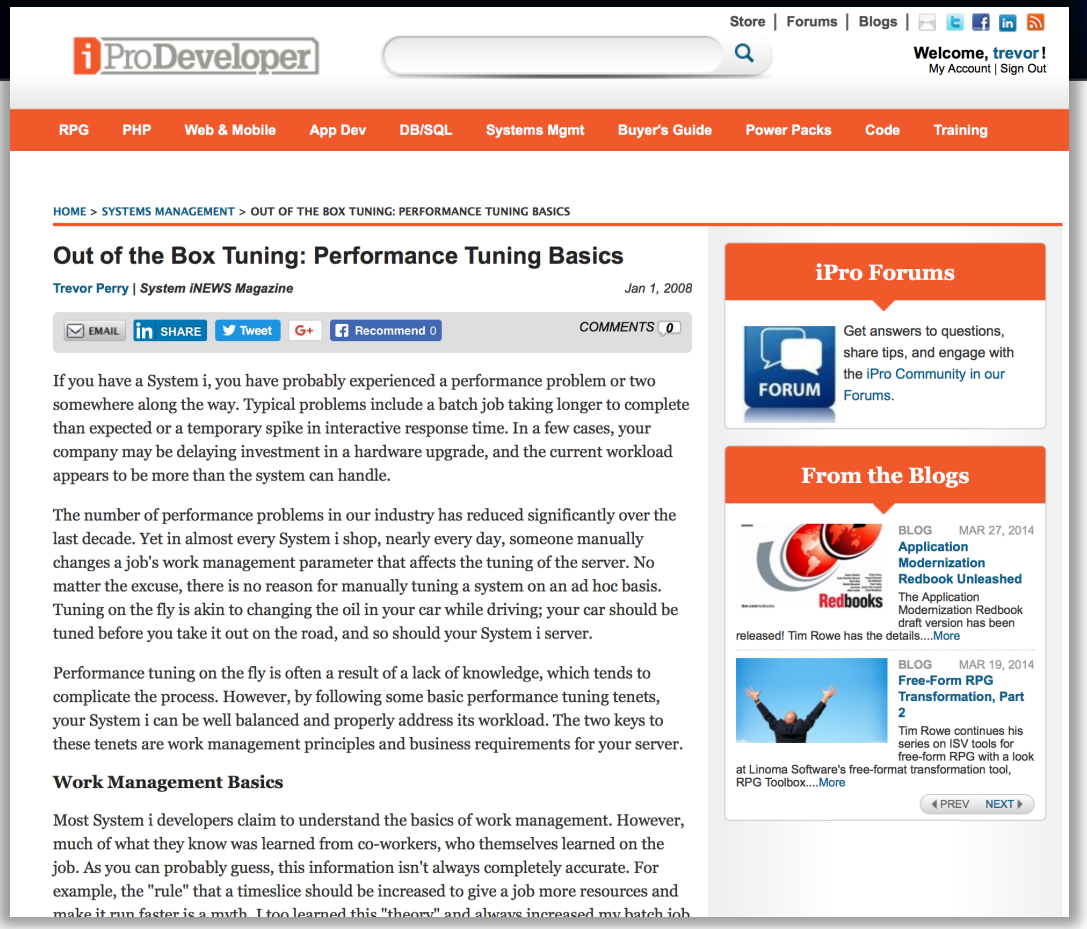
Performance Tuning
Back to Basics

Trevor Perry
FrescheThinker

IBM CHAMPION

@ericjooka

iProDeveloper Articles



@ericjooka

iProDeveloper Articles

- Performance-Tuning Basics
  - January 2008
- System Values Tuning
  - February 2008
- Subsystems and Memory Pools
  - June 2008
- Work Management Configurations
  - December 2008
- Tuning Out of the Box: The Silver Bullet
  - August, 2009

Optimum Performance

@ericjooka

What is Optimum Performance?

- You cannot make your system run any faster
- You can only allocate resources so that the system performs more efficiently
- Allocating resources
  - Fewer visible performance spikes
  - Improved response time
  - Better throughput
- It may appear that the system is running faster, but it is just running *better*

How do I achieve Optimum Performance?

- Resources are available
- Resources are allocated to the tasks that need them
- Interactive response times are consistently good
- Batch jobs finish in a timely manner

- Balancing the workload according to the needs of the business

Remember

- If memory is already being allocated poorly
- Purchasing additional memory means
  - **More** memory will be allocated poorly
  - The performance problem may *temporarily* appear to be resolved
- A better long-term solution is to configure according to your business needs

Performance Tuning Basics

@ericjooka

Performance Issues

- Typical problems
  - batch job taking longer to complete than expected
  - temporary spike in interactive response time
- Performance problems have reduced significantly
  - Someone manually changes a job's work management parameter
  - Affects the tuning of the server
- Performance tuning on the fly is often a result of a lack of knowledge

Two Key Tenets

- Well balanced
- Properly address the business workload

A Work Management Myth

- Rule to increase timeslice
  - 15000 timeslice = 16 hours
  - 500 timeslice = 9 hours

- Timeslice settings can certainly have an impact on a server
- Changing one job at a time
  - Unbalances the allocation of resources
  - Causes performance problems

Work Management Basics

- Every developer should have basic work management skills
  - How work begins
  - Routing entries
  - How to set system values
  - Connect shared pools to subsystems
- Learn these first
- Performance tuning becomes simple

Business Resource Requirements

- Every developer has his or her own theory about tuning
- Every performance tuning consultant or "expert" also has their own theory
- IBM has performance experts who will solve individual performance concerns

- People who can tune performance correctly
  - Understand what is running on the server
    - Applications
    - Business load
  - Understand the user requirements.

Business Resource Requirements

- Different types of work
  - Interactive
  - Batch
  - Web
  - Database connection
- Which applications are using what type of work and in what balance?

