Supplementing Windows 95 and Windows 98 Performance Data for Remote Measurement and Capacity Planning

BonAmi Software Corporation

Abstract

Microsoft Windows NT provides a full featured Performance Monitor program that is widely used in industry for performance measurement and capacity planning. While the Windows NT operating system itself supports a large and varied collection of performance objects, the Windows 95 and Windows 98 operating systems support a much smaller and far less useful set of performance objects.

This paper demonstrates how to supplement the performance objects collected from Windows 95 and Windows 98 and how to make these performance objects available for remote performance monitoring.

Overview – Monitoring Windows 95 and Windows 98 Machines

This paper is about remotely monitoring the performance of computers running the Microsoft Windows 95 and Windows 98 operating systems.

In 1992, Windows v3.1 ran on Intel 286 and 386 machines with as little as one (1) megabyte (MB) of random access memory (RAM). When Microsoft developed Windows 95, RAM was still expensive relative to the cost of a computer – 1MB RAM chips commonly sold for $30-40 – and new PCs came equipped with 8MB RAM. To make Windows 95 an affordable upgrade for Windows and DOS PC users, Microsoft tried very hard make the operating system run on machines with as little as 4MB of RAM.

There were many tradeoffs to make in order to create an operating system that might actually work in 4MB of RAM. For example, only a subset of the Win32 programming interface (API) was supported. Second, most of the Windows NT security options were not implemented. Third, and of direct relevance to this paper, Windows 95 did not include Windows NT's numerous performance counters nor the NT Performance Monitor application.

Nevertheless, when Windows 95 shipped in late 1995, it did include a smaller set of built-in performance counters along with a new performance monitor program called System Monitor. The full list of the built-in Windows 95 and Windows 98 performance counters is given in Appendix A. It is particularly important to note that all of the performance counters represent total values; none are per-process. Figure 1 below shows a picture of Windows 95 System Monitor displaying some selected performance data
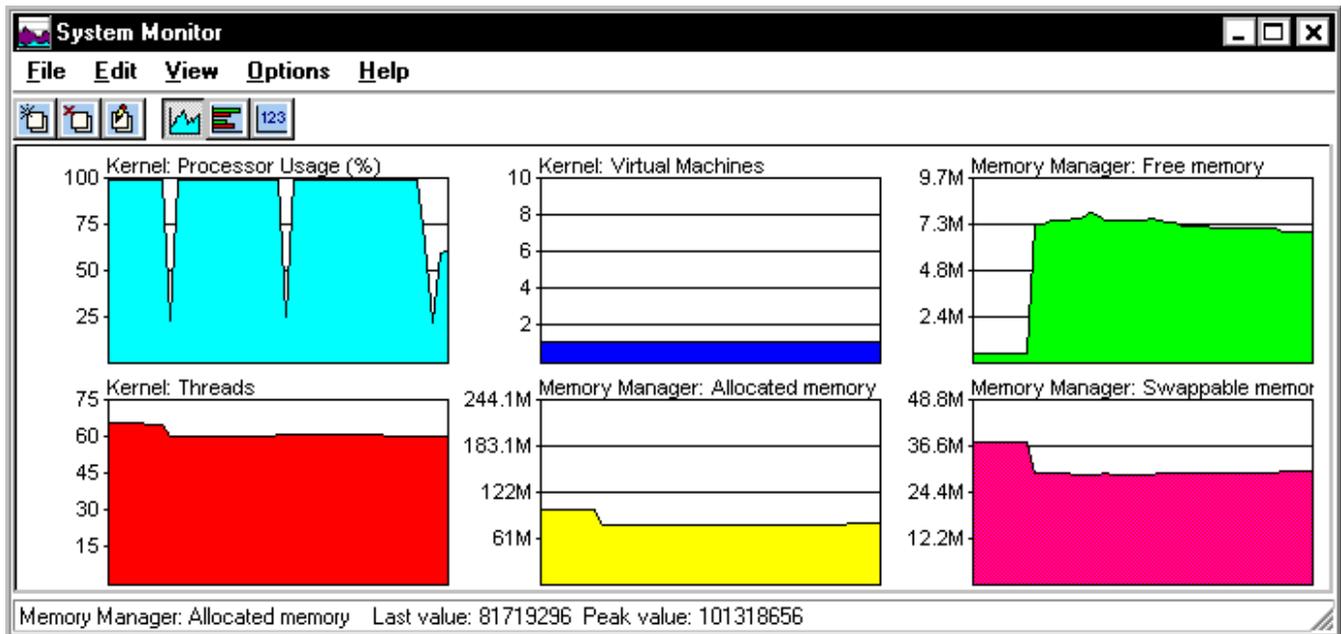

