WHITE PAPER

ANALYZING IBM i PERFORMANCE METRICS

BY CHUCK LOSINSKI, DIRECTOR OF AUTOMATION TECHNOLOGY

IN EVERY INDUSTRY, BUSINESS-CRITICAL PROCESSES DEPEND ON APPLICATION PERFORMANCE. INTERRUPTION OF REAL-TIME PROCESSING, CUSTOMER SERVICE DELAYS, AND MISSED SLAs AND DEADLINES ARE ALL PLAUSIBLE OUTCOMES WITH ALL-TOO-REAL CONSEQUENCES. FORTUNATELY, IBM i AND THIRD-PARTY VENDORS OFFER SIGNIFICANT POWER TO PREVENT SUCH DISASTER.

help/systems

ANALYZING IBM i PERFORMANCE METRICS

BY CHUCK LOSINSKI, DIRECTOR OF AUTOMATION TECHNOLOGY

The IBM i operating system is very good at supplying system administrators with built-in tools for security, database management, auditing, and journaling. But the operating system has another gem that often goes overlooked: its ability to automatically, continuously, and efficiently collect performance data.

This white paper will examine critical system metrics for performance that administrators need to monitor and review for real-time performance issues and future bottlenecks. It will also look at the commands associated with performance collection services.

In the absence of a third-party solution, many elements of performance monitoring not achieved by IBM i Navigator's Performance Tool Library—which include real-time alerting and system-specific intervals—can lead to sluggish performance, interruption, and outages. This paper also outlines the most critical features required of a third-party tool and how to incorporate them into your performance monitoring.

Understanding IBM i Collection Services

Collection Services is an IBM tool set for gathering IBM i performance data in intervals and storing it inside a collection object (object type *MGTCOL). These objects store the raw performance data that reflects an interval of time and the performance of the system, jobs, memory pools, and disk. This data can be used to diagnose performance issues, plan for growth, or monitor performance so you can be proactive with system-related tuning or application performance issues.

Configuring Collection Services

- Use the command CHKPFRCOL to determine if Collection Services is turned on.
- Use the prompted CFGPFRCOL command to see at which level of detail performance collections are currently configured.

We recommend that you set this protocol profile setting to *STANDARDP. Additionally, IBM recommends running performance collections continually as part of normal operations. The command STRPFRCOL starts the monitoring process. You can also use System i Navigator to start and stop the process.

Analyzing Collection Services

The command CRTPFRDTA processes the data in the *MGTCOL object and outputs it to performance database files. Once in database files, this data typically requires use of a third-party or IBM service, such as PM for Power Systems (formerly PM/400), to analyze the data. You can see how PM/400 is configured by typing "GO PM400" in a command line.

IBM i Navigator's Performance Tools Library creates detailed and summary reports, which can be viewed online for a fee. While the tool helps predict growth and identify offending jobs, it does not help with real-time performance issues. For enhanced performance collection and reporting, consider a third-party tool that alerts in real time.

Collecting Data: System Status & Disk Status

Most standard data from Collection Services is available on the screens for Work with System Status (WRKSYSSTS) and Work with Disk Status (WRKDSKSTS). An explanation of these metrics and their guidelines follows.

Work with System Status metrics (WRKSYSSTS):

CPU % Used – Analyze your CPU use over time and set a critical threshold level 5 to 10% above your peak "normal" level. You might also need to consider different thresholds based on time of day. A high CPU utilization may not be a problem. For instance, a high CPU with a low interactive response rate is good—there is good utilization of the processor without impact on the end user. Generally, IBM recommends an average CPU utilization of:

- 50% for a single core
- 70% for two cores
- 85% for eight cores
- 90% for 32 cores

Jobs in System – *This includes active jobs, queued jobs, and completed jobs that still have spooled files attached.* Too many jobs in the system can cause delays during IPL, when signing on to the system; it can also cause delays with certain job-related commands.

System values that affect Jobs in System are QTOTJOB (initial total jobs), QACTJOB (initial number of active jobs), QADLTOTJ (additional total jobs), QADLACTJ (additional active jobs), and QMAXJOB (maximum number of jobs).

During IPL, the operating system allocates storage for each job. If it needs to allocate additional memory for new jobs because the QACTJOB or QTOTJOB has been exceeded, there will be a short delay. If you exceed the QMAXJOB value, new work will not be able to start up on your system, so make sure this value is high enough to handle the amount of jobs historically seen on your system. Keep your joblogs and spooled files cleaned up to prevent this number from getting too high.