Methodology

- Establish some basic goals
- Measure performance against those goals
- Plan and implement work management changes
- Measure the impact of those changes
- Repeat forever

- Unique to your business and workload

Establishing Goals

- Traditional interactive jobs
  - Maximum response time for a percentage of the work
- Nontraditional interactive jobs
  - Response time for browser-based applications
  - Response time for a .NET application that connects with web services
- Database access jobs
  - Total time for a single transaction
- Batch jobs
  - Throughput goal (specific number of batch jobs processed in a certain time frame)
  - Specific batch job or a stream of batch jobs
- Other unique processing requirements

Establishing Goals

- Goals are unique to the workload that needs to be managed
- Key is to be specific
  - The more specific the goals
  - The easier it will be to measure the performance against these goals
- Review and update your goals regularly
  - Processing requirements are rarely static
  - Adding applications impacts workload and system balance

Measuring Performance

- A collection of historical data
- Avoid using resource-hungry green-screen commands
  - Work with Active Jobs (WRKACTJOB)
  - Work with System Status (WRKSYSSTS)
  - Work with System Activity (WRKSYSACT)
- Use IBM i Navigator
  - Collect data in 15-minute intervals
  - Valid performance measurement without affecting system performance

What I said back then…

- *"All the information needed for the performance methodology that I advocate can be collected with green-screen commands; however, the primary System i management tool is iSeries Navigator, and it should be the main tool that you use, especially because future releases will contain functions unavailable via green-screen commands."*

Power Systems

IBM

Navigating IBM i Performance – Tools and Best Practices

Dawn May – dmmay@us.ibm.com
@DawnMayiCan

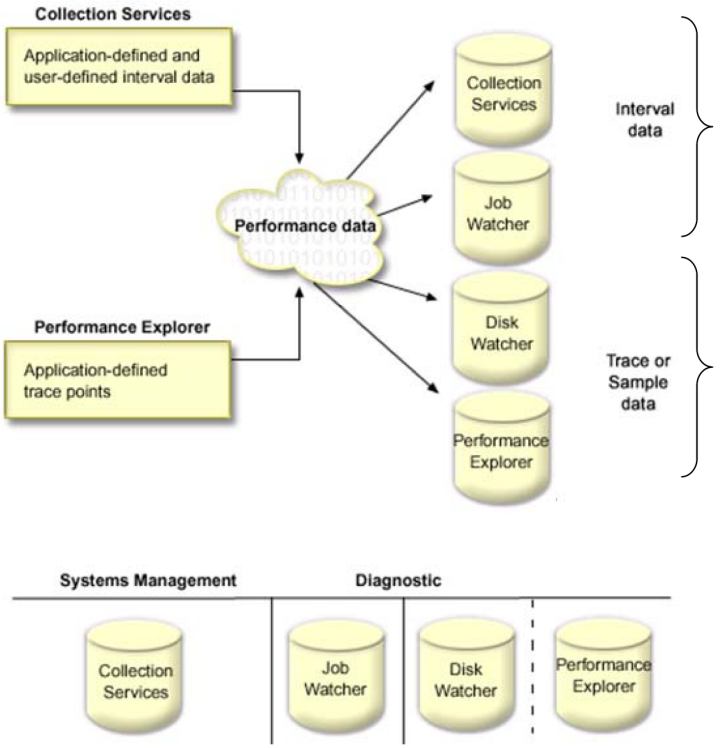
IBM

Power Systems

IBM

Performance Data Collection Architecture

- Collection Services
- Job Watcher
- Disk Watcher
- Performance Explorer



What I said back then…

- *"All the information needed for the performance methodology that I advocate can be collected with green-screen commands; however, the primary System i management tool is iSeries Navigator, and it should be the main tool that you use, especially because future releases will contain functions unavailable via green-screen commands."*

IBM i Systems Management Interfaces

Power Systems

IBM

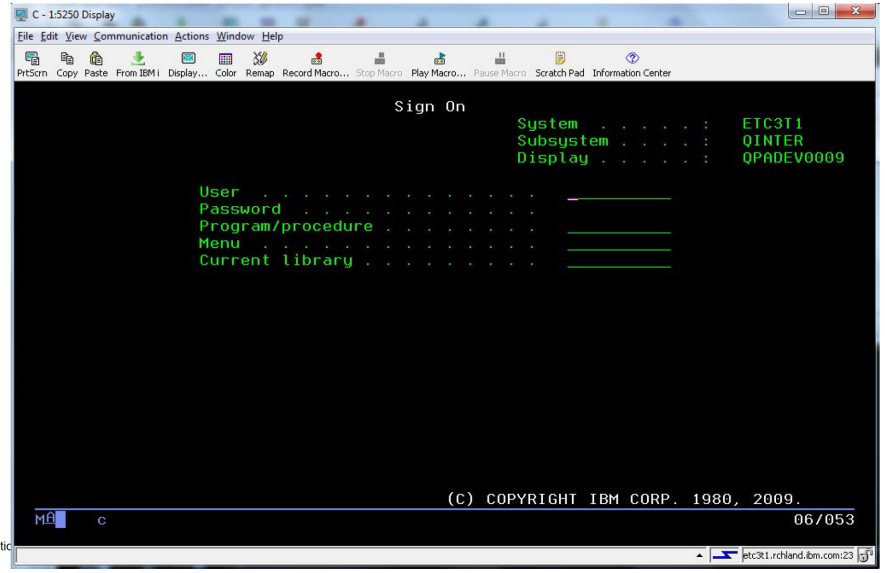
IBM i

Power Systems

IBM

Green Screen

I'm not going to spend much time on the *old-fashioned* stuff….



Power Systems

IBM

IBM i

© Copyright IBM Corporation 2014

Power Systems

IBM

Green Scr

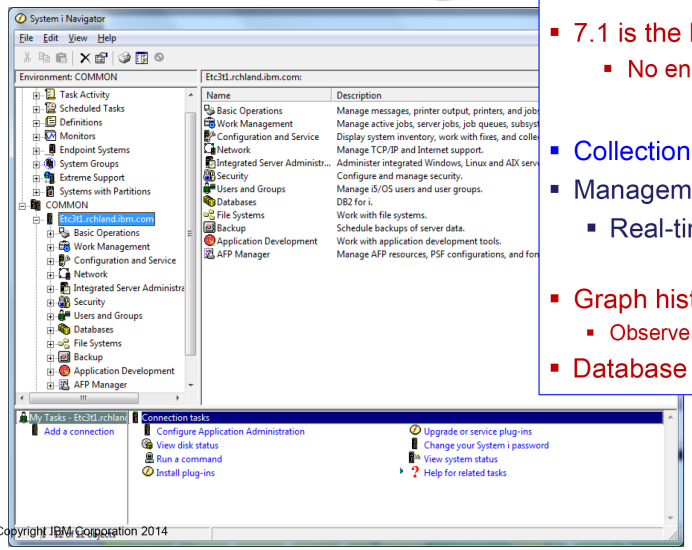
I'm not goin

© Copyright IBM Corporatio

Power Systems

System i Navigator

I'm also not going to spend very much time on the *middle-aged* stuff….