Figure 1 - Sample Windows 95 System Monitor Display

With this in mind, closer examination of the performance counters in Figure 1 shows that something is quite wrong. The most fundamental performance monitoring metrics, per-process CPU usage and per-process RAM usage, are missing! As it turns out, neither Windows 95 nor Windows 98 provides the base operating system support needed to obtain per-process data. For our purposes, this is a critical and, until now, a fatal omission.

In order to reach our objective of monitoring Windows 95 and Windows 98 performance from a remote machine (say a Windows NT machine running NT Performance Monitor), we first need to acquire the raw performance data – and the important CPU and RAM data that we want is unavailable from the operating system. To acquire this data, we will use a third-party product called Performance '95 from BonAmi Software Corporation.

The BonAmi Performance Engine in Performance '95 provides several of the missing performance counters, especially the important per-process CPU and RAM usage data.

For the purpose of this paper, we have combined BonAmi's Performance Engine with a new remote agent. The remote agent's role is to send Windows

95 and Windows 98 performance data to the Windows NT Performance Monitor application, over a TCP/IP network. The software and implementation details are described later in this paper.

Introduction – Performance Monitoring and Capacity Planning

Performance monitoring and capacity planning software helps companies to analyze their computing usage and requirements. Enterprise-wide, computers and software are expensive, and businesses struggle to appropriately plan for their acquisition, usage and retirement.

There are many commercial software packages that support analysis and planning in the enterprise. These packages support the gamut of major system and server platforms like IBM MVS, Digital VMS, Unix, Microsoft Windows NT and IBM OS/2. Servers are expensive machines and the software that runs on servers is also expensive. Considering cost alone, it is important to monitor and plan for these resources.

Nevertheless, the *vast number* of business computers today are running Microsoft Windows 3.x, Windows 95 and Windows 98. As explained previously, these operating system platforms do not adequately provide raw performance data. Thus, these platforms are not well supported (if they are supported at all) by Performance Analysis and Capacity Planning Solutions vendors.

Windows NT and NT Performance Monitor

Microsoft Windows NT 4.0 is a 32-bit, multitasking, multithreaded operating system. Figure 2 shows the position of Windows NT 4.0 in the evolution of Microsoft's personal computer operating system family.

Like all complex software, Windows NT inherited much of its architecture from its predecessors. As the focus of this paper is on performance monitoring, we will pause only to note that Windows NT appears to have inherited much of its embedded performance reporting hooks from work done on OS/2. The performance counters can be displayed using the free Windows NT Performance Monitor as shown in Figure 3.

The Windows NT Performance Monitor has several notable features:

