IBM

Cognitive Systems

IBM i Performance Tools for Application Developers

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

Cognitive Systems

IBM

Performance Disclaimer

- "it depends …"
- Performance information and recommendations in this presentation are based on measurements, analysis, and projections in a controlled environment for specific performance workloads.
- Your results may vary significantly and are dependent on the application and configuration.
- This information is provided along with general recommendations for you to better understand system performance.
- Information is provided *AS IS* without warranty of any kind.

Cognitive Systems

IBM

Agenda

- Brief Overview
- Introduction to IBM i Wait Accounting
- Performance Data Collectors
  - Collection Services
  - Job Watcher
  - Performance Explorer
- Performance Data Visualization and Diagnostics
  - Performance Data Investigator
  - iDoctor
- Examples

Cognitive Systems

IBM

Keep Current on PTFs

It's always good practice to keep current on the latest fixes from IBM

- PTFs address defects
- PTFs introduce new capabilities
  - IBM i Technology Refresh Updates
  - IBM i Group PTFs
    - Database
    - Performance tools
    - Java
    - HTTP Server
      - HTTP Server Group PTF for latest Navigator for i functionality
  - PTFs for performance data collectors
    - Collection Services, Job Watcher, Disk Watcher, Performance Explorer

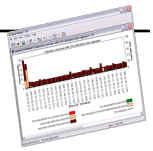
Cognitive Systems

IBM

IBM i Performance Tools

Green Screen
- Performance Tools Reports
- System commands
- Performance Explorer

System i Navigator
- Perf Tools GUI
- DB/SQL Monitor
- Active Jobs
- System Status
- Graph History



iDoctor Suite
- Job Watcher
- PEX Analyzer
- Heap Analyzer
- VIOS Investigator

PTDV

PM for Power Systems

WLE

Management Central

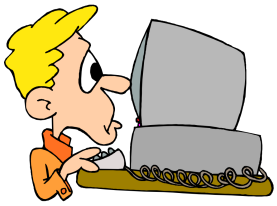
Third Party Tools

Collection Services

Job Watcher

Disk Watcher

Performance Explorer



Navigator Performance
- Investigate Data
- Batch Model
- System Monitors
- Graph History

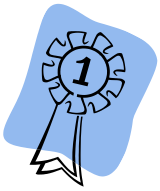
Navigator for i

Web Performance Advisor
Web Performance Monitor

Cognitive Systems

IBM

Performance Instrumentation and Data Collection



The i for Business Advantage

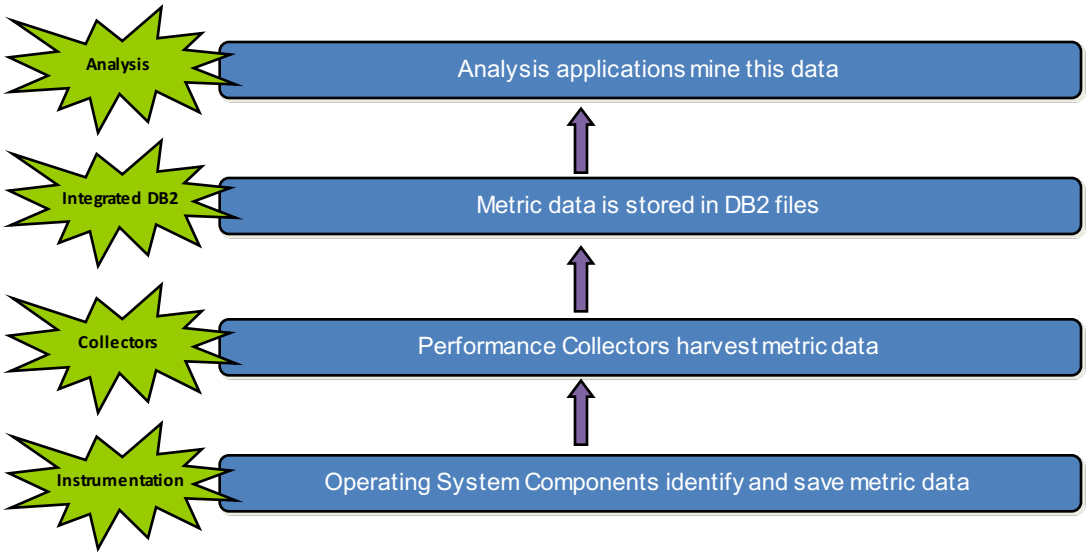
- IBM **develops the software stack**, top to bottom
  - Instruments the software with component-specific performance metrics
- IBM develops the **performance data collectors** that harvest those performance metrics
- IBM i has an **integrated database** – DB2
  - This is a BIG DEAL
  - Performance data is stored in the database automatically
    - No "add on" application is necessary – it's all in the Operating System
- IBM provides the **graphical analysis tools**
  - Analysis of the performance data in the DB2 files using SQL

IBM i has the best performance instrumentation and data collection capabilities in the industry!

Cognitive Systems

IBM

Performance Instrumentation and Data Collection



IBM

Cognitive Systems

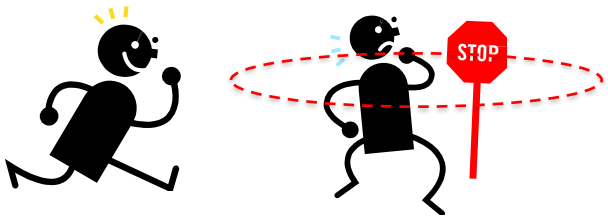
Introduction to Wait Accounting

Performance Fact

"All computers wait at the same speed"

What is Wait Accounting?

- Wait Accounting is the ability to determine what a job is doing when it is *not* running



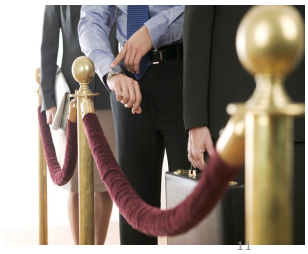
i Exclusive! Patented technology built into IBM i.

Cognitive Systems

IBM

Wait Accounting Overview

- When a job is not running (i.e., not using CPU), it is waiting
  - *But what is it waiting for?*
- Waits may be normal, some waits are not normal
  - Wait Accounting helps to determine what the wait is and if it is a problem
- IBM i has instrumented most of the key wait conditions
  - Wait information is automatically collected by **Collection Services** and **Job Watcher**



11

Cognitive Systems

IBM

Wait States

- Wait information is tracked for each job, thread and task on system
- A job/thread/task is in one of three states:

Using CPU

- "Dispatched CPU"
  Assigned to a virtual processor so it can begin execution of instructions

Waiting for CPU

- "CPU Queuing"
  Ready to use processor, but waiting for it to become available

Waiting for something else…

- Idle waits
- **Blocked waits**

These waits are typically the most interesting waits to focus on

12

Cognitive Systems

IBM

Types of Waits

- Idle waits
  - Typically a normal wait condition; for example:
    - Waiting for the "Enter" key to be pressed on a 5250 display session
    - Waiting for data from the network
- **Blocked waits**
  - Waits that occur while performing a work request
  - Blocked waits are are what we want to look at more closely

Cognitive Systems

IBM

Wait Accounting - Buckets

- **Wait Buckets** = "Wait condition groups" instrumented in the operating system.
  - **Buckets** can then be **analyzed** to determine where a job is spending it's time (running or waiting)
  - Categorized into **32** buckets
  - Buckets found in both **Collection Services** and **Job Watcher** data
  - Waits can be viewed at a **system-level** or at an **individual job/thread/task level**
    - Can also be grouped by generic job name, subsystem, user profile, pool ID, etc.

Cognitive Systems

IBM

32 Wait Buckets

1. Time dispatched on a CPU
2. CPU queuing
3. Reserved
4. Other waits
5. Disk page faults
6. Disk non-fault reads
7. Disk space usage contention
8. Disk operation start contention
9. Disk writes
10. Disk other
11. Journaling
12. Semaphore contention
13. Mutex contention
14. Machine level gate serialization
15. Seize contention
16. Database record lock contention
17. Object lock contention
18. Ineligible waits
19. Main storage pool contention
20. Classic Java™ user including locks (to 6.1)
    → **(7.2)** Journal save while active
21. Classic Java JVM (up to 6.1), now reserved
22. Classic Java other (up to 6.1), now reserved
23. Reserved
24. Socket transmits
25. Socket receives
26. Socket other
27. IFS
28. PASE
29. Data queue receives
30. Idle/waiting for work
31. Synchronization Token contention
32. Abnormal contention

http://www.ibm.com/developerworks/ibmi/library/i-ibmi-wait-accounting/
http://public.dhe.ibm.com/services/us/igsc/idoctor/Job_Waits_White_Paper_61_71.pdf

Cognitive Systems

IBM

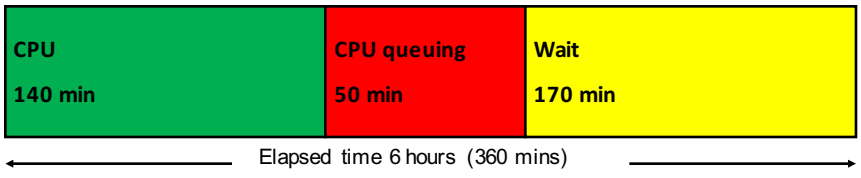
Wait Accounting – "Run-wait" signature

Applying the concepts of wait accounting, we are now able to identify the amount of time the thread/task was running and the time the thread/task was waiting.

Consider the following:

Batch job with total run time of 6 hours

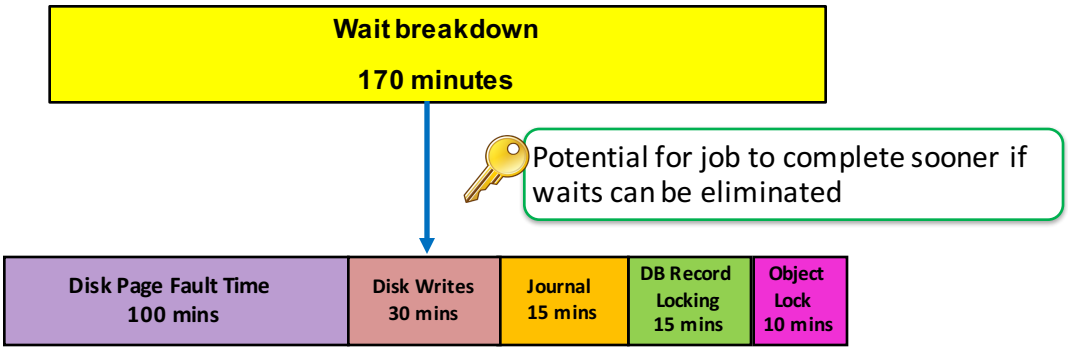
Run-wait signature



Cognitive Systems

IBM

Wait Accounting – "Run-wait" signature



Now you can start asking questions such as:

- Are my pool sizes appropriate? What objects is the faulting occurring on?
- Is the write cache being overrun? Is the application forcing writes out synchronously?
- Are all the journals optimally configured? Are unnecessary objects being journaled?
- Am I locking records or objects unnecessarily?

Cognitive Systems

IBM

Metrics related to components of wait time

	Disk reads	Disk writes	Record Locks	Journal
Total count	3,523	17,772	355	5,741
Total time	42 sec	73 sec	45 sec	44 sec
Avg time per wait	0.012 sec	0.004 sec	0.126 sec	0.007 sec

- Tools capture this level detail
- Useful to know both counts and time

Disk Page Faults Counts	Disk Page Faults Time (Seconds)
20498	26.37
7969	10.83
9080	15.86
1738	2.55
3124	1.22
1071	0.65
2013	0.94
3133	1.19
2939	1.36
1056	0.55

Cognitive Systems

IBM

Why developers should leverage wait accounting!!

- Helps you understand application characteristics
  - Is it CPU bound? I/O bound?
- Helps you to understand where to focus your effort and investment
  - Is there a bottleneck on CPU, Memory, I/O, Contention time?
  - Invest resources where greatest benefit will be
- Can offer insight into potential performance issues before end-users are affected
  - Can leverage aspects of wait accounting in test environment
    - Eliminate surprises
    - Identify bottlenecks that prevent scaling
- Provides valuable clues to help analyze performance issues as they arise
- Instrumentation part of base IBM i operating system, IBM tools available to help you analyze

Cognitive Systems

IBM

Common Waits that Applications use

- Disk Waits
- Semaphores, Mutexes, Synchronization Tokens
- Journaling
- Database record locks
- Object locks
- Sockets

Cognitive Systems

IBM

A few other things to know about waits…

- Some waits are "expected" and others "unexpected"
  - A record lock may be expected, but one that lasts for a very long duration is unexpected
- If waits can be reduced or eliminated, CPU can be used more efficiently
- One wait may be reduced/eliminated, only to have another wait surface
- Likely won't be able to remove all wait times
- When is a wait "bad"?
  - Is there a business impact? Are users complaining?
  - "**It depends**" but **waits more than 25% of run time** may need additional analysis

Cognitive Systems

IBM

Tools for analyzing Wait Accounting information

Cognitive Systems

IBM

Wait Accounting - IBM i Performance Data Collectors

- **Collection Services**
  - Collects data automatically 24 X 7 at specified intervals (typically 5 or 15 minutes)
  - System and job level data
  - Starting point!