- *Also known as ....*
  - *iSeries Navigator*
  - *Operations Navigator*
- Windows client application
- Part of the iAccess for Windows product
- 7.1 is the **last** release
  - No enhancements
- Collection Services
- Management Central Monitors
  - Real-time monitoring
- Graph history
  - Observe performance metrics over time
- Database tools

Power Systems

IBM

IBM i

© Copyright IBM Corporation 2014

Power Systems

Green Scr

I'm not goin

IBM

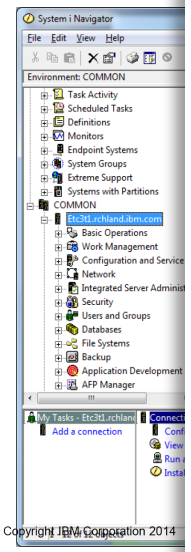
© Copyright IBM Corporatio

Power Systems

System i N

I'm also not g

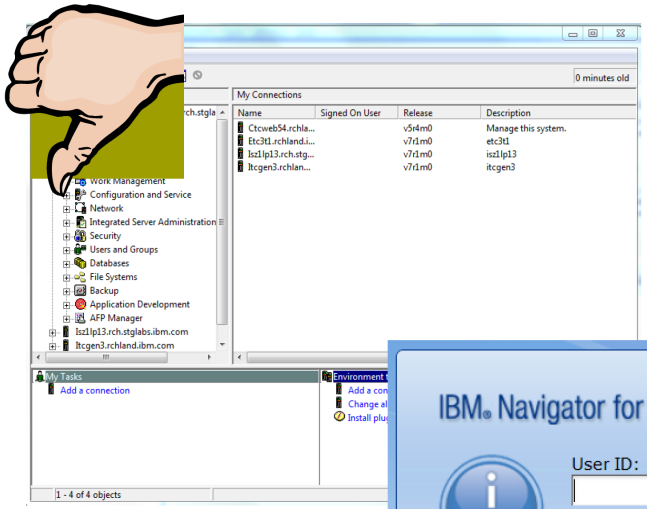
on the *middle-*



© Copyright IBM Corporation 2014

- *Also known as ....*

7.1 is the final release of System i Navigator.
It is not refreshed for 7.2.



© Copyright IBM Corporation 2014

12

Power Systems

IBM

IBM i

Power Systems

IBM

Green Scr

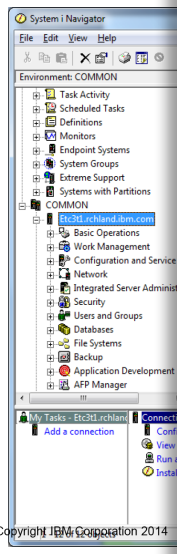
I'm not goin

Power Systems

System i N

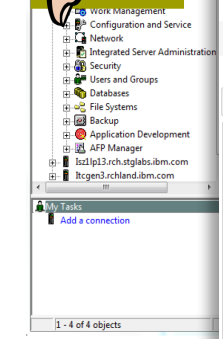
I'm also not g

on the *middle-*



- *Also known as ....*

7.1 is the final release of System i Navigator.

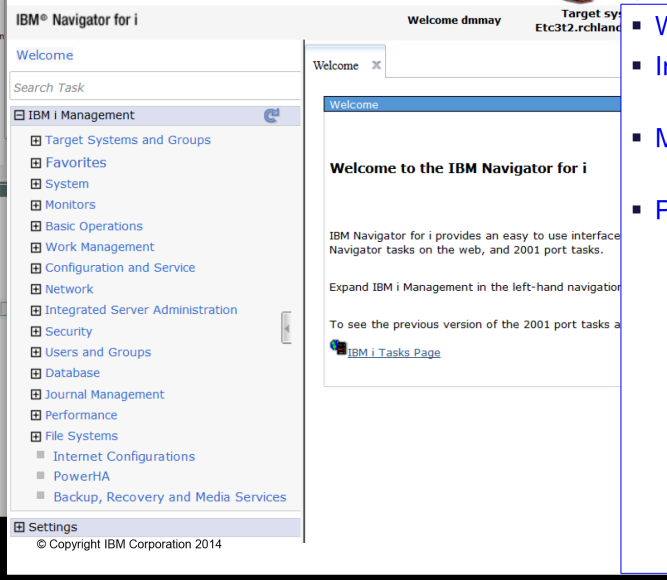


Power Systems

http://systemname:2001

IBM Navigator for i

I will spend most of the time on the *current* stuff….



- Web application
- Included with IBM i 6.1 and later
- Monitors
- Performance tasks:
  - Performance Data Investigator
    - Health indicators
    - Collection Services
    - Database
    - Job Watcher
    - Disk Watcher
    - Performance Explorer
  - Collection Manager
  - Reports
  - Sizing

Work Management Changes

- Carefully plan and document EVERY change
- Implement changes
  - When the server is in a restricted state
  - When the fewest users will be affected by the immediate changes
- Rarely change work management configuration on the fly
  - A mix of like jobs with different run priorities, which causes imbalance

Once the methodology is underway…

- Change only one work management parameter at a time
- Two or more separate changes
  - Might cause one positive result and one negative result
  - Won't know which one was positive
- Understand what comprises a single work management change
  - Shared pool sizes – one memory pool at a time
  - Floor and ceiling limits - one memory pool at a time
  - Run Priority attribute of a class object - all the class objects together

Rinse and Repeat

- Three choices
  - Reverse the change
  - Make additional changes
  - Stay put
- After maximizing the positive impact of a particular change
  - Move on to the next work management change
  - Repeat

Remember

- You cannot be an expert without knowing the business requirements
- Don't be reactive
- Tune for balance.
- You cannot make the server run faster
- You can improve throughput.

System Values Tuning

@ericjooka

System Values Tuning

- Many resource-allocation system values affect performance
- Goal is: OS spends less time managing itself and more time managing work

Automatic Performance Adjustment (QPFRADJ)

- Set to Automatic Adjustment
- Never let your system be adjusted at IPL (i.e., system restart)
  - Replaces a significant portion of your configuration with its own "guess”
- Some shops leave this turned off
  - After their shared memory pool limits correctly established

Active and Total Job Settings

- How much space the system allocates for managing system jobs
  - QACTJOB = *initial number of active jobs for which auxiliary storage is allocated at restart*
  - QTOTJOB = *minimum number of jobs for which storage is allocated at restart*
  - QADLACTJ = *additional number of active jobs for which auxiliary storage is to be allocated when the initial number of active jobs at restart is reached*
  - QADLTOTJ = *additional number of jobs for which auxiliary storage is to be allocated when the initial number of jobs at restart is reached*