1. It is free. Microsoft has bundled Performance Monitor with Windows NT ever since the earliest versions of NT. While OS/2 was the first PC operating system to include hooks for
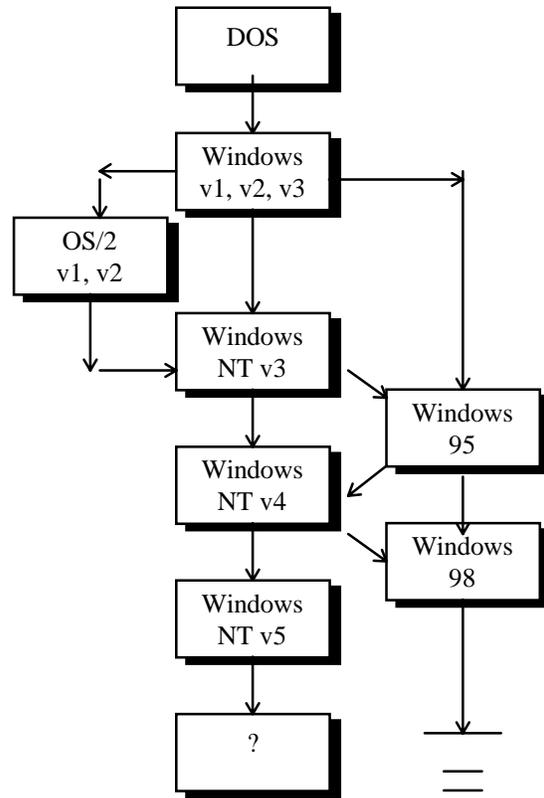

Figure 2 - Important Milestones in Microsoft Operating Systems Development

performance monitoring, there was no high-quality performance monitor included with the operating system. Companies like IBM and BonAmi did sell OS/2 add-on performance monitors, however.

2. Source code and sample programs that work with Performance Monitor are freely available as part of the Microsoft Win32 Software Development Kit (Win32 SDK).

3. It is extendable. Performance Monitor can be extended to include performance counters from both applications and system programs. Although the programming interface is cumbersome, many custom programs do take advantage of these extensions.

4. It can monitor both local and remote machines. Performance Monitor can monitor remote NT machines just as easily as it monitors the local machine.
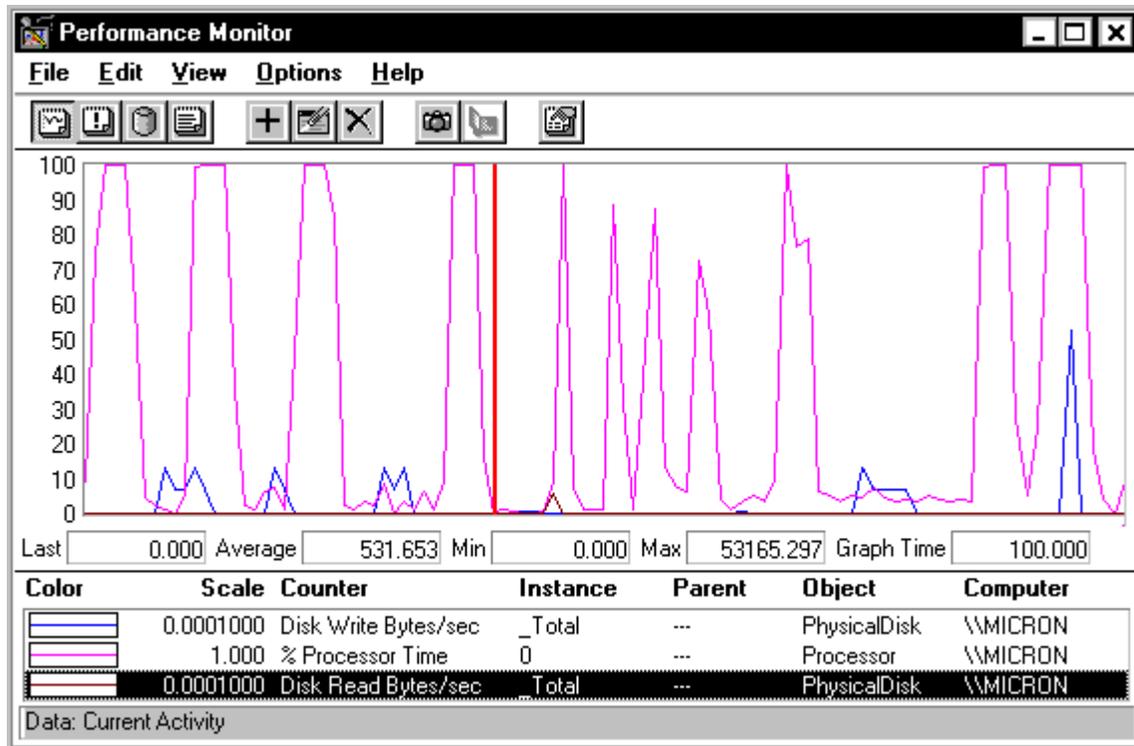
Figure 3 - Sample Windows NT Performance Monitor Display