- **Job Watcher**
  - Needs to be started/stopped (typically 5 or 10 second intervals)
  - Additional detailed data such as call stacks, object waited on, holder
  - Frequently needed to solve performance issues

Cognitive Systems

IBM

Performance Data

Collection vs Visualization

- The performance data ***collection*** capability is built into the operating system
  - **Everyone** can collect all types of IBM i performance data

- The ***visualization*** of the data may require additional products

  - Visualization is the display of charts or tables of performance data

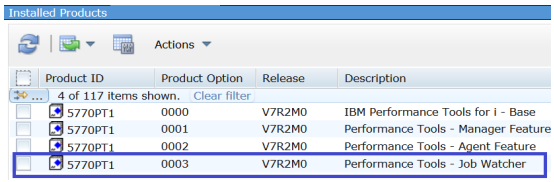
  - The Performance Data Investigator is part of the operating system
    - **Everyone** can visualize Collection Services data

Cognitive Systems

IBM

Two Graphical Analysis Tools

- Performance Data Investigator – Job Watcher
  - Requires 57xx-PT1 – Job Watcher feature
  - Geared to *average* user



- iDoctor – Job Watcher
  - IBM i Service offering, yearly license by serial number
  - Geared to *advanced* user

Cognitive Systems

IBM

Wait Accounting IBM i Graphical Analysis Tools

- Two powerful graphical tools to help make your analysis more efficient and productive:

Performance Data Investigator (PDI)

- Component in IBM Navigator for i (browser-based)
- Nothing to install, can view Collection Services for free
- http://www.ibm.com/developerworks/ibmi/library/i-pdi/index.html

IBM iDoctor for IBM i

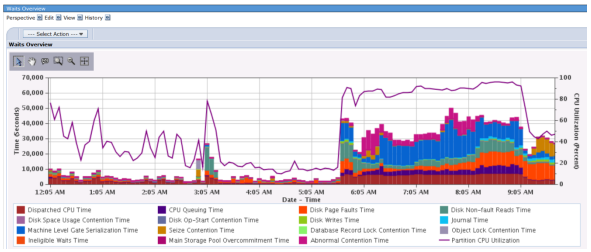
- Microsoft Windows based client
- Requires Job Watcher yearly license to see Collection Services data (IBM Service offering)
- https://www-912.ibm.com/i_dir/idoctor.nsf

Cognitive Systems

IBM

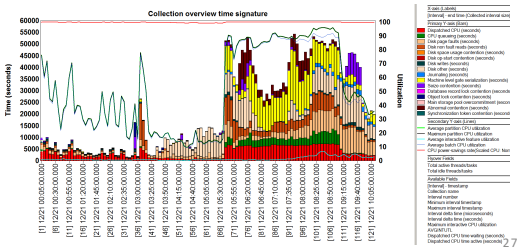
Wait Accounting IBM i Graphical Analysis Tools

- Both graphical tools sit on top of same rich IBM i instrumentation, but not equivalent in presentation and function



← PDI

iDoctor ->



Cognitive Systems

IBM

Wait Accounting Analysis Strategy

- Understand the "big picture" first
  - Understand overall partition characteristics first and where system bottlenecks may be that affecting your application
  - Typically done using **Collection Services** data
  - Drill down to job level
  - Waits can be analyzed in various useful ways:

- Waits by Job or Task
- Waits by Generic Job or Task
- Waits by Job User Profile
- Waits by Job Current User Profile
- Waits by Pool
- Waits by Subsystem
- Waits by Server Type
- Waits by Job Priority

- Continue detailed analysis at a Job Level using **Job Watcher**
  - Narrow focus to interesting timeframes / jobs
  - Many more job level details available

Cognitive Systems

IBM

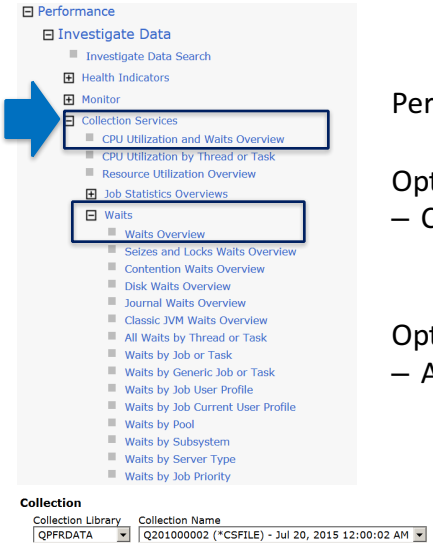
Using Performance Data Investigator (PDI)

- IBM Navigator for i is the Web console for managing IBM i
  - Has much of the function as System i Navigator
    - but with a browser user interface
  - Simply point your browser to http://systemname:2001

Cognitive Systems

IBM

PDI Wait Accounting Perspectives - Where to start



Performance -> Investigate Data -> Collection Services:

Option 1: **CPU Utilization and Waits Overview**
– Combines related waits into higher level buckets

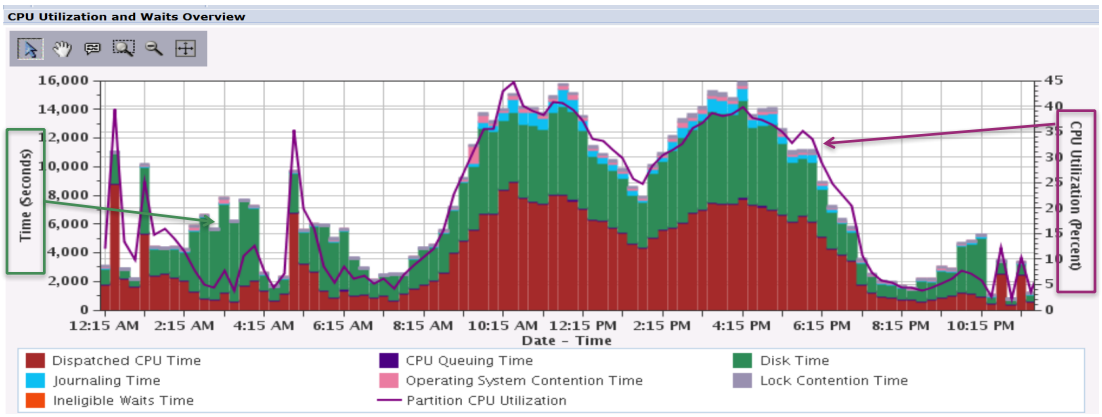
Option 2: **Waits Overview**
– All individual "blocked" wait buckets shown

Cognitive Systems

IBM

System "run-wait" signature

CPU Utilization and Waits Overview



- A chart that combines CPU utilization as well as the wait buckets can be very beneficial in assessing the health of your partition
- In this chart, we can see that the majority of the time, the jobs were spending time in CPU as well as in Disk. Minor amounts of Journal wait time and operating system contention time are also present.

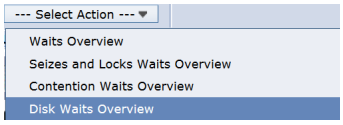
31

Cognitive Systems

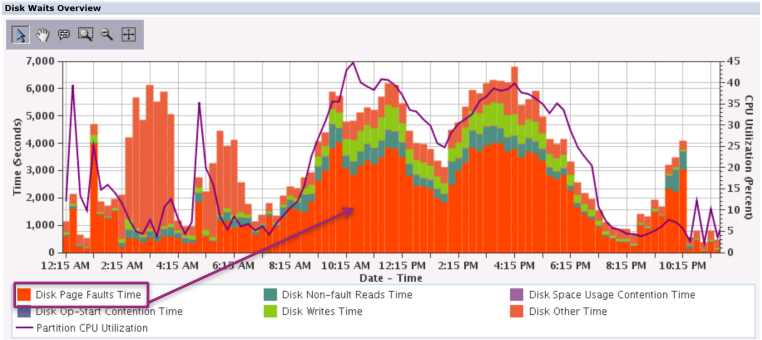
IBM

CPU Utilization and Waits Overview

Drilldown analysis



Because Disk wait time was fairly significant, drilldown to **Disk Waits Over view** to further examine the detailed waits contributing to this time:



We can see that Disk Page Fault time is the biggest contributor to Disk Time.
(A job needed something in memory, it wasn't there, had to do an I/O to bring it into memory before job could continue running).

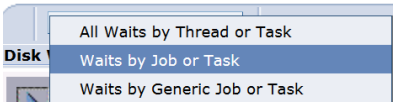
32

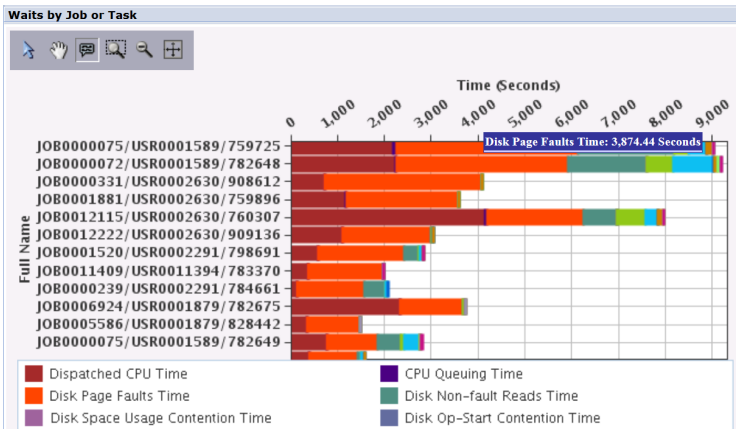
Cognitive Systems

IBM

Waits by Job or Task

The next question likely would be which job(s) are incurring this wait time. Drilling down further, we can see the list of jobs incurring this wait time:



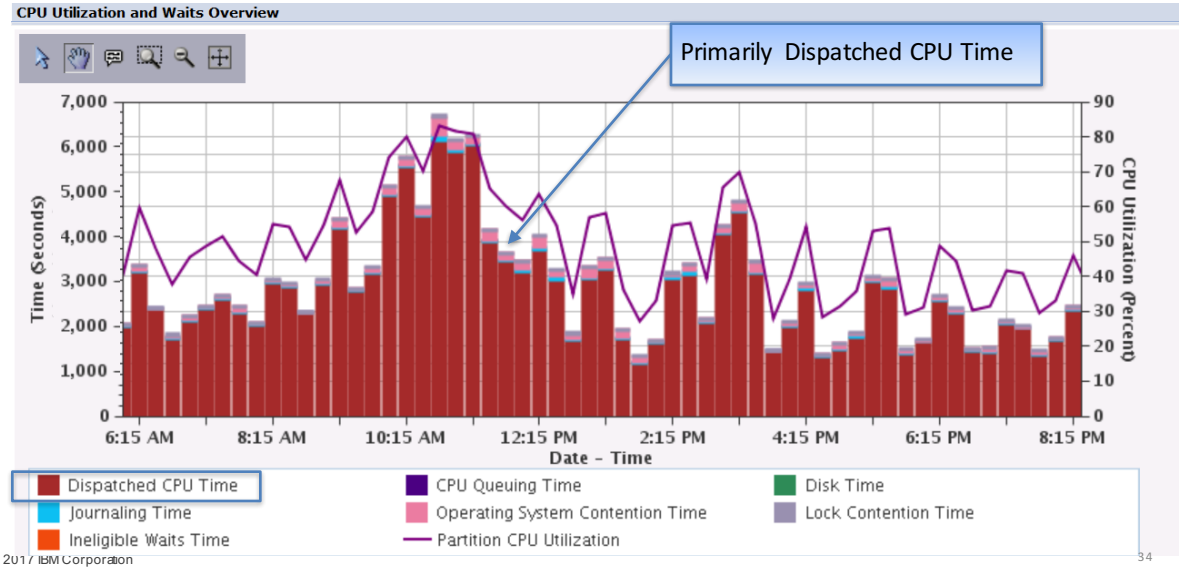


This type of chart can also be used to understand a job(s) "run-wait" signature.