QTOTJOB and QADLTOTJ

- At IPL, OS sets aside some working room for all the jobs that it will manage
  - Uses the QTOTJOB value to determine how much working room to allocate
- As more work is performed, the number of total jobs might be exceeded
  - Allocates more resources based on the QADLTOTJ value
- Cycle continues until the system is IPLed again

QTOTJOB and QADLTOTJ

- Determining the optimum settings for these system values is important
- *Too small?*
  - OS has to spend time allocating a small amount of resources a lot
  - OS spends a lot of time managing its own workspace and less time managing the work
- *Too large?*
  - OS reserves too much workspace
  - OS could be reserving resources that could be used to help work process efficiently

QTOTJOB and QADLTOTJ

- Use IBM i Navigator to find the current total job count and active job count
  - Use your trend data
  - Include heavy periods – Month end, etc.
- Set QTOTJOB and QADLTOTJ values
  - OS allocates enough workspace to minimize the number of times it needs to allocate more workspace
- Example (fictional numbers):
  - 100 jobs at 9:00 a.m., 150 jobs at noon, 200 jobs at 3:00 p.m., 50 jobs at midnight
  - QTOTJOB = 125
  - QADLTOTJ = 25
  - OS has resources for all the jobs when the day starts
  - OS will allocate more workspace three times each day

Active Jobs

- Active jobs may end without leaving the system
- Spooled files remaining can cause that job to be included in the total job count
- QSPLFACN = *detach printer output after jobs have ended*
  - Default is set to keep spooled files attached to jobs
    - Total job count includes every job that has a remaining spooled file

Printed Output

- Reduce the number of spooled files remaining in your system
- Archive or offload them to a content management application
- Asking the OS to manage takes resources away from current work
- QRCLSPLSTG = automatically clean up unused printer output storage
- Requires an understanding of the pattern of the work on your system
- Find a balance
  - Not spending a lot of time managing these internal data files
  - Not wasting too much space on deleted spooled files
  - Simple rule of thumb
    - Large turnover of spooled files = smaller number of days
    - Small number of spooled files with a low turnover rate = larger number of days

Subsystems and Memory Pools

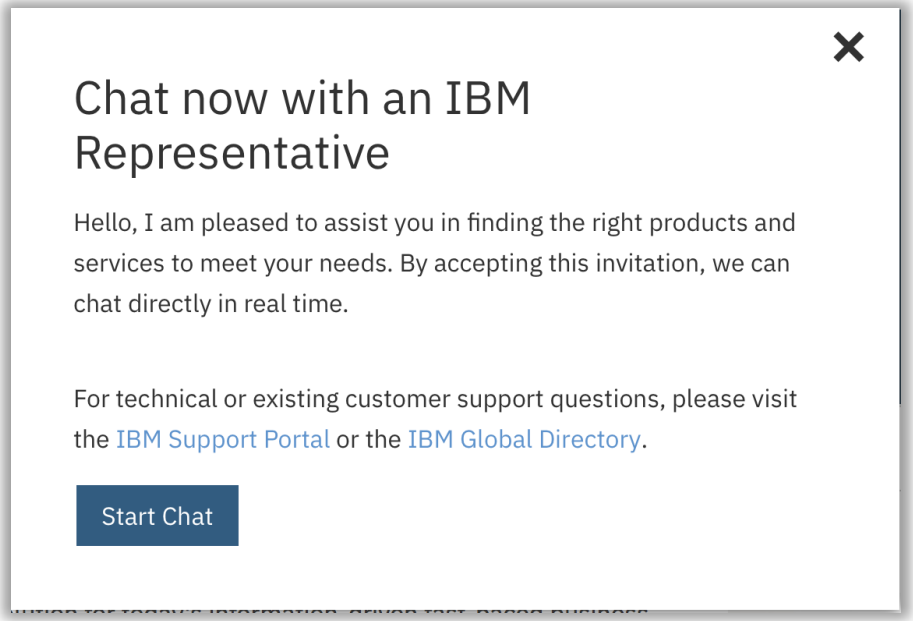
@ericjooka

Work Management

- Originally: System i Systems management Work Management
- Now: https://www-03.ibm.com/systems/power/software/i/management/