5. It can monitor "foreign" computers. Foreign computers are located on a common network with the Performance Monitor computer, but they are not running Windows NT. For the purposes of this paper, Windows 95 and Windows 98 are examples of foreign computers. Significant programming steps are required in order for Performance Monitor to communicate with foreign computers. This will be discussed later.

In summary, Performance Monitor can be used to monitor remote non-Windows NT machines over a network. We will exploit this capability to remotely monitor Windows 95 and Windows 98 machines from Windows NT.

Windows NT Performance Counters

Fundamental to Performance Monitor is the concept of a counter. On hardware devices, counters count usage or accesses to the device. In Windows NT, counters only increment in value and they are never cleared. Performance Monitor uses counters as follows:

1. Take an instantaneous reading (snapshot) of a counter

2. Wait a measured period of time (interval)

3. Take a second snapshot of the counter

4. Subtract the first snapshot from the second for a raw performance count

5. Subtract the first snapshot from the second and divide by the time interval for a performance rate

As an example, the Windows NT operating system includes counters to measure activity on a physical disk drive. The preceding Figure 2 showed Performance Monitor displaying several counters including Disk Reads in bytes per second and Disk Writes in bytes per second

Performance Counters are not limited to hardware devices alone. *Instrumentation* is a process whereby performance counters are inserted in software to help analyze performance. Software performance counters function like hardware counters and can also be displayed by Performance Monitor. Windows NT allows both system and application programs to be instrumented for display by Performance Monitor.

Windows NT Registry and Performance Counters

The Windows Registry is the central repository for hardware and software configuration information in Microsoft's 32-bit operating systems. Software programs use the registry for a myriad of purposes including storage and retrieval of configuration information, locating program objects, and storing license information. The registry supports both static and dynamic data and is used both by the operating system and by application programs.

Windows NT performance counters are accessed through the registry as shown below:

The registry key:

```
HKEY_LOCAL_MACHINE
    \SYSTEM
        \CurrentControlSet
            \Services
```

is the root for most Windows NT performance counters as well as additional application-specific counters.

Since the registry is a general purpose database, it would be unwise to continuously update its entries with all types of performance data. For this and other related reasons, performance data is not actually stored in the registry. Instead, the action of accessing performance registry keys causes other programs to actually retrieve the data and return it to the caller.

Applications supply performance data by carrying out the following steps:

1. First, an application adds its custom keys under the registry key:

    ```
    HKEY_LOCAL_MACHINE
        \SYSTEM
            \CurrentControlSet
                \Services
    ```

2. Second, a "Performance Dynamic Link Library (DLL)" must be created to serve as the link between the application program and Performance Monitor.

The Performance DLL is invoked whenever a program requests to see an application's performance objects. The Performance DLL funnels requests from Performance Monitor to the application and it funnels performance date from the application back to Performance Monitor.

Not only does this technique support retrieving data from custom application programs, but it can also be used to retrieve performance data from non-Windows

NT machines. In both cases, it is the Performance DLL that is the intermediary between Performance Monitor and the application or foreign computer. We will use this same technique to retrieve performance data from Windows 95 and Windows 98 machines for display by Performance Monitor.

Windows 95 and Windows 98 Architecture Overview

Microsoft Windows 95 and Windows 98 (Windows 9x) are mixed 32-bit and 16-bit, multitasking, multithreaded operating systems. Figure 1 earlier showed the position of Windows 9x in the evolution of Microsoft's personal computer operating system family.