Permanent Address % Used and Temporary Address % Used – If this number is increasing rapidly, it could mean that you have applications that are creating and deleting objects at a rapid pace, which could affect performance. When the temporary address percentage nears 60, consider doing an IPL to reset this value. (This may require a change to a System Service Tools setting. The IBM default is to reset on IPL only if the temporary address space is above 85%.)

DB Capability % - *The maximum CPU utilization available for database processing on this server.* This value was originally added for Domino-dedicated systems but was later added as a statistic to all systems. Generally, it indicates that there is SQL activity going on.

There is a performance trade-off to collect this information. Per IBM, "System support is changed in i 7.1 to not collect CPU utilization data specific to database processing. Interfaces that report database CPU utilization data such as WRKSYSACT now show a zero in the database CPU utilization fields."

System ASP Size, System ASP % Used, and Total Storage – *The amount of hard drive disk space on your system versus the total and available amount in the system ASP (auxiliary storage pool).* The ASP storage threshold is defined in SST (System Service Tools). If you have multiple auxiliary storage pools defined on your system, you'll want to use the WRKDSKSTS command for a better picture of the disk space used.

Generally, performance is affected when you reach 80% of system ASP used; some things stop working at 90% used. If you have a huge, multi-terabyte system, these values will be higher. At 100% your system will automatically IPL and may also delete all your spooled file reports.

Current and Maximum Unprotected Used – *The current amount of storage used for temporary objects and machine data (which can increase due to application memory leaks); and the most used since the system was IPLed.* These are reset when the system is restarted. Generally, a value of 5% or less of the total ASP used is considered to be acceptable for unprotected used.

DSPSYSSTS Memory Pool Values: System memory pools are numbered 1 to 64. Typically, there are four predefined memory pools on your system:

- System Pool 1 - *MACHINE pool where the operating system and LIC reside.
- System Pool 2 - *BASE pool where all unused memory is placed and which most jobs typically use as that is the default memory pool for most jobs.
- System Pool 3, System Pool 4, and Above - May change based on the order in which the subsystems start; but typically system pool 3 is the *INTERACT pool and pool 4 is the *SPOOL pool for print writers.

To prevent memory conflicts, previous administrators or application providers may have created additional pools to segregate work on your system. Workloads that are similar should be routed into shared pools (i.e. interactive subsystems into the interactive pool, batch subsystems into a batch type memory pool). Pool size is determined with a combination of system values, the WRKSHRPOOL screen, and the CHGSBSD command for private pools. The built-in performance adjuster—if turned on with system value QPFRADJ and set to a 2 or 3—will adjust memory pools based on demand and threshold values that are set in system values and the WRKSHRPOOL screen.

Machine Pool – The amount of memory allocated to this particular pool will change based on the automated performance tuning adjustments (system value QPFRADJ); or by other performance tuners such as Robot/AUTOTUNE. System value QMCHPOOL is the minimum machine pool size. The performance adjuster has been known to adjust the machine pool up to 40% of the total memory on the system due to shared machine and operating system programs. The performance adjuster will attempt to adjust this pool until the faulting rate is below 10. It is critical that the faulting rate for this pool be kept below 10 faults per second (see database and non-database faulting below).

Reserved Size (Memory Pool) – This value is calculated by the operating system, is for system use only, and cannot be used by jobs. It is critical that enough memory be allocated to the machine pool at the outset so that reserved size does not exhaust it. The QMCHPOOL system value will be changed by the built-in performance adjuster automatically so a manual adjustment should not be necessary.

As a starting point, we recommended these minimum pool sizes for systems with 10GB main memory or more:

- 7-10% of total memory for *MACHINE (system pool 1)
- 5-8% of total for *BASE memory pool (system pool 2)
- 1 MB per interactive session for *INTERACT (system pool 3)
- 1 MB for *SPOOL (system pool 4)
- Custom batch memory pools should be large enough for 5 MB per batch job at a minimum.

Web and Java applications have specific tuning requirements, but typical of those applications are their multi-threaded characteristics. Those processes need special handling and will cause the Max Active Thread requirements to go up. Use WRKACTJOB OUTPUT (*PRINT) or WRKACTJOB F11-twice to see the thread count for those jobs. The performance tuner, system value QPFRADJ, will adjust your pool sizes and activity levels based on this type of activity as well.