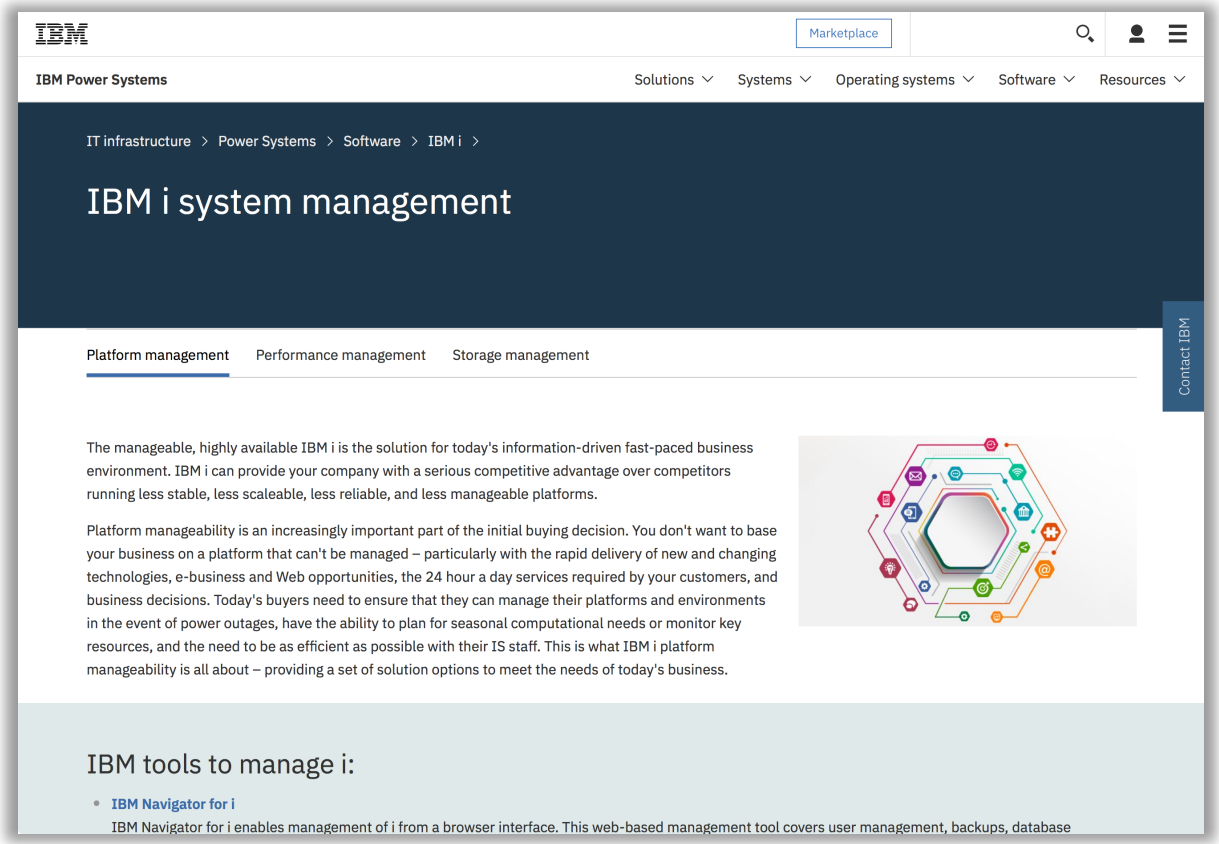
Work Management

- Originally: System i Systems management Work Management
- Now: https://www-03.ibm.com/systems/power/software/i/management/



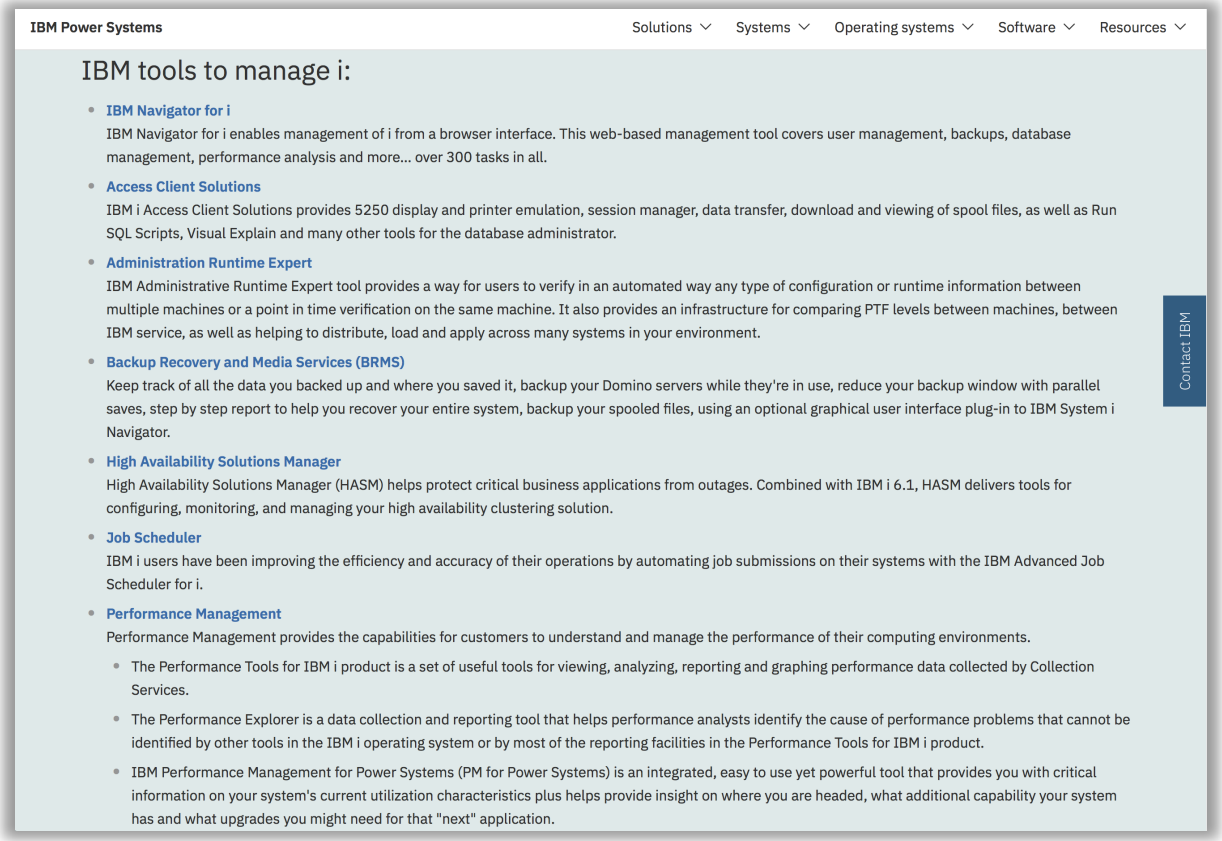
@ericjooka

ibm.com/systems/power/software/i/management/



@ericjooka

ibm.com/systems/power/software/i/management/



IBM Power Systems

Solutions | Systems | Operating systems | Software | Resources

IBM tools to manage i:

- **IBM Navigator for i**
  IBM Navigator for i enables management of i from a browser interface. This web-based management tool covers user management, backups, database management, performance analysis and more... over 300 tasks in all.
- **Access Client Solutions**
  IBM i Access Client Solutions provides 5250 display and printer emulation, session manager, data transfer, download and viewing of spool files, as well as Run SQL Scripts, Visual Explain and many other tools for the database administrator.
- **Administration Runtime Expert**
  IBM Administrative Runtime Expert tool provides a way for users to verify in an automated way any type of configuration or runtime information between multiple machines or a point in time verification on the same machine. It also provides an infrastructure for comparing PTF levels between machines, between IBM service, as well as helping to distribute, load and apply across many systems in your environment.
- **Backup Recovery and Media Services (BRMS)**
  Keep track of all the data you backed up and where you saved it, backup your Domino servers while they're in use, reduce your backup window with parallel saves, step by step report to help you recover your entire system, backup your spooled files, using an optional graphical user interface plug-in to IBM System i Navigator.
- **High Availability Solutions Manager**
  High Availability Solutions Manager (HASM) helps protect critical business applications from outages. Combined with IBM i 6.1, HASM delivers tools for configuring, monitoring, and managing your high availability clustering solution.
- **Job Scheduler**
  IBM i users have been improving the efficiency and accuracy of their operations by automating job submissions on their systems with the IBM Advanced Job Scheduler for i.
- **Performance Management**
  Performance Management provides the capabilities for customers to understand and manage the performance of their computing environments.
  - The Performance Tools for IBM i product is a set of useful tools for viewing, analyzing, reporting and graphing performance data collected by Collection Services.
  - The Performance Explorer is a data collection and reporting tool that helps performance analysts identify the cause of performance problems that cannot be identified by other tools in the IBM i operating system or by most of the reporting facilities in the Performance Tools for IBM i product.
  - IBM Performance Management for Power Systems (PM for Power Systems) is an integrated, easy to use yet powerful tool that provides you with critical information on your system's current utilization characteristics plus helps provide insight on where you are headed, what additional capability your system has and what upgrades you might need for that "next" application.