The Windows 9x and Windows NT operating systems share a common but not identical architecture for embedding performance counters into the operating system. To read and display these counters, Windows 9x includes a performance monitor program called System Monitor

The Windows 9x System Monitor has several notable features:

1. It is free. Like the Windows NT Performance Monitor, Microsoft has bundled System Monitor with both Windows 95 and Windows 98.

2. Unlike the Windows NT Performance Monitor application, source code for System Monitor is *not* provided.

3. It is extendable. System Monitor can be extended to include performance counters from both applications and system programs. Although the programming interface is cumbersome, many programs do take advantage of these extensions. Adding counters to Windows 9x is similar but not identical to adding counters to Windows NT.

4. When properly configured, it can monitor both local and remote machines. System Monitor can monitor remote Windows 9x machines just as easily as it monitors the local machine.

5. It *could possibly* monitor "foreign" computers. Foreign computers are located on a common network, but they are running an operating system other than Windows 9x. The author suggests that techniques similar to those used to get Windows NT Performance Monitor to monitor foreign machines could also be applied here. An experiment like this is beyond the scope of this paper, however, and is left as an exercise to the reader. ☺

Windows 95 and Windows 98 Registry and Performance Counters

Like Windows NT, the Windows 95 and Windows 98 (Windows 9x) registries are the central repository for configuration information. Also like Windows NT, the Windows 9x performance counters are accessed through the registry as shown below:

The name, description and type of the currently active performance data objects in the system are stored under the following Windows 9x registry key:

```
HKEY_LOCAL_MACHINE
    \System
        \CurrentControlSet
            \control
                \PerfStats
                    \Enum
```

Performance data is collected using the registry keys starting at:

```
HKEY_DYN_DATA
    \PerfStats
        \StartStat
```

For example, in order to retrieve the current number of threads in the system, performance data would be collected from:

```
HKEY_DYN_DATA
    \PerfStats
        \StartStat
            \KERNEL
                \Threads
```

Performance Monitoring Windows 95 and Windows 98 Machines

As described previously, System Monitor is used to view the current value of the performance objects referenced in the Windows 9x registry. System Monitor can also be used from one Windows 9x machine to view the performance of other (remote) Windows 9x machines.

Note that while these steps allow Windows 9x machines to remotely monitor other Windows 9x machines, a Windows 9x machine running System Monitor cannot directly monitor a Windows NT machine. Nor can a Windows NT machine running Performance Monitor directly monitor a Windows 9x machine.

Data Collection Using the Performance '95 Performance Engine

Unlike Windows NT, neither Windows 95 nor Windows 98 (Windows 9x) provide the built-in counters for collecting per-process CPU and RAM usage performance data. As these are the cornerstone metrics for a useful performance evaluation, we need to acquire these elsewhere.

BonAmi Software Corporation's Performance '95 product is a Windows 9x Performance Monitor and tuning software product. Performance '95 is functionally composed of two parts: a graphical user interface (GUI) and a low-level Performance Engine. The Performance Engine itself is composed of a dynamic virtual device driver (VxD) and a dynamic link library (DLL) as shown below:
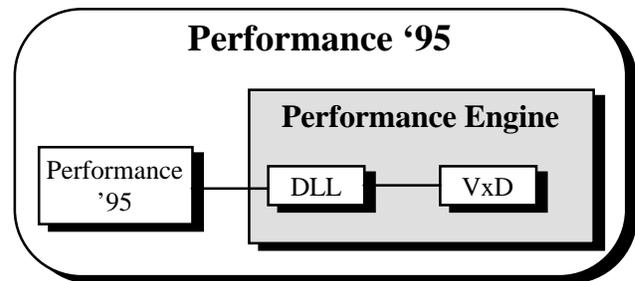

Figure 3 - Performance '95 Architecture

This paper used the BonAmi Performance Engine without modification.