Cognitive Systems

IBM

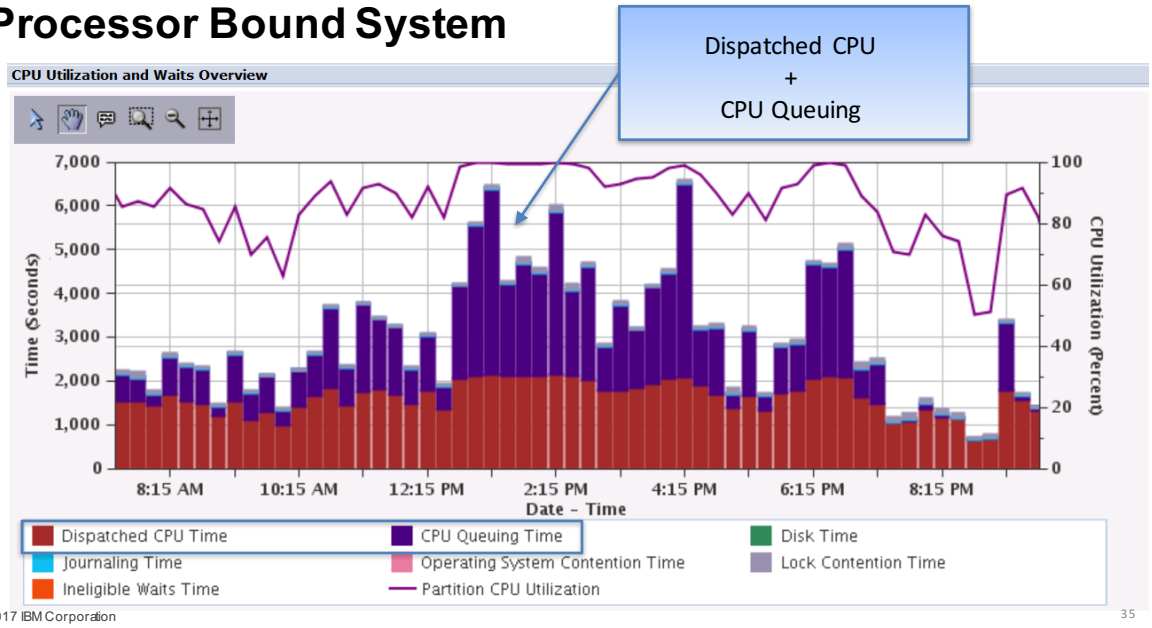
Efficient System with Little Waiting



Cognitive Systems

IBM

Processor Bound System

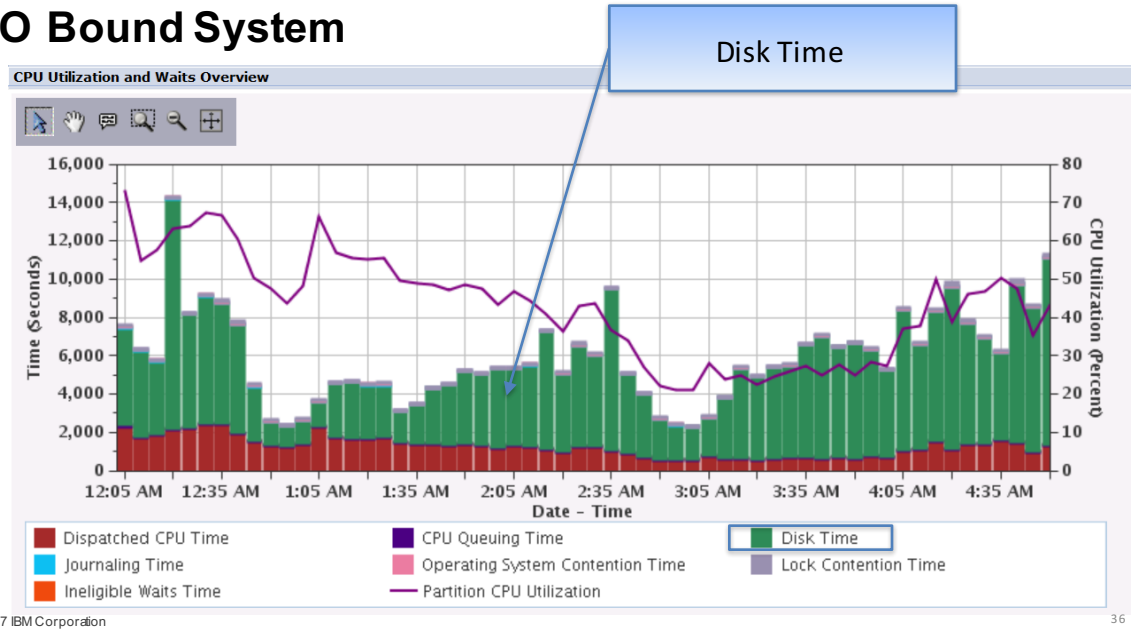


35

Cognitive Systems

IBM

I/O Bound System

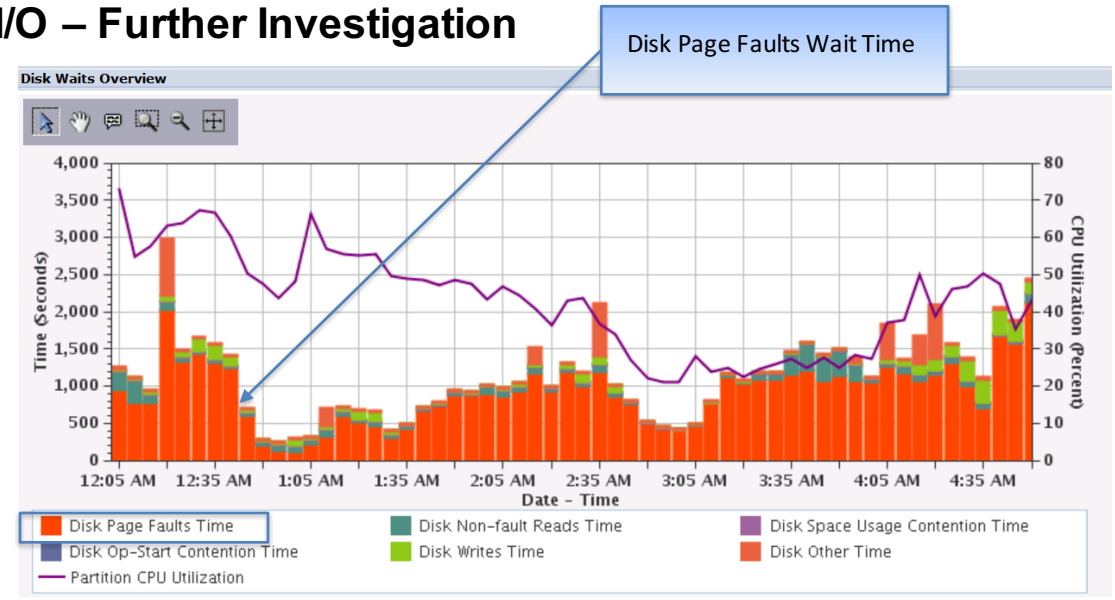


36

Cognitive Systems

IBM

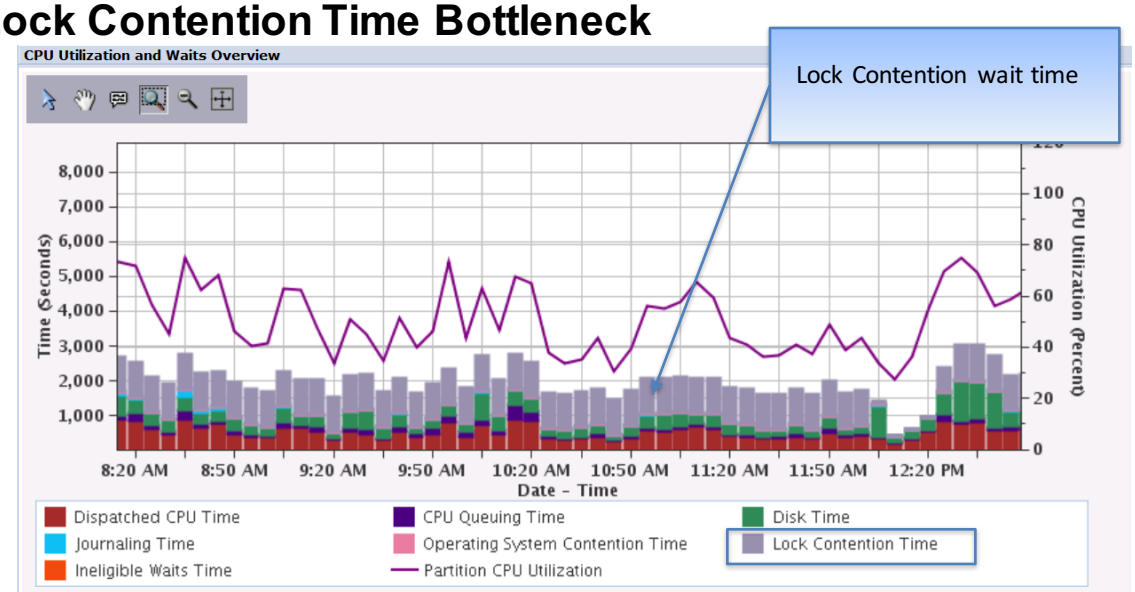
I/O – Further Investigation



Cognitive Systems

IBM

Lock Contention Time Bottleneck

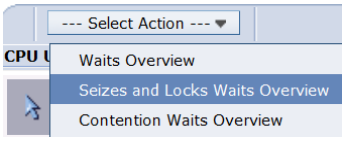


Job Watcher data is typically needed to solve lock related issues.

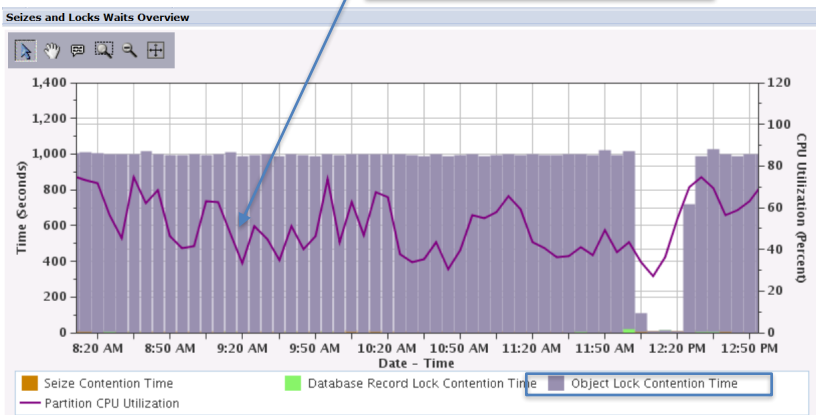
Cognitive Systems

IBM

Lock – Further Investigation



Object Lock Contention time



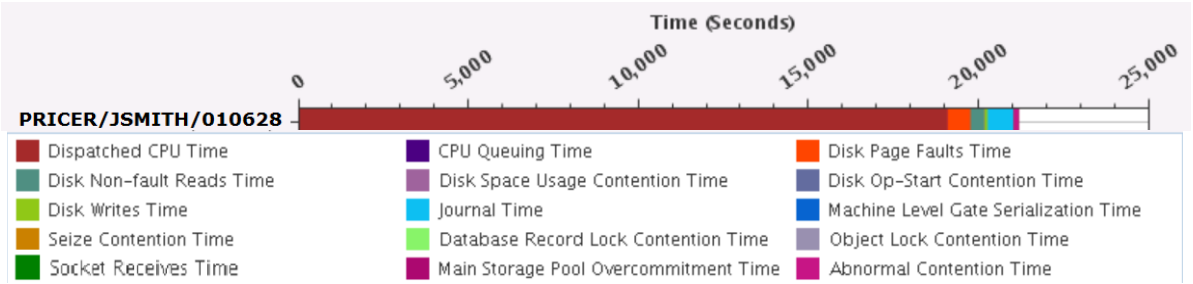
Job Watcher data will show object waited on, the holder, and call stacks for both the waiter and the holder (example shown later on…)

Cognitive Systems

IBM

Wait Accounting at a Job Level

Would this job benefit from additional memory?

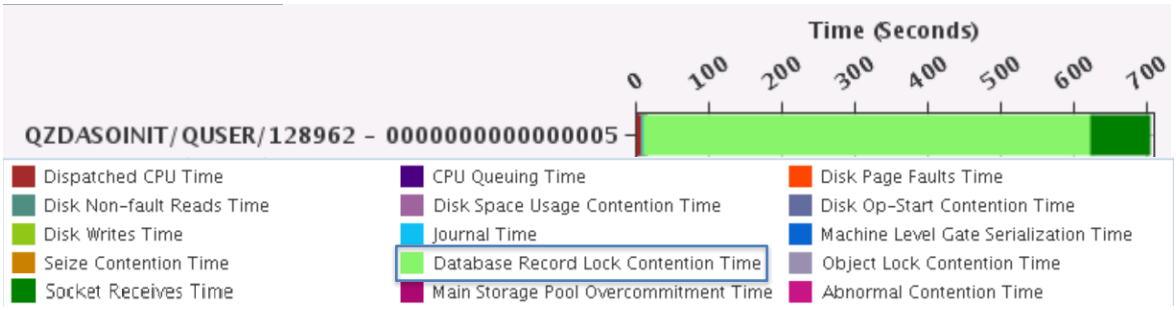


Cognitive Systems

IBM

Wait Accounting at a Job Level

Would this job benefit from additional memory? CPU? Disk?

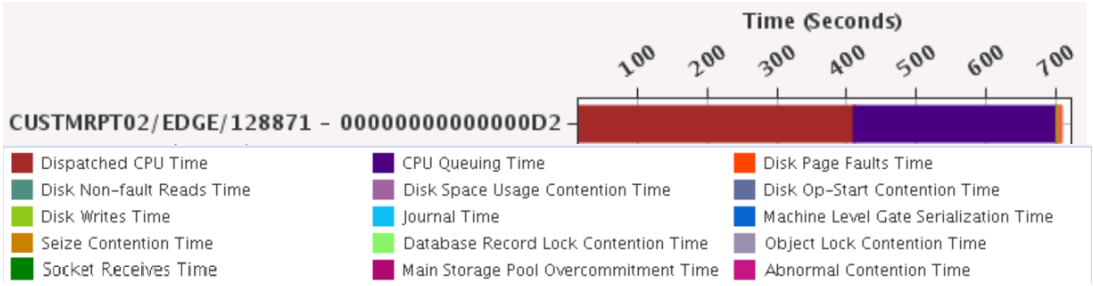


Cognitive Systems

IBM

Wait Accounting at a Job Level

Would this job benefit from an improved I/O subsystem?