Contact IBM

@ericjooka

Subsystems and Memory Pools

- Work is managed within subsystems using memory pools
- Must establish a correct combination of shared pools connected to subsystems
- Consider two things
  - Running similar types of jobs together in subsystems
  - Allocating enough memory to each job so it can run according to your methodology

Basics of Work Management

- Do you understand how to manage work in a subsystem?
- Do you know what a routing entry is?
- Do you get the relationship between a work entry and a routing entry?
- Are you familiar with a job's "routing step”?
- Where is the routing data identified for each job?
- Do you know the routing entry uses the routing data to assign the class?
- Do you know the class assigns the runtime priority and timeslice to the job?

Memory Pools

- Private pools
  - Memory allocated for use on only one subsystem
- Shared pools
  - Memory shared between multiple subsystems

Getting Started

- First, identify all the jobs running on your system
- Next, define the jobs by their usage type and amount

identify all the jobs running on your system

- Normal interactive jobs and batch jobs
- Jobs running at different schedules
  - month-end jobs, year-end jobs, one-time conversions, etc.
- Database connection jobs running to support various client applications
- Web server jobs – interactive and service-enabled
- All planned applications and expected growth in application use
- Backdoor applications
  - A job queue that is used to submit "special" work?
  - Unusual compiles
  - One-time repair jobs
  - Reports for executives

identify all the jobs running on your system

- Will require a detailed review of the performance data collected on your system
- It will take some time
- Attention to detail will make the difference
- Results will be true performance tuning

Define the jobs by their usage type and amount

- Interactive 5250 jobs use short bursts of resources and need fast response
  - High CPU usage and less database usage
- Basic batch jobs require extended access to database I/O
  - Less CPU
- Web jobs may be similar to 5250 interactive jobs
- Database access requires different resource requirements
  - Based on the connected application
- Group jobs by subcategory
  - Time-zone groupings?
  - Functional areas

Define the jobs by their usage type and amount

- List of groups of jobs similar in business need and resource requirements
- Each group can be managed together in a subsystem
  - If the subsystem is stopped and started at any time
  - All the jobs in that group will be affected in the same way at the same time
- Result is a complete list of potential subsystems

Define the jobs by their usage type and amount

- Next task is to make the list of subsystems smaller
- Fit the new groups into current subsystems where you can
- Balance the number of subsystems
  - With the amount of time you can spend on managing them

Configuring Work: Part One

- It will take time to configure all the subsystems for the first time
- Configure new and change subsystems in a progressive manner
- Use a development partition to test these changes
  - or
- Make changes one at a time
  - Check for repercussions
  - Reverse the change if needed
- Have patience
  - It may take several days or weeks to complete this task

Allocating Resources: Part One

- Decide which subsystems can share memory
- Small system example
  - One pool for interactive
  - One for batch
  - One for everything else
- Larger system example
  - Several pools for different batch subsystems
  - One for all outside database access
  - One for serving web applications
  - One for all interactive jobs

Allocating Resources: Part One

- Provide memory pools within which similar type jobs are running
- Similar jobs means jobs that require similar types of resources
  - A batch job with long database I/O should be run in a memory pool with other jobs requiring long database I/O
    - Interactive jobs requiring more CPU-bound activity with less database I/O should be run in a separate memory pool
- Jobs running in a single memory pool will steal resources
  - From other jobs in that memory pool as their resource needs grow
  - Release resources to as needs diminish

The shipped configuration

- *MACHINE pool
- *BASE pool
- *INTERACT pool
- *SPOOL pool

*MACHINE pool

- THE most important
- Where the operating system runs
- The smaller this memory pool, the slower the system will run
- If the pool is too large, memory is unavailable for the business
- Tune this pool so that its nondatabase faults are fewer than 10 per second
- Note this process may take some time
  - As you adjust the minimum pool size up or down
  - And review the results each time

Incorrect *MACHINE pool size

- System running well
- Work balanced
- Throughput within the goals
- Total CPU percentage remained below 45 percent
- Increased the minimum size of the machine pool memory
- CPU percentage was over 80 percent during peak times
- System appeared faster

Some observations

- Larger systems
  - Start with a minimum machine pool of 7.5% of total system memory
- Smaller systems
- Start with a minimum machine pool of 12.5% of total system memory. Of course, if you have a starting point inside that range, begin tuning at that point according to the 10-faults-per-second rule.

*BASE pool

- The default memory pool where the majority of work is performed
- Used by the auto-tuner when it needs to draw or return memory
- Always a minimum amount of memory remaining in the *BASE pool
- You'll first tune by moving work out of the *BASE pool
  - Into other shared and private pools
- If you plan to move most work out of *BASE
  - Minimum *BASE pool size should be close to 5 percent of total system memory
- If you plan to run some work in the *BASE pool
  - Minimum pool size may be over 10 percent of total system memory

After *MACHINE and *BASE

- Choose the remaining percentages by applying your business requirements
  - Small system example - with a lot of 5250 applications
    - *10 percent *MACHINE*
    - *10 percent *BASE*
    - *50 percent *INTERACT*
    - *20 percent batch*
    - *10 percent database*
  - Web server example
    - *10 percent *MACHINE*
    - *25 percent *BASE (including interactive)*
    - *65 percent web*

Shared vs Private memory pool

- Simple rules

1. *If a memory pool will be used by multiple subsystems*
   it must be a shared pool
2. *If a memory pool is to be used by only one subsystem*
   you want that pool to be adjusted to increase when it needs more memory
   you want that pool to be decreased when it needs less memory
   it must be a shared pool
3. *Those pools that are used by a single subsystem*
   and require a permanently fixed storage size
   are candidates to be considered private pools.