Max Active Threads (by memory pool) – *Sets the number of jobs that can have a status of running on the WRKACTJOB panel and get an activity level at one time.* This setting is more art than science and can greatly affect the performance of the work on your system.

Generally, allowing a large number of jobs to get an activity level from the CPU is not good. Your throughput may actually be better by having a smaller Max Active, allowing its work to get done, before moving on to new work. For instance, if your interactive pool has a Max Active of 25, that means 25 users could hit the Enter key at one time and any others would be input-inhibited until there was an opening in the one of the 25 occupied activity levels.

Paging Option – Also known as "expert cache," this value can be set to *CALC for all shared memory pools. System pool 1 is reserved for the operating system and LIC and must remain as *FIXED.

*CALC uses an algorithm that calculates a read-ahead value that will bring blocks of data into main memory of an increasingly large amount, which makes data available for their respective programs to avoid "faulting". *A fault means data is not in main memory and needs to be loaded from auxiliary storage.* *CALC works best for memory pools doing batch work.

Database and Non-Database Faults – Also known as a page fault, database faults are measured in faults per second. This value indicates that blocks of memory from auxiliary storage are being read into memory. It may be programs (non-database fault) or records from DB2 (database fault).

As mentioned in Paging Option, a high page fault rate will adversely affect system performance. Acceptable faulting rates are not easily calculated without referring to additional IBM documentation. Generally, a high faulting rate indicates poorer system performance and can be adjusted by increasing Pool Size, decreasing Max Active, or both. However, for system pool 1 (*MACHINE), IBM recommends a faulting rate of no more than 10 for combined database and non-database faults. Too many active jobs or threads and not enough memory means that programs and data will continually be paged in and out of disk, which is known as "thrashing." As the name applies, "thrashing" can and will bring your processing to a virtual standstill if you aren't doing automated performance tuning.

Active to Wait – *Measured in transactions per minute, a job in the active state is crunching data and actively using system CPU.* If it goes into a short wait (Active to Wait), such as for disk I/O, it keeps its activity level, which allows it to access CPU time. When a job has a long wait for a system resource, such as a tape drive response or input from a user, it loses its activity level, which allows other jobs to access CPU time.

Wait to Ineligible – *Jobs that use up their CPU time slice, have a long wait for a system resource, or lose their activity level go to an ineligible status.* They are prioritized based on run priority and several other factors; eventually, when an activity level opens up again, they get their turn for additional CPU cycles. The process repeats until the job is complete. Generally, IBM says that if the Wait to Ineligible is always zero, your activity level for that pool might be set too high and the pool not using all its activity levels. Another rule of thumb: your Wait to Ineligible should not be more than 20% of your Active to Wait value. If the ratio is too high, too many jobs are being paged out to auxiliary storage and back in.

Active to Ineligible – *Shows how often jobs are moved to the ineligible queue without getting an activity level.* This number should be as low as possible without being zero. If it is always zero, then there is always an available activity level, which means the pool's activity level is set too high.

Work with Disk Status metrics (WRKDSKSTS):

Disk % Busy by Disk Unit – Values of 40% and below are considered to be acceptable for Disk Busy. Greater than 40% may indicate that data has not been spread to enough disk units; and that you may need additional disk units and disk arms to ease the load.

Disk % Used by Disk Unit – *Indicates how data is spread across your disk units.* Each disk unit is assigned to an ASP; as data is written to disk it is scattered evenly across all units in the pool of disks in the ASP. ASP information for each disk unit can be viewed from the WRKDSKSTS command (press F11).

Disk Average Response Time for a Disk Unit – It is difficult to recommend a guideline for "good" disk response time. There are many factors including various disk technologies, the ability for a disk to take advantage of cache, number of reads versus writes, and an application's tolerance to disk delays.

With current disk technology, an average response time of less than 5 milliseconds is good; between 5 and 10 milliseconds is normal; above 10 milliseconds requires analysis; and above 100 milliseconds is bad. As disk technology undergoes new trends—such as the adoption of solid-state disk drives—these values will change.

Disk Protection Status – If system performance is slow but your CPU usage is low, check this column in the WRKDSKSTS information. If this status is DEGRADED or FAILED the disk unit should be replaced; contact your hardware provider immediately. You will also receive an informational message in the QSYSOPR message queue with this information.