Cognitive Systems

IBM

Job Watcher - Additional Benefits

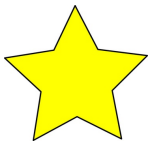
- Collects **more detailed** performance data than Collection Services
  - Call Stacks
  - SQL Statements
  - Additional wait accounting information:
    - Objects being waited on
    - Holder of object
- **More frequent** intervals (seconds)
- Need to start/stop Job Watcher
  - Navigator for i, iDoctor, green screen commands
- To see charts in PDI, need Performance Tools LPP Job Watcher option (chargeable) or iDoctor Job Watcher license for viewing in iDoctor

Cognitive Systems

IBM

Job Watcher - Holders versus Waiters

- IBM i keeps track of who is holding a resource, and if applicable, who is waiting to access that resource
  - A **Holder** is the job/thread/task that is holding the serialized resource
  - A **Waiter** is the job/thread/task that wants to access the serialized resource
- IBM i also maintains call stacks for every job/thread/task



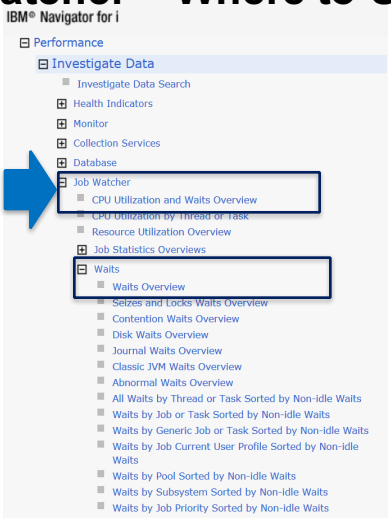
- The combination of
  - **Who** - holders and waiters … *who has it? who wants it?*
  - **What** – object being waited on
  - **How** - call stacks

  provides a very powerful solution for analyzing wait conditions

Cognitive Systems

IBM

Job Watcher – Where to Start



Performance -> Investigate Data -> Job Watcher:

Option 1: **CPU Utilization and Waits Overview**
– Combines related waits into higher level buckets

Option 2: **Waits Overview**
– All individual "blocked" wait buckets shown

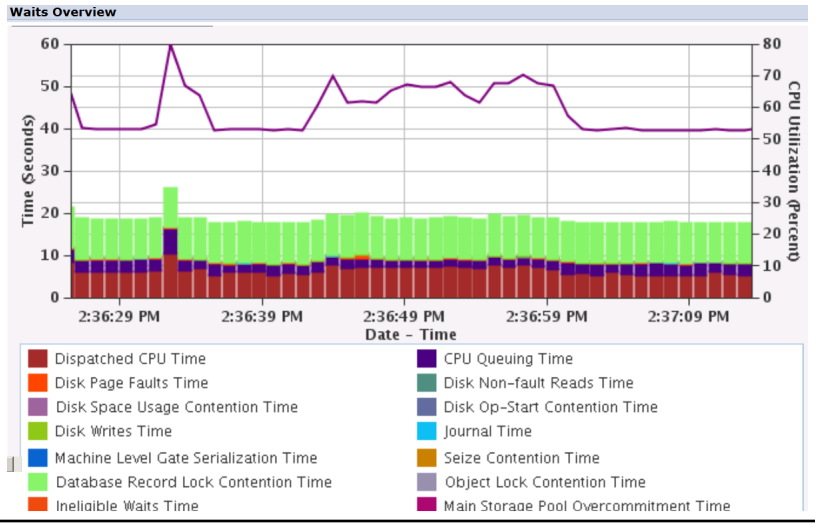
Notice similar perspectives available as Collection Services

Cognitive Systems

IBM

Job Watcher – Waits Overview

Notice same wait buckets, but more granular intervals



Cognitive Systems

IBM

Job Watcher – Additional Interval Details

Thread or Task Details

Job information:	QZDASOINIT/QUSER/128962 - 0000000000000005	Priority	20
Current user profile:	LISAW	Pool:	2
Object waited on:	INVENTORY INVENTORY	Type description:	PHYSCIAL FILE MBR - DATA PART
Wait duration:	581 milliseconds	Segment type description:	DB PHYSICAL FILE MEMBER RECORDS
Current or last wait:	DB record lock: update	Wait object library:	None detected this interval
Holding job or task:	QZDASOINIT/QUSER/128890	Interval timestamp:	Jan 3, 2014 2:36:28 PM
SQL client job:	None detected this interval	Interval (1 to 684):	< 174 >

Show Holder

Call Stack

--- Select Action ---

Call Level	Program	Module	Procedure
1			qutde_block_tra
2			longWaitReceive__9QuCounterFR12RmprReceiverPv
3			DBLockConflict__15RmslDBHashClassFR11RmslPlm
4			rmslDBHLock__FR11RmslPlmpLad
5			getLockWithWait__18DbpmUpdateResourcede
6			getLock__18DbpmUpdateResourcead
7			getRowLock__18DbpmUpdateResourceFCUlRC9Dbp
8			execute__18DbpmUpdateLockNodeFR13DbpmQuer
9			vPositionNextAndExecute__18DbpmUpdateLockNod
10			positionNextEntryAndFetchOutline__17DbpmReadO

SQL Statement

☑ Include Host Variables

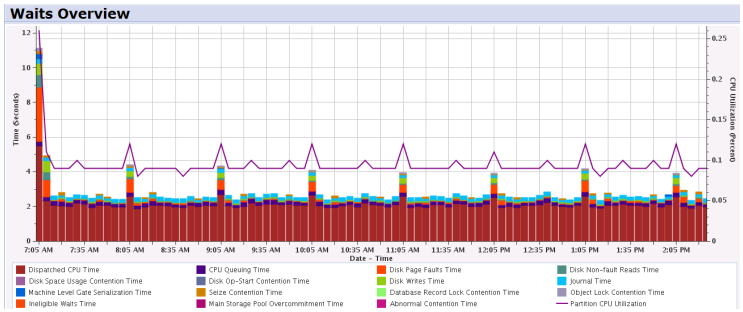
SELECT QUANTITY FROM WAREHSE42.INVENTORY WHERE ID=*DATA FORMAT ERRORTITY FROM WAREHSE42.INVENTORY WHERE ID=? FOR UPDATE

Cognitive Systems

IBM

Wait Accounting: Be proactive!

- Use the rich IBM i wait accounting instrumentation found in
  - Collection Services & Job Watcher
  - Use PDI or iDoctor to view/analyze
- Understand your partition's "run-wait" signature and normal patterns
- Identify bottlenecks

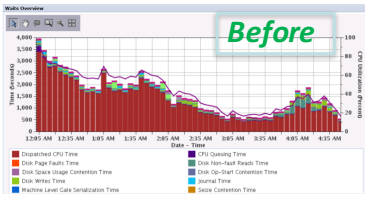


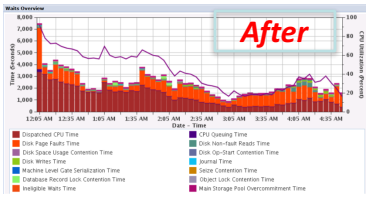
Cognitive Systems

IBM

Recommendations: Be proactive!

- Keep a **baseline**
  - Collection Services (Job Watcher data is also nice to have)
    - Weekly, end-of-month, end-of-year
    - Prior to any hardware, software, configuration related change
- A baseline provides a **reference point**
  - It is the expected performance characteristics over a defined period of time
  - Having one makes it easier to recognize changes and its effect





- **Wait bucket information can make it easier to determine what has changed! Both at a partition level as well as an individual job level**

IBM

Cognitive Systems

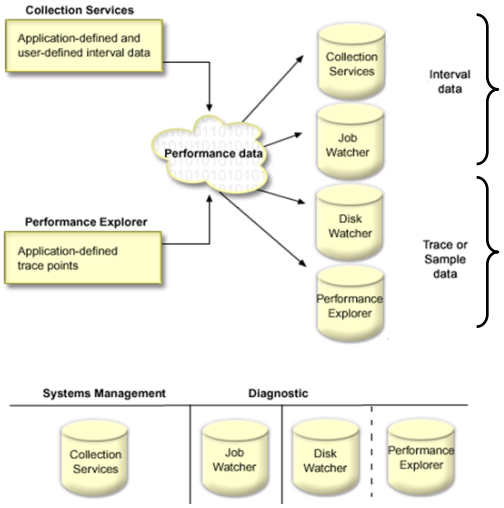
IBM i Performance Data Collectors

Cognitive Systems

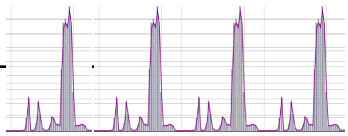
IBM

Performance Data Collection Architecture

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



Cognitive Systems



Collect System-wide Performance Data

Collect Performance Data 24/7

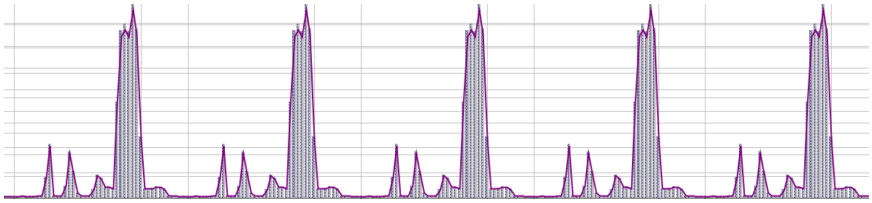
If something goes wrong, you have data that will help **analyze** the problem, **fix** it, and **prevent** it from happening in the future

- If you can't solve the problem, you have information that makes it easier for IBM Support to **solve the problem faster**
- To provide a **reliable baseline** so you can **understand the impact** that a software, network, or environmental **change** had on the performance of your system
- To provide historical information that enables you to **plan for future growth** based on real trends, not guesses.

Cognitive Systems

IBM

Patterns in Performance Data



- Performance data typically has patterns
  - Daily, weekly, monthly, yearly

Understand your typical patterns

  - Recognize change

IBM

Cognitive Systems

Job Watcher

Cognitive Systems

IBM

Job Watcher

- Job Watcher returns real-time information about a selected set of jobs, threads, or LIC tasks
- Job Watcher collects additional types of data that Collection Services does not, as well as more frequent intervals
  - Job Watcher has more overhead than Collection Services
- Data collected by Job Watcher includes
  - Wait times
  - CPU
  - I/O activity
  - Call Stacks
  - SQL statements
  - Communications statistics
  - Activation Group statistics

Run Job Watcher when you need detailed performance data for diagnostic purposes.

There are clients that run Job Watcher 24x7 to always have diagnostic data available.

Need to manage the data carefully.

Cognitive Systems

IBM

Job Watcher

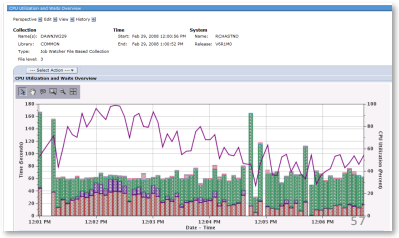
- Job Watcher collects **more detailed** performance data than Collection Services and at **more frequent intervals**
  - CPU and I/O (like Collection Services)
  - Call Stacks
  - SQL Statements
  - Detailed Wait information:
    - **Objects being waited on**, even records number of files
    - **Holder** of object
- Job Watcher does not collect everything that Collection Services collects.
  - It does not always collect information about every thread
    - Thread must use CPU during interval
    - Thread must exist for entire interval
  - It does not collect memory pool or detailed I/O statistics
- Data is written to DB2 files

Cognitive Systems

IBM

Job Watcher Usage Tips

- Use Job Watcher when you need detailed performance data to **resolve a problem**
  - Typically problem has been scoped first by Collection Services
  - Consider using a definition like the IBM-supplied Q10SECSQL (10 second intervals, gathers Call Stacks and SQL if it can) as a starting point
- For problem determination Job Watcher can be run on **specific jobs**
  - **Caution:** When using Job Watcher on specific job(s), you may not get detailed Holder information
- Multiple collections can be run at the same time
- Need to manage the amount of data collected
  - Specify Maximum Disk Space to consume on Start
  - Will stop collecting if specified ASP threshold is reached
  - iDoctor Monitors will clean up after themselves



Cognitive Systems

IBM

Basic Job Watcher Data Collection Steps

- Select an an IBM-supplied **definition** to use
  - Or create your own custom definition
- **Start** the Job Watcher collection
- Let it **run** until the problem has occurred
- **Stop** the Job Watcher collection
- **Analyze** the data
- There may be times when you want to run Job Watcher for long periods of time
  - Consider 1 hour collection size
  - Manage size of data

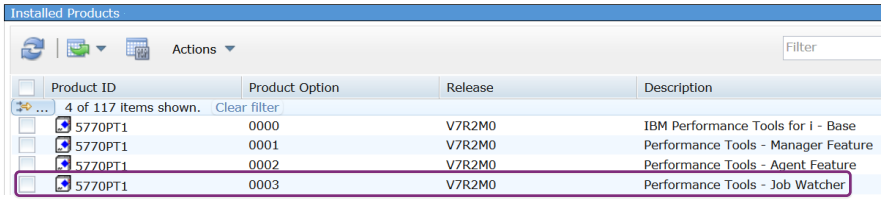
58

Cognitive Systems

IBM

Job Watcher in Navigator for i

- Navigator for i Web console provides a GUI for Job Watcher
  - Requires the Performance Tools LPP, Job Watcher option to be installed