Unique for you

- Your server has unique business requirements
- It is rare that two servers should have identical configuration
- Don't be concerned about being overly precise at this point
  - You are creating a starting point
  - From which you will conduct further performance tuning

Floors and Ceilings

- Auto tuner limits
  - Based on minimum and maximum shared pool sizes
- Default minimums are very low
  Default maximums are always set to 100 percent
  - Auto-tuner will be working hard to constantly move memory around
  - If a pool has a job requiring a large amount of memory
    it is possible that without a ceiling, the system could become unbalanced

Floors and Ceilings

- Set the sizes for the *MACHINE and *BASE pools
  - Minimums match your memory allocation
  - Maximums should always be 100 percent
- All other memory pools
  - Minimum sizes of each pool about 10 percent lower than your plan
  - A reasonable maximum pool size to set a ceiling is important
    - Set too low will affect performance
    - Set too high will require the auto-tuner to overwork
    - A rule of thumb is maximum ~=150 percent of the minimum pool size
- *Note: you cannot assign more than 100 percent of your total memory to the minimum shared pool sizes*

Configuring Work: Part Two

- Get your list of shared and private pools
  - With a minimum and maximum pool size for each shared pool
- Configure the work management of the system to match your list
- You might choose to establish this pool configuration all at once
  - before or after a backup cycle where the system is in a restricted state.
- Or make the pool changes one at a time
  - Preferably starting with the smaller pools first

Work Management Configurations

@ericjooka

A Balanced Workload

- Your workload requirements are unique
- Your work management configuration needs to be unique
- Workload requirements can change at different times of any day, month, or year
- Provide several configurations for changing workloads

Attending to Special Jobs

- Certain server jobs run as prestart jobs
  - QUSRWRK
  - QSYSWRK
- Allocate a new shared memory pool
- Assign it to the subsystem
- Change the prestart job entry to use the new pool

Tuning It Up

- Certain server jobs run as prestart jobs
  - QUSRWRK
  - QSYSWRK
- Configuration
  - Initial number of jobs – default 1
  - Threshold – default 1
  - Additional number of jobs – default 2

Tuning It Up

- Certain server jobs run as prestart jobs
  - QUSRWRK
  - QSYSWRK
- Configuration
  - Initial number of jobs – default 1
  - Threshold – default 1
  - Additional number of jobs – default 2
- DO NOT USE THE DEFAULTS

The Silver Bullet

@ericjooka

Faster! Faster!

- A common myth is that increasing the timeslice will make a job run 'faster'
- IBM tells us that the timeslice is
  - *"The maximum amount of processor time that the system allows the job to run when it is allowed to begin"*
- The resource in this case is the processor
- The time slice is the amount of time the processor works for a job
  - A timeslice is not a resource
  - A timeslice is an indication of how to use a resource
- Increasing the timeslice does not apply more resource

Continuing…

- *"The time slice indicates the amount of time needed for the job to accomplish a meaningful amount of work...."*
- What is a 'meaningful' amount of work?
  - Interactive = the time between pressing Enter and the response being returned
    - If your interactive program is written well
      this amount of work is efficient and will not require a large amount of processing
    - Jobs with interactive attributes tend to have smaller timeslices
  - Batch = performing one transactionB
    - Batch jobs are intended to be long running, and require lots of processing
      their timeslices tend to be longer
- "Meaningful amount of work" will differ based on
  - Job attributes, the application, the company requirements

Continuing…

- *“When the time slice ends, the job waits while other queued jobs of the same or higher priority are allowed to run (up to the time specified in their time slices); then the job is given another time slice.”*
- “While the job is using the processor, no other jobs can. The longer your job is using the processor, the longer other jobs must wait for the processor”
- This refutes the myth of longer timeslices
- Increasing the timeslice of one job means all the other jobs requiring processor time will spend more time waiting

Faster! Faster!

- System/38 and CPF shipped timeslice defaults
  - interactive jobs = 2,000 milliseconds
  - batch jobs = 5,000 milliseconds
- S/38 processors had an internal timeslice of 500 milliseconds
- IBM considered 2,000 milliseconds to finish a “meaningful amount of work”
- Current processors can run thousands of times faster
  - How long does it take to do a “meaningful amount of work”?

Experiential Evidence

- A large amount of work to be done on a development system
- Evening 1
  - Set my batch job to a timeslice of 15,000
  - 16 hours to run
- Evening 2
  - Set my batch job to a timeslice of 15,000
  - 9 hours to run

Experiential Evidence

- A large amount of work to be done on a development system
- Evening 1
  - Set my batch job to a timeslice of 15,000
  - 16 hours to run
- Evening 2
  - Set my batch job to a timeslice of 15,000
  - 9 hours to run
  - Another programmer decided to cripple my job and set my timeslice to 500

Timeslice Research

- Types of jobs
  - Processor bound jobs
  - I/O bound jobs
  - Jobs with mixed resource requirements
- Job counts
  - One job running
  - Many jobs competing
- End result disproved the myth
  - The smaller the timeslices for all jobs on the system the more balanced the performance

The Silver Bullet Methodology

@ericjooka

The Silver Bullet Methodology

A. Monitoring
B. Configuration
C. Monitoring
D. Performance methodology enhancements

A. Monitoring

- Collect a base set of performance data as a base point for comparison
- Multiple months will allow trends to be identified
- One month is a good base point
- Without a base point you cannot prove any change has taken place

B. Configuration

- List all Class objects on your system (in use)
  - Identify currently defined timeslice
- Decide upon a new value for every timeslice on every Class object
  - Start with 10 percent of their current values
  - Reduce the timeslice for interactive jobs from 2,000 to 200
    batch jobs from 5,000 to 500
  - Apply the same rule for all timeslices, but do not reduce any below a value of 200
  - If a timeslice is set to over 5,000, reduce it to 500
- If you are feeling conservative, use a 50 percent reduction factor
  - If positive, repeat the process with a second 50 percent factor

B. Configuration

- Your server is unique
- The values you use should be determined by your workload
- If you have a batch-heavy system
  - Timeslices for batch should be treated with priority
- If you have long processing interactive jobs
  - Timeslices for interactive should be treated with priority
- The workload requirements on your server will dictate the new values

B. Configuration

- Write two programs
  - BEFORE: change the timeslice on all Class objects to their current values
  - AFTER: change the timeslice on all Class objects to their new value
- In a system restricted state, run the AFTER program
  - Changing ALL the timeslice values in the system together is important
- If you change the Class objects while the system is active
  - Your system will be unbalanced until an IPL
    or until the restricted state is reached and the system restarted

C. Monitoring

- Monitor the users and their experience with the system
- The first place where you will uncover any negative impact
- You will not likely hear from the users about any positive impact
  - Unless you solicit their experiences.