A Remote Agent program was written to communicate between the Performance Engine and the Windows NT Performance Monitor. The relationship between Performance Monitor and the Performance DLL on Windows NT, and the Remote Agent and the Performance Engine on Windows 9x is illustrated in Figure 4.
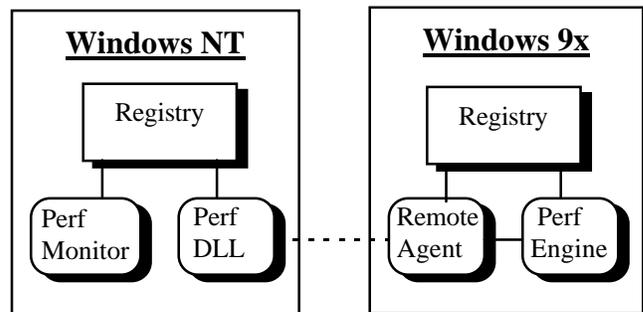

Figure 4 - Windows NT Performance Monitor Communicating with Windows 9x Remote Agent

The Remote Agent program collects data from the Performance Engine and communicates with Performance Monitor via a predefined TCP/IP socket. Other than collecting data from the Performance Engine and sending it to Performance Monitor, the Remote Agent is otherwise invisible. Its own CPU and RAM usage are measurable on either the Windows 9x platform (by using Performance '95) or as part of the data transmitted to Performance Monitor.

Note that the Remote Agent can return all of the performance data items collected by the Performance Engine along with items from the Windows 9x registry. Appendix A lists all the Windows 9x registry items that are collected by the Remote Agent.

Remotely Monitoring Windows 9x Machines from Windows NT

Now that we have a remote data collection engine, we can supply Windows 9x performance data to Performance Monitor. The final product looks like the earlier Figure 4.

We have inserted an extended performance object in the Windows NT registry that references our Performance DLL. When it is called, the Performance DLL communicates directly with the remote agent to collect Windows 9x performance data. Windows NT Performance Monitor then displays this data, just as if it were acquired locally.

Inserting the extended performance object requires adding a registry entry:

```
HKEY_LOCAL_MACHINE
    \SYSTEM
        \CurrentControlSet
            \Services
                \BAPrfSvc
```

(BAPrfSvc is our extended object)

The sub-keys under the BAPrfSvc key above specify the location of the Performance DLL and also specify three entry points into the DLL: Open, Close, Collect. The system calls these entry points when Performance Monitor starts (Open), when it stops (Close), and when it wishes to collect data (Collect).

Concluding Remarks

This paper has described techniques to:

1. Supplement the performance data available in Windows 95 and Windows 98 machines.

2. Retrieve Windows 95 and Windows 98 performance data for display in Windows NT Performance Monitor.

The supplemented Windows 9x performance data should prove to be a valuable addition to data collected from Windows NT and other platforms. This data will be amenable to industry normal standards of performance analysis and capacity planning.

As a system's performance is monitored, it is unavoidable that the process of observation itself distorts system performance. While the author is not aware of any published data for data collection and processing overhead on Windows PC platforms, the following general notes may be of some help.

1. The overhead of collecting disk performance data on Windows NT machines is approximately 1.5% on an Intel 386 20Mhz computer, and less on faster computers (Windows NT 3.51 Resource Kit, Volume 4: *How to Optimize Windows NT*).

2. Performance monitoring overhead in general can be reduced by collecting less data, collecting data less frequently, and by avoiding any data that is *costly* to collect. Windows NT identifies costly data via a "Costly" attribute and such data is not collected by Performance Monitor.

3. Another way that performance monitoring overhead can be reduced is to offload the process of analyzing collected data. Windows NT Performance Monitor and Windows 98 System Monitor both allow data to be logged for post analysis. While the Windows 95 System Monitor does not provide this feature, BonAmi's Performance '95 supports logging for both Windows 95 and Windows 98.