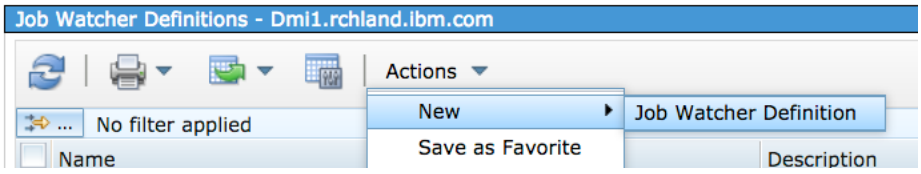


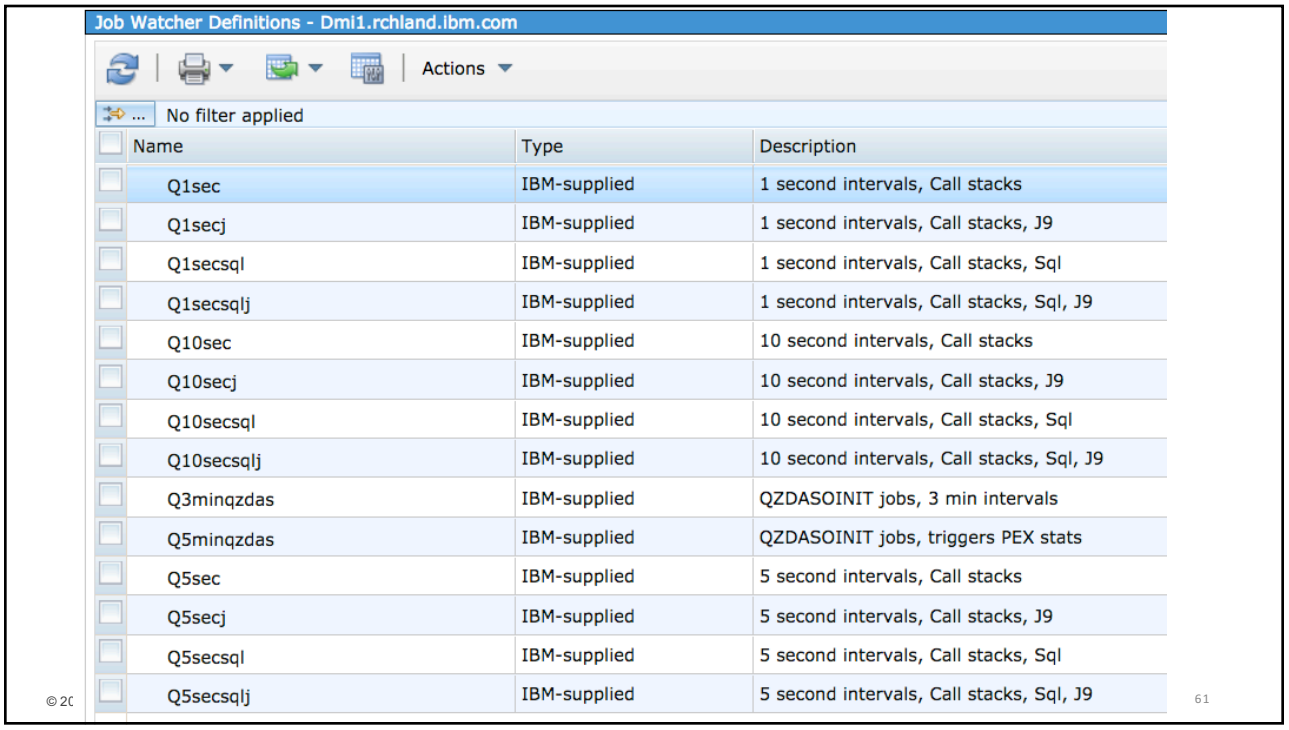
Cognitive Systems

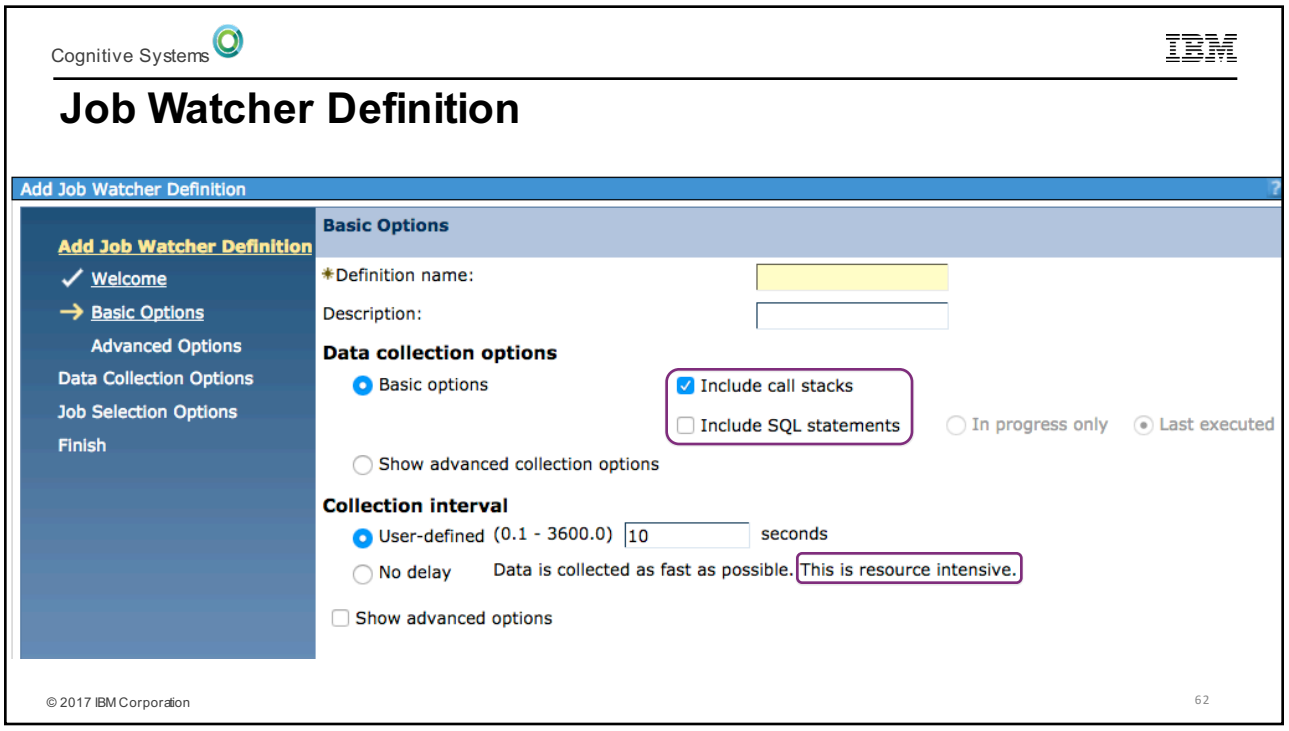
IBM

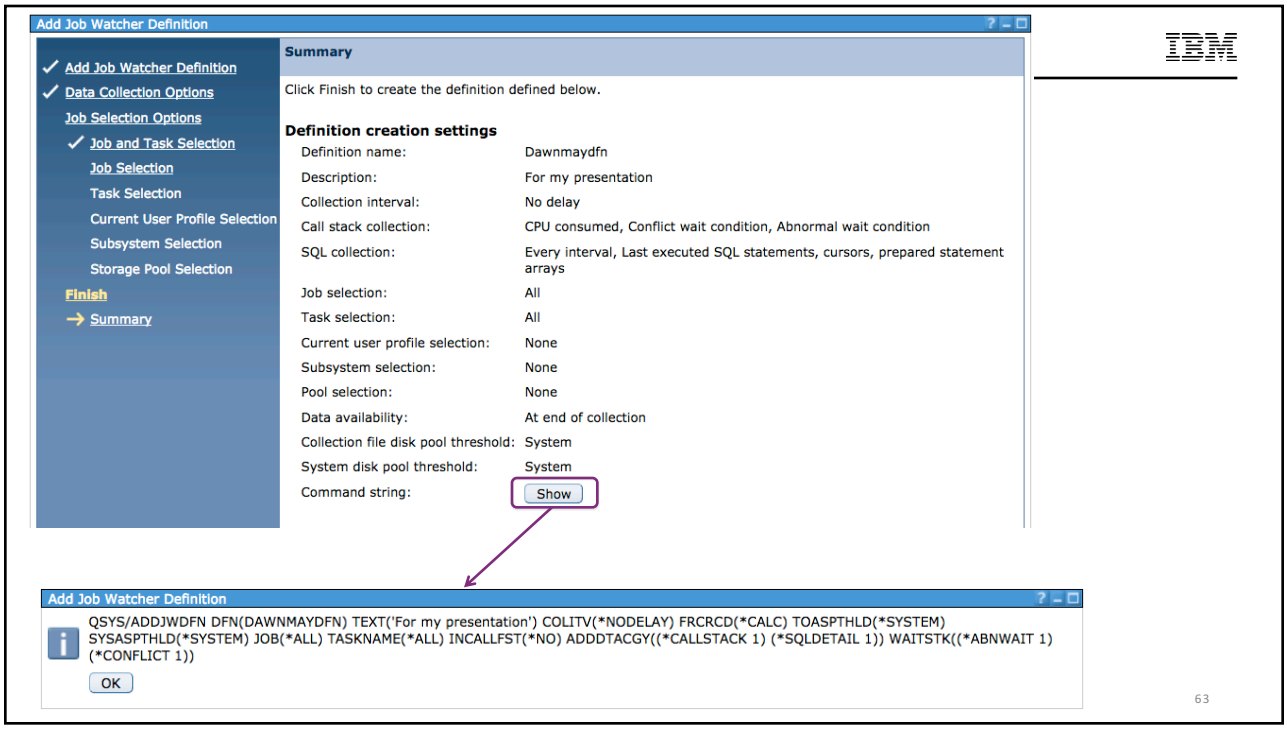
IBM-Supplied Job Watcher Definitions

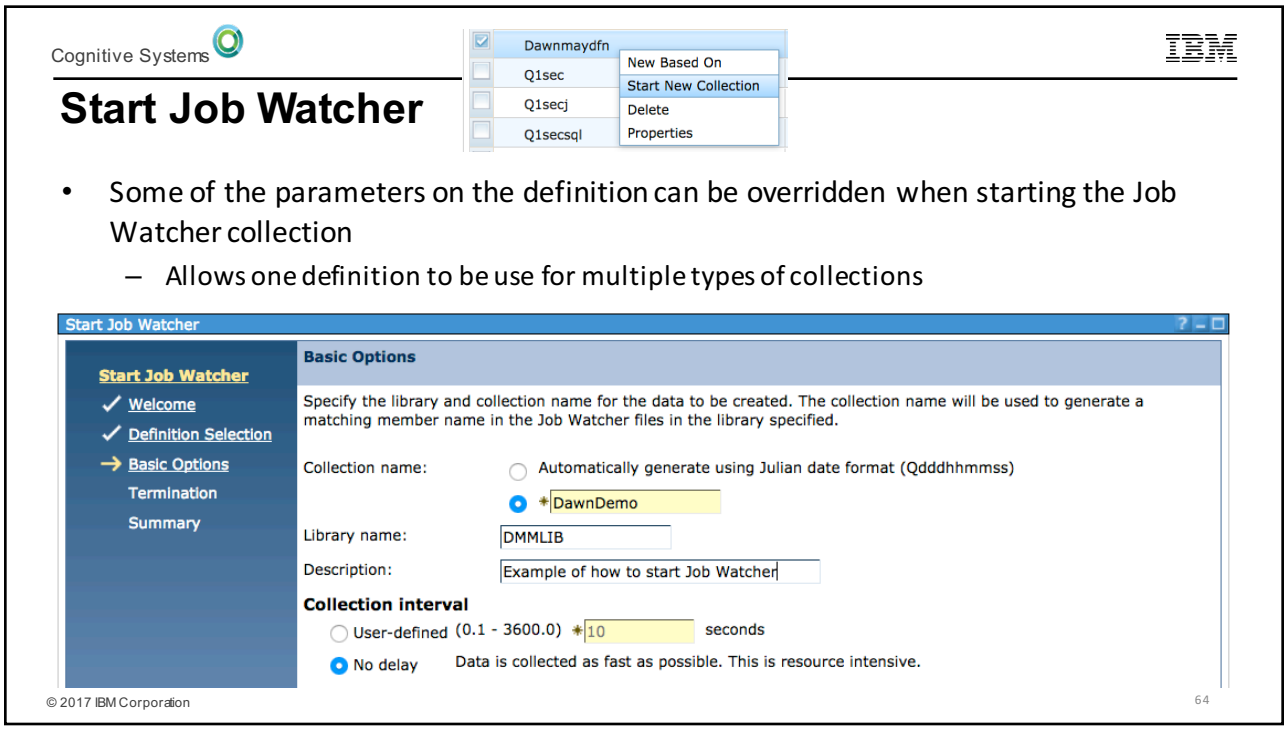
- Many pre-defined Job Watcher definitions are available
  - The main difference is the sample interval
- Recommendations:
  - Collect with Call Stacks and SQL
  - Use 10 second intervals for general analysis (Q10secsql)
  - Use 5 second intervals for complex or intermittent issues, or for contention related problems (Q5secsql)

