the potential security exposures of bypassing normal password checks. Most applications have robust security features designed into them to mitigate this risk.

Transmission errors occur periodically on all network lines. In most instances, the errors are quickly detected, the network request is retried, and—since most errors are temporary in nature—the retransmission is successful. If the retransmission is not successful, multiple retries are attempted. After a certain number of unsuccessful retries, the error is designated as permanent and more sophisticated diagnostics are activated. The number of retries to be attempted impacts performance and should be monitored and adjusted appropriately. Suppliers should be involved when retries originate from the line side of the network.

Connecting devices with **nonstandard interfaces** to the network can cause major performance problems such as locking up lines, introducing interference, or, in extreme cases, flooding the network with nonstop transmissions called **broadcast storms**. This becomes another trade-off between performance and security. In many shops, executives may approve the temporary use of a nonstandard device for testing, evaluation, or special business needs. Suppliers can sometimes offer interfaces to increase the compatibility of these devices with a network. Caution should be exercised and a heightened awareness of network performance put in place whenever an exception to network standards is permitted.

Desktop Computer Environment

The final area of performance that we will cover involves the desktop computing environment. While there are far fewer issues to deal with in this environment, they can still have serious impact on end-users. The following lists the six issues we will discuss:

1. Processors
2. Memory
3. Disk storage space
4. Network connections
5. Diagnostic tools
6. Administrative tools

The size and power of the **processing chip** (see Figure 8-2) required by a desktop computer is influenced by the number and type of applications that will be run. Spreadsheets, statistical analysis programs, and highly graphical routines need much more processing power to deliver acceptable response than word-processing applications. The amount of processing power and **memory** needed to ensure acceptable performance depends on the number and types of windows expected to be open simultaneously. Two windows consisting of email and textual documents need far fewer resources than six windows that include compound documents and multimedia streaming videos.

Another factor impacting desktop performance is the number of applications that run in the background that are automatically loaded during start-up. These background applications include:

- Automatic updaters for applications
- Automatic updaters for operating systems
- Anti-virus software
- Restriction software to protect children
- Anti-spam software
- Personal firewalls
- Anti-spyware

These various applications can consume half of the processing capability of the desktop without the user even being aware of it. New and advanced features of modern desktop operating systems may also use large amounts of capacity. Microsoft's Vista operating system is functionally impressive in terms of security and ease of use. But its advanced, 3-D graphics (refined from its video game experiences), while at times dazzling in their displays, requires a lot of memory and processing power to operate satisfactorily.

The amount of **disk storage space** has a less direct effect on the performance of desktop computers than either the processor or memory. In this regard, it differs from the role external disk storage plays in servers and mainframes. External desktop disks do impact performance in the sense that retrieving data locally rather than over the network usually shortens response times.

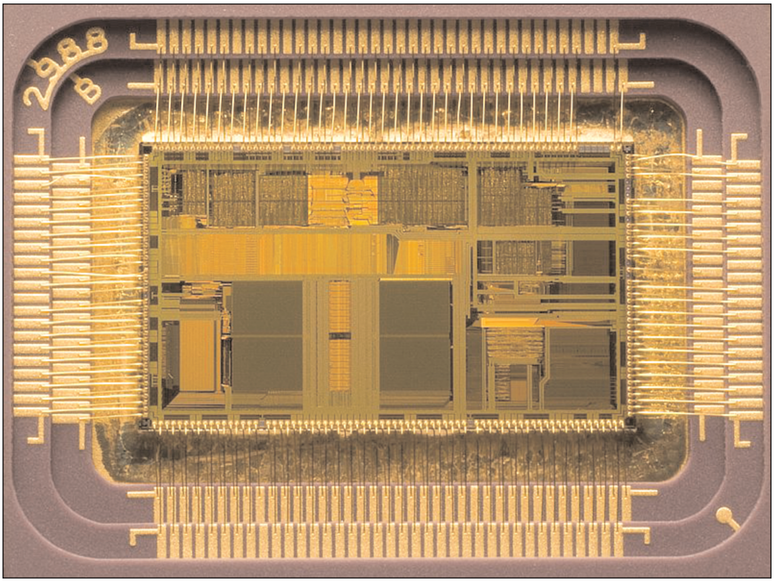

Figure 8-2 Intel Microprocessor (12x6.75mm)

Desktop devices are attached to LANs and WANs through **network connections.** The speed of the lines and of the modems that utilize these network connections obviously impact performance at the desktop; line speed and modems should be tuned to meet requirements at reasonable costs. The number of desktops on the networks, and the corresponding number of hubs and routers that interconnect the various network segments and devices, also influences desktop performance.

Diagnostic tools that enable a remote technician to take control of a desktop computer to troubleshoot problems can adversely impact performance if parameters are not initialized properly. Some diagnostic tools also are capable of running in real-time on the desktop. If these tools are inadvertently left running, they can easily impact desktop performance adversely.

Performance can also be impacted by **administrative tools**, such as those that automatically inventory the configuration of the desktop through an asset-management system. Sophisticated tools of this type can interpret and process large amounts of detailed information about the internal specifications of the desktop, which can slow down normal operations.