Additional disk drive statistics available from Collection Services:

- Disk Unit Size (Disk Unit)
- Disk Data per Read (Disk Unit)
- Disk Data per Write (Disk Unit)
- Disk I/O Requests (Disk Unit)
- Disk Reads per Second (Disk Unit)
- Disk Request Size (Disk Unit)
- Disk Writes per Second (by disk unit)

To establish an appropriate threshold, first collect the data for a metric over several weeks. Then develop a baseline for the non-static values, such as Reads per Second, and set your monitoring threshold a bit higher than the historical high.

Other performance factors supplied by Collection Services follow. They should be checked if users are experiencing poor application performance while your system is using only a small amount of CPU.

- Maximum Job Seize/Wait Time
- Synchronous Lock Conflicts
- System Seize Count
- System Seize/Wait Time
- Asynchronous Lock Conflicts
- Average Job Seize/Wait Time

A *seize* is a wait point created for processes where many can access the same object at the same time. A record lock is a specific wait point; waiting for a data queue update is a wait point; waiting for an index to be updated on a database is a wait point. The counts and times for all these different wait points are independently recorded in the performance data files.

Think of seizes as the Licensed Internal Code's (LIC's) equivalent of locks. A seize almost always occurs on or against an MI object (like a DB2 physical file member, Data Queue, Program, Library, or User Profile). Seizes can conflict with locks and can cause lock conflicts. For example, if there are many object creates and deletes happening for a user, there will be many waits for the owning user profile to be updated.

Seizing objects is a normal system function. It is a system control to allow only one object access to another object. When there is a large amount of seize/wait time, investigate the holder of the seize to find out why the job has held a seize on the object for so long.

The ideal case is for small or no wait times, which indicate that jobs did not have to wait to get a seize.

When Collection Services Isn't Enough

While it serves as a powerful base for monitoring, Collection Services is not sufficient to ensure the reliable performance of your system. Third-party tools build on that base with more options for your performance collection. When searching for a tool, look for ones that give you the ability to:

Establish unique collection intervals for each system. A development box, for example, will not need collection nearly as often as a production box. If your tool doesn't let you establish a longer interval, you will receive unnecessary information and degrade performance.

Dictate the level of detail in your statistics. Analyzing performance metrics is impossible without the right information.

Establish thresholds on data. Thresholds are central to performance collection. Without thresholds, you run the same risk of overuse as someone who doesn't monitor. Smart use of thresholds can save valuable resources and prevent system firefighting.

Alert in real-time when thresholds are exceeded. Real-time alerts are the companion to smart thresholds. Without alerting, thresholds are powerless—the operator must know about a problem to be able to address it.

Create graphs, dashboards, and reports. Rich interfaces animate data in a way that is easy to understand and is useful for reports.

Consolidate data from multiple systems into a central location for reporting and alerting.

The ideal tool will also provide adequate support that is available when operators need it. While guides like this white paper provide operators with general guidelines, support staff for the tool may need to help you configure settings in more detail to fit your system. Ideally, support should be available 24 hours a day, 7 days a week to deal with issues and questions whenever they might arise.

Consequences of Inadequate Monitoring

Inadequate monitoring can produce a range of outcomes from sluggish performance to interruption and outages. In every industry, business-critical processes depend on application performance. Interruption of real-time processing, customer service delays, and missed SLAs and deadlines are all plausible outcomes with all-too-real consequences.

Fortunately, IBM i and third-party vendors offer significant power to prevent such disaster. Robot/NETWORK, the performance monitoring solution by Help/Systems, gives you the ability to establish, on unique collection intervals, multiple thresholds for performance and real-time alerts when they are exceeded. Its Performance Center offers both drill-down capacity with summary tabs and customizable dashboards, graphs, and reports to animate current and historical data. It also brings data across IBM i to a central location for reporting and alerting.

Conclusion

It is essential that your systems administrators have Collection Services running and that they know how to maintain it. The benefits of monitoring your performance data—analyzing application performance, managing by exception, and planning for the future—are too great to leave the feature unused. The consequences of lax monitoring are equally compelling: jeopardized deadlines, missed SLAs, interruption of real-time processing, customer service delays, and any number of disaster scenarios that disrupt your business and hurt your reputation.

For More Information

Call us at **1-800-328-1000** or email **info@helpsystems.com** to set up a personal consultation to review your current setup and see how the Robot products can help you achieve your automation goals.

help/systems

World's Leader in Power Systems™ Software Solutions

www.helpsystems.com | +1 (952) 933-0609 | info@helpsystems.com