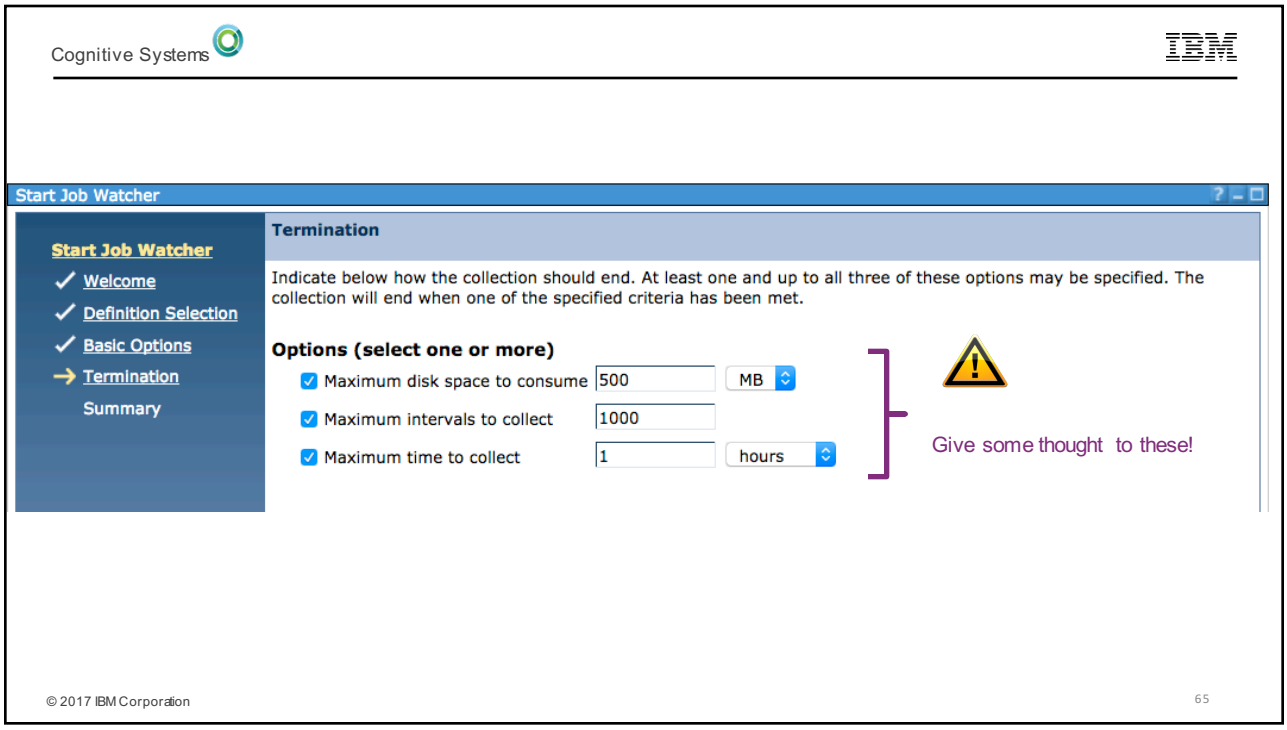


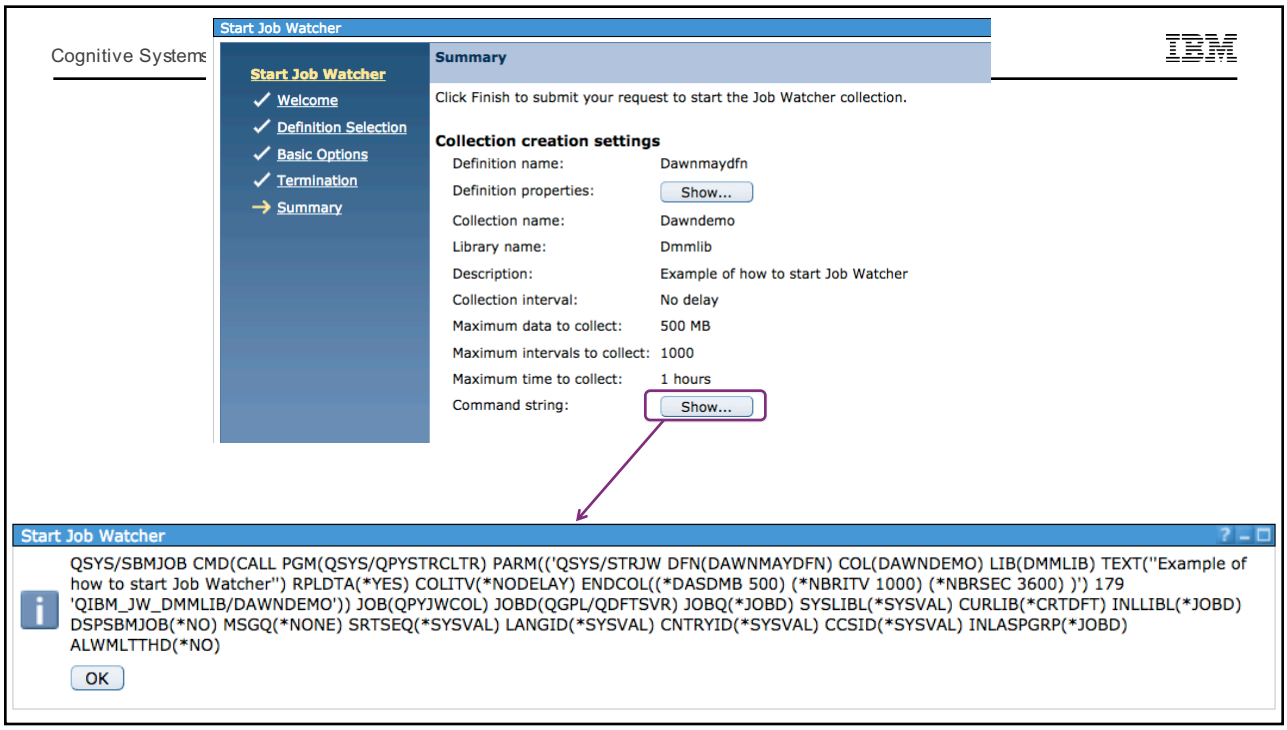

Job Watcher Definitions - Dmi1.rchland.ibm.com
Actions
No filter applied
Name
Type
Description
Q1sec
IBM-supplied
1 second intervals, Call stacks
Q1secj
IBM-supplied
1 second intervals, Call stacks, J9
Q1secsql
IBM-supplied
1 second intervals, Call stacks, Sql
Q1secsqlj
IBM-supplied
1 second intervals, Call stacks, Sql, J9
Q10sec
IBM-supplied
10 second intervals, Call stacks
Q10secj
IBM-supplied
10 second intervals, Call stacks, J9
Q10secsql
IBM-supplied
10 second intervals, Call stacks, Sql
Q10secsqlj
IBM-supplied
10 second intervals, Call stacks, Sql, J9
Q3minqzdas
IBM-supplied
QZDASOINIT jobs, 3 min intervals
Q5minqzdas
IBM-supplied
QZDASOINIT jobs, triggers PEX stats
Q5sec
IBM-supplied
5 second intervals, Call stacks
Q5secj
IBM-supplied
5 second intervals, Call stacks, J9
Q5secsql
IBM-supplied
5 second intervals, Call stacks, Sql
Q5secsqlj
IBM-supplied
5 second intervals, Call stacks, Sql, J9
© 2(
61


Cognitive Systems
IBM
Job Watcher Definition
Add Job Watcher Definition
Add Job Watcher Definition
Welcome
Basic Options
Advanced Options
Data Collection Options
Job Selection Options
Finish
Basic Options
*Definition name:
Description:
Data collection options
Basic options
Include call stacks
Include SQL statements
In progress only
Last executed
Show advanced collection options
Collection interval
User-defined (0.1 - 3600.0) 10 seconds
No delay
Data is collected as fast as possible. This is resource intensive.
Show advanced options
© 2017 IBM Corporation
62


Add Job Watcher Definition
Add Job Watcher Definition
Data Collection Options
Job Selection Options
Job and Task Selection
Job Selection
Task Selection
Current User Profile Selection
Subsystem Selection
Storage Pool Selection
Finish
Summary
Summary
Click Finish to create the definition defined below.
Definition creation settings
Definition name: Dawnmaydfn
Description: For my presentation
Collection interval: No delay
Call stack collection: CPU consumed, Conflict wait condition, Abnormal wait condition
SQL collection: Every interval, Last executed SQL statements, cursors, prepared statement arrays
Job selection: All
Task selection: All
Current user profile selection: None
Subsystem selection: None
Pool selection: None
Data availability: At end of collection
Collection file disk pool threshold: System
System disk pool threshold: System
Command string: Show
Add Job Watcher Definition
QSYS/ADDJWDFN DFN(DAWNMAYDFN) TEXT('For my presentation') COLITV(*NODELAY) FRCRCD(*CALC) TOASPTHLD(*SYSTEM) SYSASPTHLD(*SYSTEM) JOB(*ALL) TASKNAME(*ALL) INCALLFST(*NO) ADDDTACGY((*CALLSTACK 1) (*SQLDETAIL 1)) WAITSTK((*ABNWAIT 1) (*CONFLICT 1))
OK
63


Cognitive Systems
Start Job Watcher
Dawnmaydfn
Q1sec
Q1secj
Q1secsql
New Based On
Start New Collection
Delete
Properties
• Some of the parameters on the definition can be overridden when starting the Job Watcher collection
– Allows one definition to be use for multiple types of collections
Start Job Watcher
Start Job Watcher
Welcome
Definition Selection
Basic Options
Termination
Summary
Basic Options
Specify the library and collection name for the data to be created. The collection name will be used to generate a matching member name in the Job Watcher files in the library specified.
Collection name: Automatically generate using Julian date format (Qdddhhmmss)
DawnDemo
Library name: DMMLIB
Description: Example of how to start Job Watcher
Collection interval
User-defined (0.1 - 3600.0) 10 seconds
No delay Data is collected as fast as possible. This is resource intensive.
© 2017 IBM Corporation
64

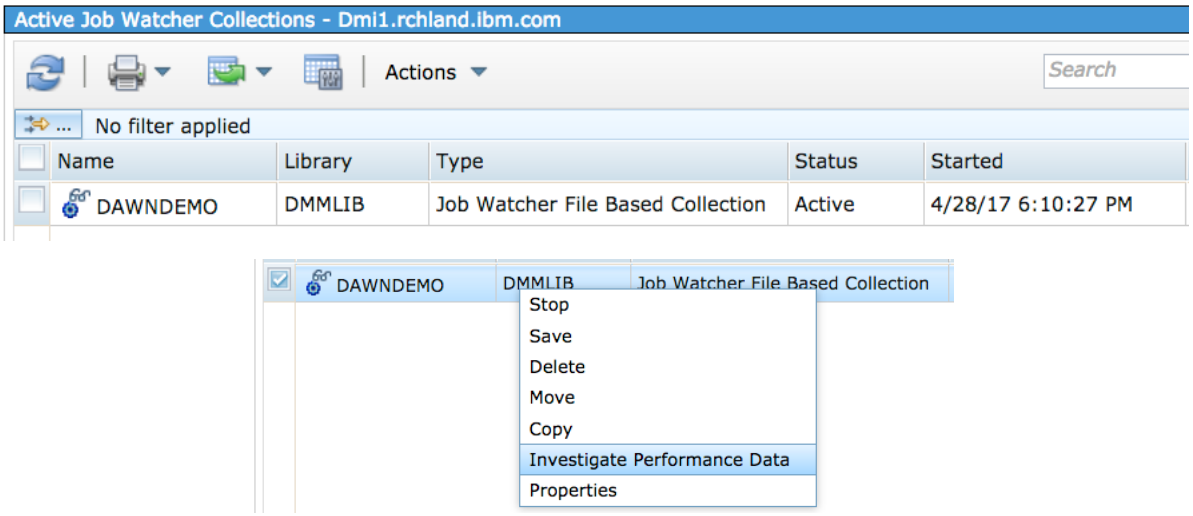

Cognitive Systems
IBM
Start Job Watcher
Start Job Watcher
Welcome
Definition Selection
Basic Options
Termination
Summary
Termination
Indicate below how the collection should end. At least one and up to all three of these options may be specified. The collection will end when one of the specified criteria has been met.
Options (select one or more)
Maximum disk space to consume 500 MB
Maximum intervals to collect 1000
Maximum time to collect 1 hours
Give some thought to these!
© 2017 IBM Corporation
65


Cognitive Systems
IBM
Start Job Watcher
Start Job Watcher
Welcome
Definition Selection
Basic Options
Termination
Summary
Summary
Click Finish to submit your request to start the Job Watcher collection.
Collection creation settings
Definition name: Dawnmaydfn
Definition properties: Show...
Collection name: Dawndemo
Library name: Dmmlib
Description: Example of how to start Job Watcher
Collection interval: No delay
Maximum data to collect: 500 MB
Maximum intervals to collect: 1000
Maximum time to collect: 1 hours
Command string: Show...
Start Job Watcher
QSYS/SBMJOB CMD(CALL PGM(QSYS/QPYSTRCLTR) PARM(('QSYS/STRJW DFN(DAWNMAYDFN) COL(DAWNDEMO) LIB(DMMLIB) TEXT(''Example of how to start Job Watcher'') RPLDTA(*YES) COLITV(*NODELAY) ENDCOL((*DASDMB 500) (*NBRITV 1000) (*NBRSEC 3600))') 179 'QIBM_JW_DMMLIB/DAWNDEMO')) JOB(QPYJWCOL) JOBD(QGPL/QDFTSVR) JOBQ(*JOBD) SYSLIBL(*SYSVAL) CURLIB(*CRTDFT) INLLIBL(*JOBD) DSPSBMJOB(*NO) MSGQ(*NONE) SRTSEQ(*SYSVAL) LANGID(*SYSVAL) CNTRYID(*SYSVAL) CCSID(*SYSVAL) INLASPGRP(*JOBD) ALWMLTTHD(*NO)
OK