4. The performance overhead due to the remote agent described in this paper also varies with the type and quantity of data collected. Collecting RAM data is a most expensive operation; this is due to the architecture of the Performance Engine's data collection algorithms.

The techniques discussed here now present us with the opportunity to efficiently and accurately monitor the vast expanse of Windows 95 and Windows 98 machines. With the goal of remote measurement and capacity planning, a not disingenuous argument can be made as to choosing when, where and how frequently these machines should be so monitored. Answering those questions is beyond the experience of this author, however, and he gladly defers to other more experienced practitioners. ☺

Finally, while the Performance Engine does significantly improve the availability, quantity and

value of the data collected from Windows 9x machines, other important performance objects are still lacking.   Among these are the lack of working-set counters and disk utilization counters.

Appendix A - Windows 95 and Windows 98 Performance Counters

Windows 95 and Windows 98 provides performance data for the following objects:

- Dial-Up Adapter
- Disk Cache (Windows 98 only)
- File System
- Kernel
- Memory Manager
- Microsoft Network Client
- Microsoft Network Server file and printer sharing services

The BonAmi Performance Engine adds performance data for the following objects:

- Process objects
- Process memory objects
- Thread objects

The Windows 95 and Windows 98 performance counter descriptions are summarized   from the explanatory help text messages from the System Monitor program.   Additional information is available in the Windows 98 Resource Kit, Chapter 26, Performance Tuning.

File System

Setting	Description
Bytes read/sec	Bytes read from the file system per second.
Bytes written/sec	Bytes written to the file system per second.
Dirty data	The number of bytes waiting to be written to the disk. Note that dirty data is kept on a per cache block basis, and so a number higher than the actual number of bytes written may be reported.
Reads/sec	The number of read requests made to the file system per second.
Writes/sec	The number of write requests made to the file system per second.

Process Objects

Setting	Description
Current CPU	Current CPU usage for all threads in this process.
Cumulative CPU	Cumulative CPU usage for all threads in this process.
Running Time	Elapsed time since this process started.
Base Priority	Base execution priority for this process
Threads	Number of threads belonging to this process

Disk Cache (Windows 98 only)

Setting	Description
Cache buffers	Number of active buffers in a cache. This includes any and all compressed buffers as well.
Cache hits	Number of times data was found in the cache, resulting in I/O requests.
Cache misses	Number of times data was not found in the cache, resulting in I/O requests.
Cache pages	Current number of disk cache pages.
Failed cache recycles	Number of times a recycling request (either LRU or random) has failed. This can happen in low memory situations, or when all cache buffers are currently in use.
LRU cache recycles	Number of times the cache is sequentially searched for a buffer to recycle, beginning with the oldest data. This happens when new data needs to be added to the cache, or when memory manager needs to borrow memory from the cache.
Max cache pages	Maximum number of disk cache pages.
Min cache pages	Minimum number of disk cache pages.
Random cache recycles	Number of times the cache is randomly searched for a buffer to recycle. This can happen whenever the cache becomes filled with data that has not been reused recently.

Dial-Up Adapter

Setting	Description
Alignment Errors	Serial port alignment errors.
Buffer Overruns	Serial port buffer overrun errors.
Bytes Rcvd/Sec	Number of bytes received per second
Bytes Xmit/Sec	Number of bytes transmitted per second.
Connection Speed[1]	Connection speed in bits per second
CRC Errors	Number of frames with CRC errors.
Frames Rcvd/Sec	Number of good frames received per second.
Frames Xmt/Sec	Number of frames transmitted per second.
Framing Errors	Serial port framing errors.
Incomplete Frames	Number of incomplete frames received.
Overrun Errors	Serial port overrun errors.
Timeout Errors	Serial port timeout errors.
Total Bytes Rcvd[1]	Total number of bytes received.
Total Bytes Xmit[1]	Total number of bytes transmitted.