- *NOTE: If you decide that the impact is too severe on your system*
  - *Run the BEFORE program to reset all timeslice values*

C. Monitoring

- Monitor the system for a week
- Compare that to the previous week
- Compare that to the same week in the previous month (or months)

D. Performance Methodology Enhancements

- Add a rule to prevent changing the timeslice value on a Class object.
- Add a rule to prevent changing the timeslice value on any running job
- Add a rule to prevent changing the timeslice value on any queued job
- Add a check for all new Class objects
  - For new applications, new software packages, etc
  - Ensure they are changed to match the rest of the timeslices on the system

The Conclusion

@ericjooka

The Tenets of Performance Tuning

- Use a performance methodology
- Tune for balance and throughput
- Never change your configuration on the fly
- Monitor, monitor, monitor
- Change one thing at a time
- Repeat, repeat, repeat

The Tenets of Performance Tuning

- Establish a new performance regime for your company
- The end result is the best starting point for performance
- Follow the performance methodology rigorously
- If you encounter performance issues that required a more detailed level of attention and work management configuration
  - the consultant or the IBM rep will appreciate your diligence

One Final Caveat

- 95 percent of all performance issues are application program related

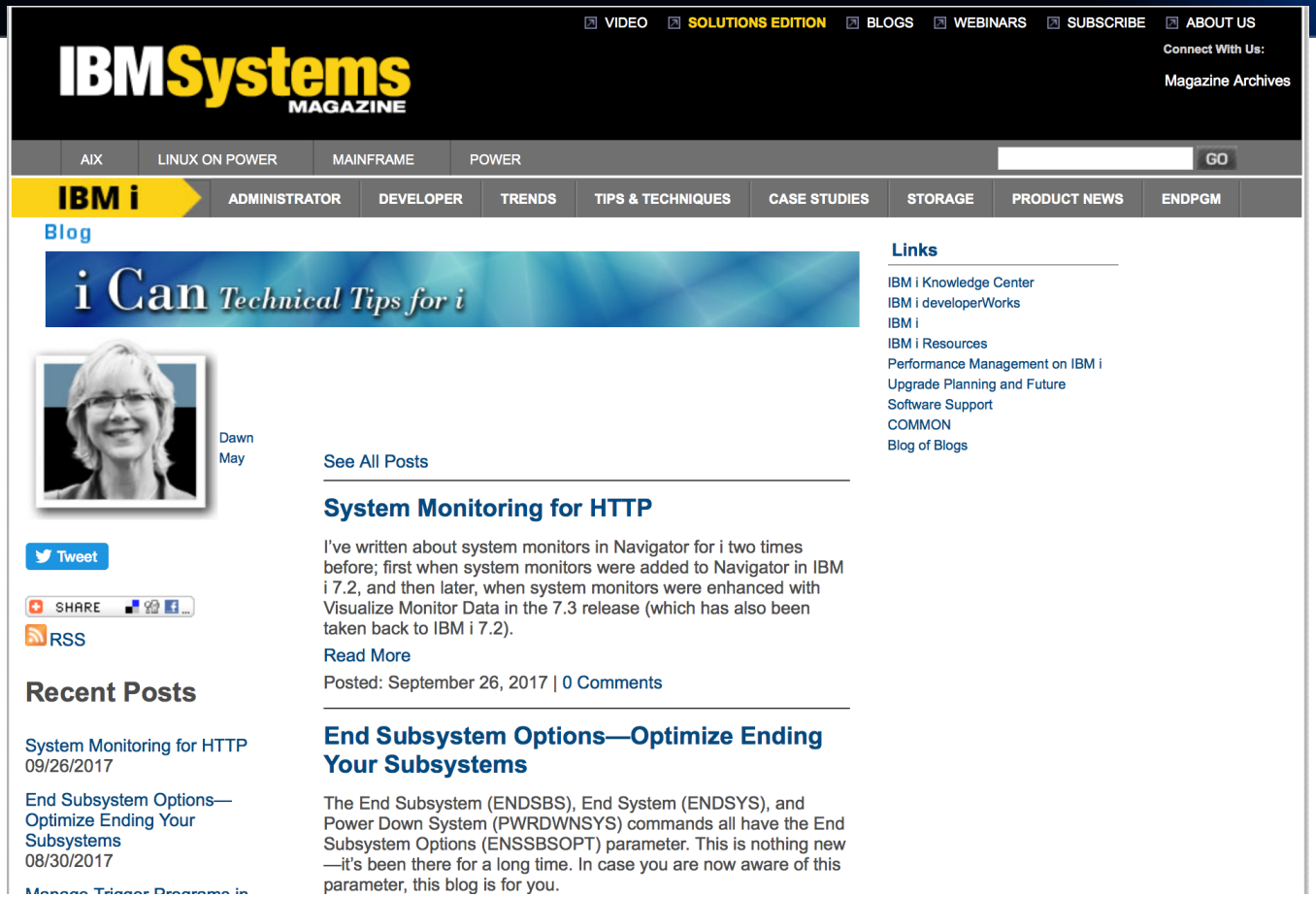
One Final Caveat

- 95 percent of all performance issues are application program related
- **This is not a myth**

What's Next?

@ericjooka

http://ibmsystemsmag.com/blogs/i-can/



@ericjooka

Power Systems

IBM

Navigating IBM i Performance – Tools and Best Practices

Dawn May – dmmay@us.ibm.com
@DawnMayiCan

IBM

Introduction to the IBM i Performance Data Investigator

Dawn May - dmmay@us.ibm.com
@DawnMayiCan

IBM i on Power - Performance FAQ

October 3, 2016

IBM Corporation

Database Performance Tuning

DB2 for i

Insight and perspectives on data management using IBM i

Friday, November 16, 2012

DB2 for i Database Engineer – A Description of the Job

For a number of years now, a few enlightened folks have been ~~raving about~~ sharing the importance of having someone in (or close to) your organization that focuses on DB2 for i.

Back in 2000, Kent Milligan posed the question: To DBA or Not to DBA? here

Back in 2009, I made the case for a DBA on IBM i here

Two months ago, Jon Paris and Susan Gantner asked: Who Needs a DBA? here

Last month at the Fall 2012 RPG & DB2 Summit conference I led my audience to the answer in a session entitled: To DBA or Not to DBA?

I'm getting dizzy from revisiting this topic over and over again. Let me regain my balance and say once again, it is important AND advantageous to have someone who is knowledgeable, skilled and focused on DB2 for i.

What I tell IT executives every chance I get:

It is a critical success factor to have a DB2 for i database engineer PERIOD

Popular Posts

- DBE Job Description
- In Memory Database
- SMP and Parallelism
- Single Level Storage

Links

- IBM DB2 for i site
- IBM DB2 for i Wiki
- IBM DB2 for i Technology Updates wiki
- IBM DB2 Web Query blog

Follow by Email

Email address... Submit

@ericjooka

One Final Caveat

- 95 percent of all performance issues are application program related
- **This is not a myth**

Performance Tuning
Back to Basics

Trevor Perry
FrescheThinker

IBM CHAMPION

@ericjooka