Cognitive Systems

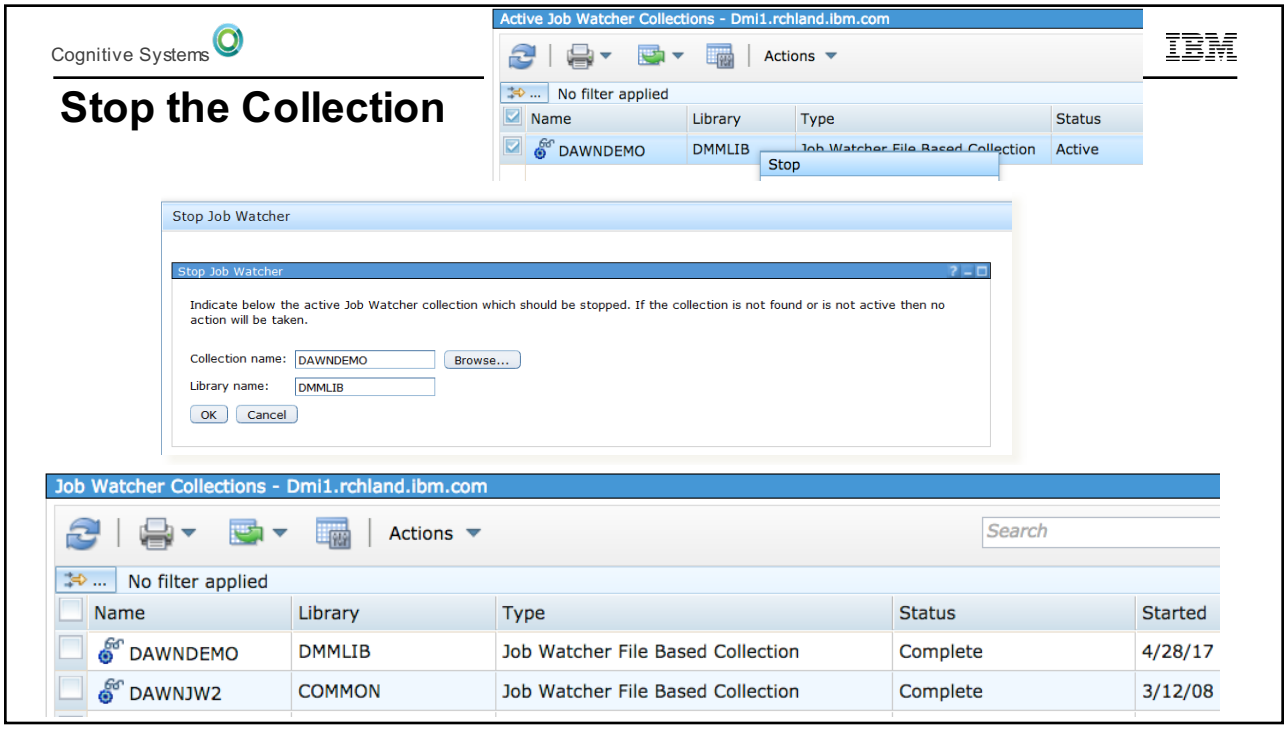
Job Watcher
Active Job Watcher Collections

Active Job Watcher Collections



Cognitive Systems

Stop the Collection



Cognitive Systems

IBM

How Do I Run Job Watcher with the Commands?

- Add Job Watcher Definition (**ADDJWDFN**) to define the collection
  - Identifies the performance data that is to be collected
- Remove Job Watcher Definition (**RMVJWDFN**) to remove a definition
  (Note: Job Watcher Definitions can only be displayed through the GUI)
- Start Job Watcher (**STRJW**) to start the collection
- End Job Watcher (**ENDJW**) to end the collection (optionally)

Cognitive Systems

IBM

Running Job Watcher Continuously

- If you have a problem but Job Watcher is not running, you will need to recreate the problem to capture Job Watcher data
- Many clients prefer to run Job Watcher 24x7
  - Ensures you have diagnostic data the first time the problem occurs
- Options:
  - QMGTOOLS has the ability to run Job Watcher
  - iDoctor monitors
  - Your own CL program

Cognitive Systems

IBM

Running Job Watcher Continuously - example

- Create the objects and the job watcher data in a your own library – e.g., CRTLIB JOBWATCH
- Create a dataarea to control how many collections you want to have saved. In the STRJW command, we use the RPLDTA parameter to automatically replace the older collections with the same name. This way we don't have to worry about deleting older collections - they are simply overwritten when the new one with the same name is started.
- CRTDTAARA DTAARA(JOBWATCH/JOBWATCH) TYPE(*DEC) LEN(15 5) VALUE(5)
- When the following program is called with the data area at 5 it will save 5 collections that have the names JWCOL1 .. to JWCOL5.
- *Note: The sample program has a hardcoded "5" in it for the reset number when the counter goes to 1 - if you want to have more than 5 collections saved, you need to have that 5 changed in the little CLP. A simple improvement to this program would be to save the value retrieved from the data area and reset to that value rather than a hard-coded number.
- In this example, we used the system supplied definition Q5SECSQL- you'll may want to set up your own definition and use it when you do the STRJW.
- Every hour a new JW collection will be started. The collection will run for 65 minutes (to have a 5 minute overlap for starting the next one and the previous one ending). This program will loop forever, keeping 5 job watcher collections, until the program is canceled.
- Submit the job to batch and let it run; end the batch job when you want to end your data collections. The last job watcher collection will remain active until it completes.

Cognitive Systems

IBM

Sample CL Program – Provided AS IS

```
PGM
             DCL        VAR(&DECNUMBER) TYPE(*DEC) LEN(15 5)

             DCL        VAR(&NUMBER) TYPE(*UINT) LEN(2)
             DCL        VAR(&COLNAME) TYPE(*CHAR) STG(*AUTO) LEN(10) +
                          VALUE(JWCOL)

ADDLIBLE JOBWATCH
MONMSG CPF9999

BEGIN:

  RTVDTAARA  DTAARA(JOBWATCH/JOBWATCH *ALL) +
               RTNVAR(&DECNUMBER)

  CHGVAR     VAR(&NUMBER) VALUE(&DECNUMBER)

  IF         COND(&DECNUMBER *EQ 1) THEN(CHGVAR +
               VAR(&DECNUMBER) VALUE(5))
  ELSE       CMD(CHGVAR VAR(&DECNUMBER) VALUE(&DECNUMBER +
               - 1))

  CHGDTAARA  DTAARA(JOBWATCH/JOBWATCH *ALL) +
               VALUE(&DECNUMBER)

  CHGVAR     VAR(%SST(&COLNAME 6 2)) VALUE(&NUMBER)

     STRJW      DFN(Q5SECSQL) COL(&COLNAME) LIB(JOBWATCH) +
                  RPLDTA(*YES) ENDCOL((*NBRSEC 3900))

DLYJOB 3600

GOTO BEGIN
ENDPGM
```

Cognitive Systems

IBM

Job Watcher Authority Requirements

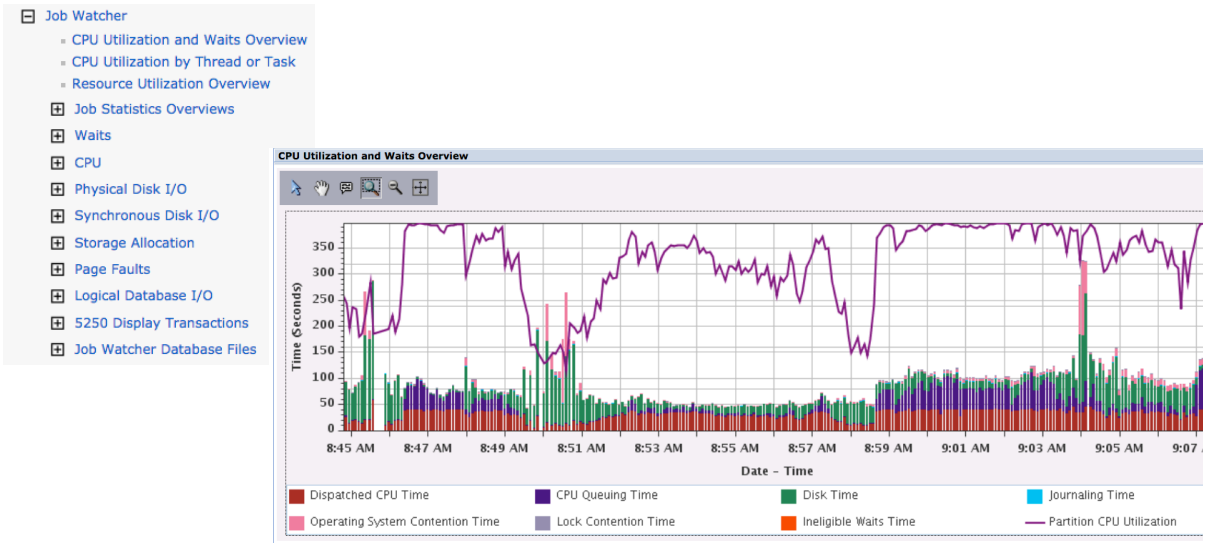
- **Commands**:
  - You must have service (***SERVICE**) special authority
    - Change User Profile to add *SERVICE authority to create Job Watcher Definitions or to Start Job Watcher
  - -**OR**- Be authorized to the Job Watcher function of the operating system
    - Change Function Usage (CHGFCNUSG) command, with a function ID of QIBM_SERVICE_JOB_WATCHER can be used to change the list of users that are allowed to use this command.
    - `CHGFCNUSG FCNID(QIBM_SERVICE_JOB_WATCHER) USER(<usrprofile>) USAGE(*ALLOWED)`
- **Definitions**:
  - Additional authority is needed to see the definitions for each as they are shipped with public authority *EXCLUDE. To see the definitions shipped in Job Watcher, users will need authority to the QAPYJWDFN file in QUSRSYS

- www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/IBM%20i%20Technology%20Updates/page/Authority
- https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/IBM%20i%20Technology%20Updates/page/Authorityadd more guidance on creating the JW definition

Cognitive Systems

IBM

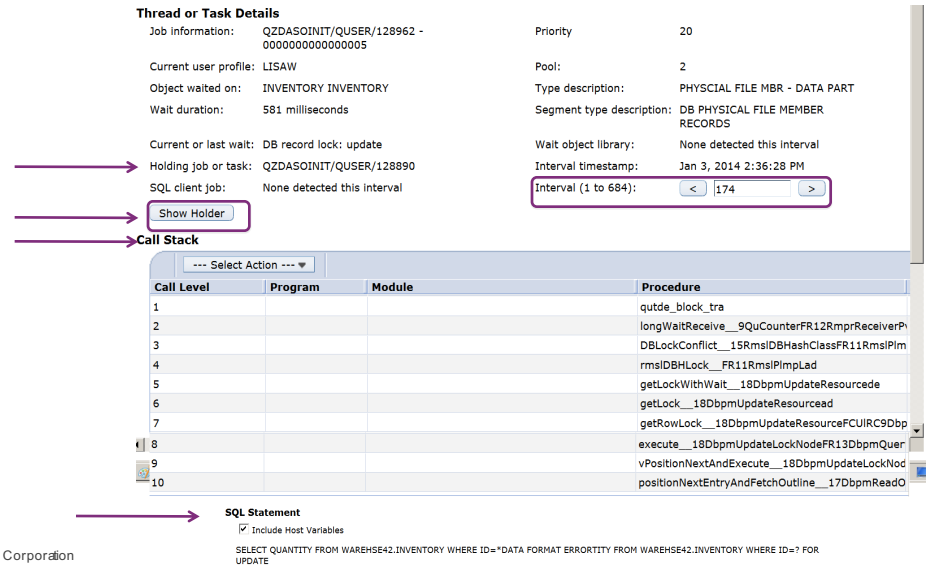
Job Watcher – PDI interface



Cognitive Systems

IBM

Job Watcher – Interval Details



IBM

Cognitive Systems

Performance Explorer

Cognitive Systems

IBM

Performance Explorer

- Performance Explorer helps identify the causes of performance problems that cannot be resolved using one of the other performance data collectors
  - Collects more detailed information about a specific application, program, or resource
- Performance Explorer is typically used for two main reasons:
  - Detailed performance trace data is needed to identify the performance problem
  - Analyzing the performance of applications
- Performance Data Investigator supports profile collections only
- iDoctor is required for advanced PEX Analysis

77

Cognitive Systems

IBM

Performance Explorer

- Performance Explorer is the most sophisticated IBM i performance tool
  - Can collect the details of every I/O operation, every task switch
  - Hundreds of events collected
  - Thus, most complex to use
  - More overhead
- Typically, problem has been scoped by other tools first
- Generally used by IBM performance analysis experts
  - Except…..