[1] Windows 98 only

Kernel

Setting	Description
Processor Usage (%)	The percent of the processor time that is not spent idle. This value is approximate.
Threads	The number of processor threads present in the system.
Virtual Machines	The number of virtual machines (VMS) present in the system.

Process Memory Objects

Setting	Description
Private Code	Private read-only (code) memory currently in-use by this process.
Private Data	Private read-write (data) memory currently in-use by this process.
Shared Code	Shared read-only (code) memory currently in-use by this process.
Shared Data	Shared read-write (data) memory currently in-use by this process.

Thread Objects

Setting	Description
Current CPU	Current CPU usage for this thread.
Cumulative CPU	Cumulative CPU usage for this thread.
Running Time	Elapsed time since this thread started.
Base Priority	Base execution priority for this process
Current Priority	Current execution priority for this thread.

Memory Manager

Setting	Description
Allocated memory[1]	Number of bytes of memory committed in the system. This is the total amount of memory that has been allocated in the system, across all components and applications.
Discards	Number of page discards per second.
Disk cache size	Current size of disk cache.
Free Memory[2] **Unused phys memory**[3]	Amount of physical memory (RAM) not currently in use.
Instance faults	Number of instance faults taken per second.
Locked memory[1]	Amount of memory allocated and locked.
Locked non-cache pages[3]	Number of non-cache locked pages.
Maximum disk cache size	Maximum disk cache size.
Mid disk cache size[3]	Mid disk cache size.
Minimum disk cache size	Minimum disk cache size.
Other memory[1]	Number of bytes of memory allocated which are not stored in the swap file. Examples of 'other' memory are disk cache pages, memory allocated fixed (non-pageable), and memory-mapped files.
Page faults	Number of page faults taken per second.

Page-ins	Number of page-in operations per second.
Page-outs	Number of page-out operations per second.
Pages mapped from cache[3]	Number of pages mapped directly out of the cache file.
Swap file defective	Number of bytes in the swap file found to be physically defective on the swap medium. Swap file space is allocated in 4K frames; a single damaged sector causes the entire frame to be marked defective.
Swap file in use	Number of bytes in the swap file currently in use.
Swap file size	Size of swap file in bytes.
Swappable memory[1]	Number of bytes allocated from the swap file. Note that the swap file pages which are locked still count as 'swappable' for the purpose of this metric.

[1] This number includes the disk cache size.
[2] Windows 95 only.   [3] Windows 98 only

Microsoft Network Client

Setting	Description
Bytes read/sec	Bytes read from the redirector per second.
Bytes written/sec	Bytes written to the redirector per second.
Number of nets	Number of nets running.
Open files	Number of open files on net.
Resources	Number of resources.
Sessions	Number of sessions.
Transactions/sec	Number of SMB transactions per second.

Microsoft Network Server

Setting	Description
Buffers	Server working buffers.
Bytes Read/sec	Disk reads in bytes per second.
Bytes Written/sec	Disk writes in bytes per second.
Bytes/sec	Sum of disk reads and disk writes in bytes per second.
Memory	Memory used by the server.
NBs	Server network buffers.
Server Threads	Windows threads used by the server.

Bibliography

Helen Custer, *Inside Windows NT*, Redmond WA, Microsoft Press, 1993.

H.M. Deitel and M.S. Kogan, *The Design of OS/2,* Reading MA: Addison-Wesley, 1992.

Microsoft Windows 95, Windows 98 and Windows NT Resource Kits, Microsoft Corporation.

Private correspondences from Microsoft and IBM regarding Windows 95 and OS/2 performance monitor architectures.

Jeffrey Richter, *Advanced Windows: The Developer's Guide to the Win32 API for Windows NT and Windows 95*, Redmond WA: Microsoft Press, 1995.

Frederick Scholl, *Performance Strategies*, Network Magazine, Vol 12, No. 12, Nov 1997.

Andrew Schulman, *Unauthorized Windows 95*, San Mateo CA, IDG Books, 1994.