78

Cognitive Systems

IBM

Performance Explorer – "TPROF" usage

- "Trace-profile" is a fairly easy, and fairly "light-weight" PEX collection that can be useful to application developers, especially when trying to diagnose high CPU issues
  - Provides CPU usage at a program/module/procedure level
  - Make sure you have latest PEX PTFs applied
    http://www-01.ibm.com/support/docview.wss?uid=nas8N1012020
  - Can be run over subset of jobs

- Steps:
  1. Add a PEX definition:

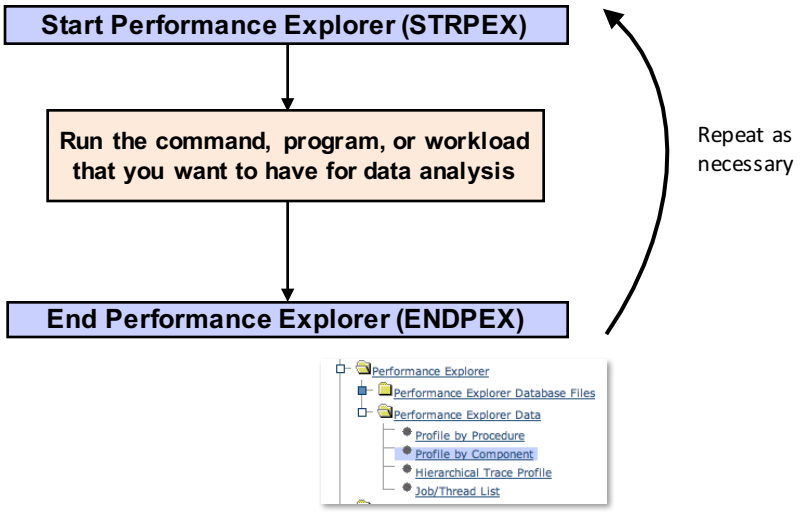
```
ADDPEXDFN DFN(TPROF)
    TYPE(*TRACE)
    JOB((*ALL *ALL))
    TASK(*ALL)
    MAXSTG(4000000)
    INTERVAL(1)
    TRCTYPE(*SLTEVT)
    SLTEVT(*YES)
    MCHINST(*NONE)
    BASEVT((*PMCO *NONE *FORMAT2))
```


Cognitive Systems

IBM

Performance Explorer – "TPROF" usage

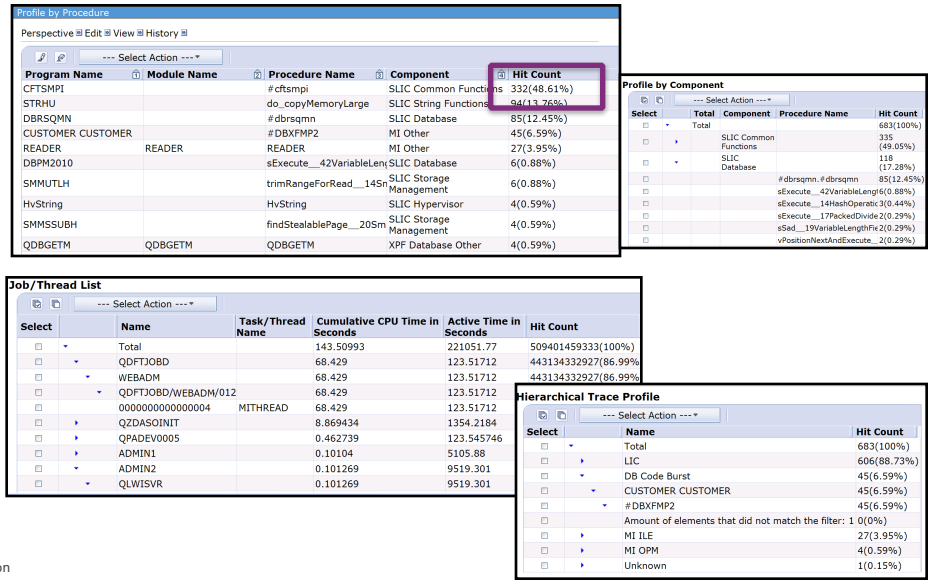
2. Collect data



Cognitive Systems

IBM

Performance Explorer TPROF reports – PDI



Profile by Procedure

Perspective Edit View History

--- Select Action ---

Program Name	Module Name	Procedure Name	Component	Hit Count
CFTSMPI		#cftsmpi	SLIC Common Functions	332(48.61%)
STRHU		do_copyMemoryLarge	SLIC String Functions	94(13.76%)
DBRSQMN		#dbrsqmn	SLIC Database	85(12.45%)
CUSTOMER CUSTOMER		#DBXFMP2	MI Other	45(6.59%)
READER	READER	READER	MI Other	27(3.95%)
DBPM2010		sExecute__42VariableLen	SLIC Database	6(0.88%)
SMMUTLH		trimRangeForRead__14Sn	SLIC Storage Management	6(0.88%)
HvString		HvString	SLIC Hypervisor	4(0.59%)
SMMSSUBH		findStealablePage__20Sm	SLIC Storage Management	4(0.59%)
QDBGETM	QDBGETM	QDBGETM	XPF Database Other	4(0.59%)

Profile by Component

--- Select Action ---

Select		Total	Component	Procedure Name	Hit Count
		Total			683(100%)
			SLIC Common Functions		335 (49.05%)
			SLIC Database		118 (17.28%)
				#dbrsqmn, #dbrsqmn	85(12.45%)
				sExecute__42VariableLengt	6(0.88%)
				sExecute__14HashOperatic	3(0.44%)
				sExecute__17PackedDivide	2(0.29%)
				sSad__19VariableLengthFie	2(0.29%)
				vPositionNextAndExecute_	2(0.29%)

Job/Thread List

--- Select Action ---

Select		Name	Task/Thread Name	Cumulative CPU Time in Seconds	Active Time in Seconds	Hit Count
		Total		143.50993	221051.77	509401459333(100%)
		QDFTJOBD		68.429	123.51712	443134332927(86.99%
		WEBADM		68.429	123.51712	443134332927(86.99%
		QDFTJOBD/WEBADM/012		68.429	123.51712	
		0000000000000004	MITHREAD	68.429	123.51712	
		QZDASOINIT		8.869434	1354.2184	
		QPADEV0005		0.462739	123.545746	
		ADMIN1		0.10104	5105.88	
		ADMIN2		0.101269	9519.301	
		QLWISVR		0.101269	9519.301	

Hierarchical Trace Profile

--- Select Action ---

Select	Name	Hit Count
	Total	683(100%)
	LIC	606(88.73%)
	DB Code Burst	45(6.59%)
	CUSTOMER CUSTOMER	45(6.59%)
	#DBXFMP2	45(6.59%)
	Amount of elements that did not match the filter: 1	0(0%)
	MI ILE	27(3.95%)
	MI OPM	4(0.59%)
	Unknown	1(0.15%)

IBM

Cognitive Systems

Examples

IBM

Cognitive Systems

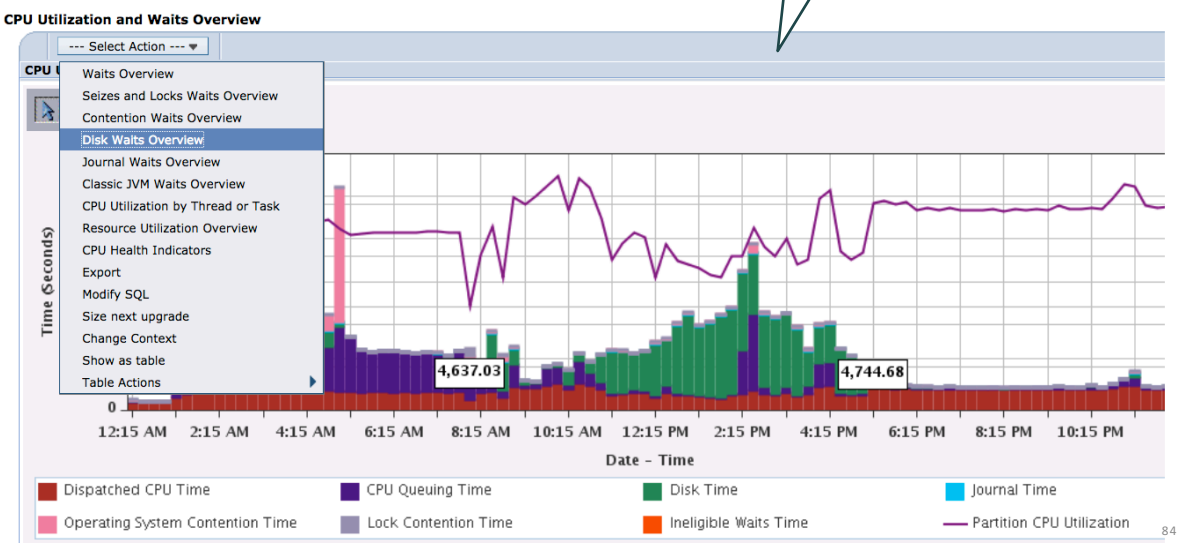
What is causing disk wait time?
Reviewing wait times with Collection Services

Cognitive Systems

IBM

Let's Look at the Disk Waits

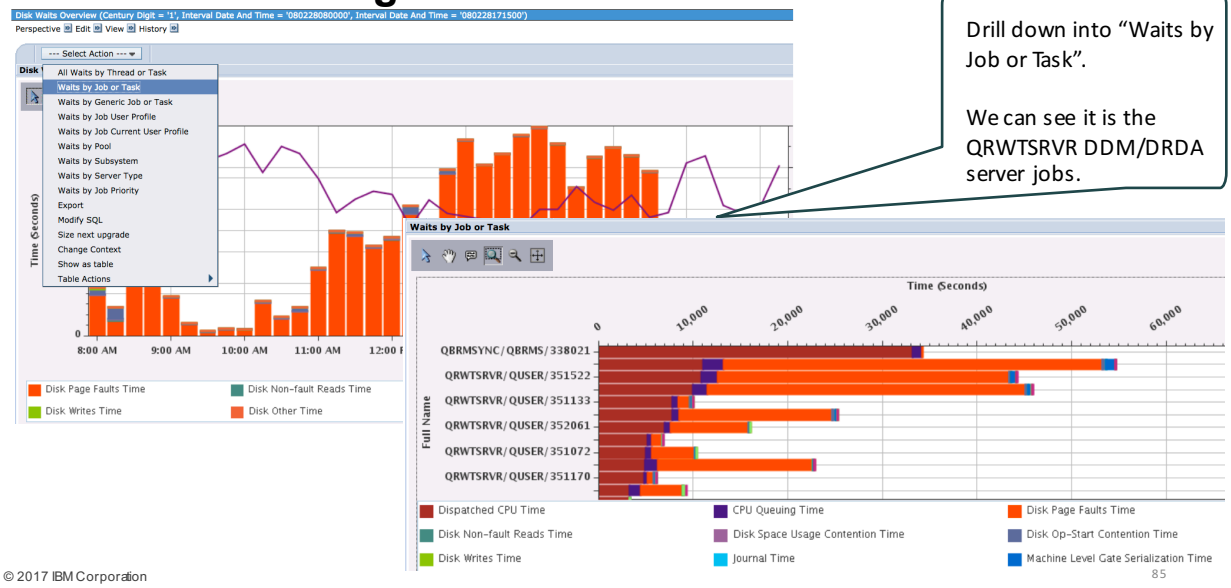
That mountain of green is Disk Time. What caused it?



Cognitive Systems

IBM

We see it's faulting…. let's find out who did it

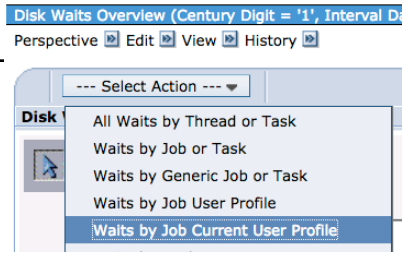


Drill down into "Waits by Job or Task".

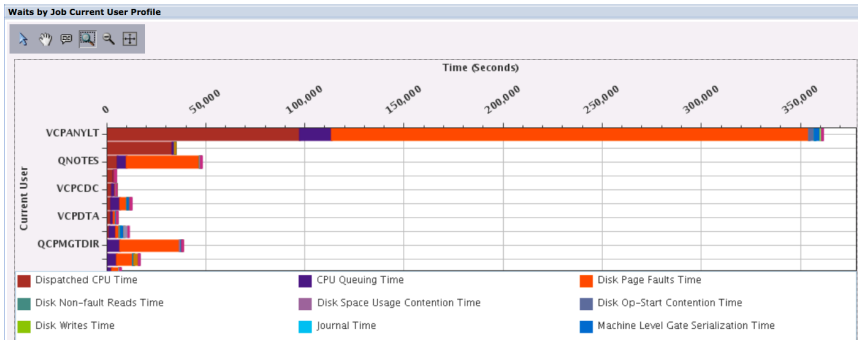
We can see it is the QRWTSRVR DDM/DRDA server jobs.

Cognitive Systems

Let's find out who the user is



- We now have several clues:
  - We know the jobs - QRWTSRVR - **DDM/DRDA server jobs**
  - We know the time - early afternoon
  - We know the user profile - **QUSER**
    - But QUSER isn't helpful. We need the job's **current user profile**
  - Waits by Job Current User Profile shows us **VCPANYLT** is the guilty party



IBM

Cognitive Systems

Viewing Waits with Job Watcher

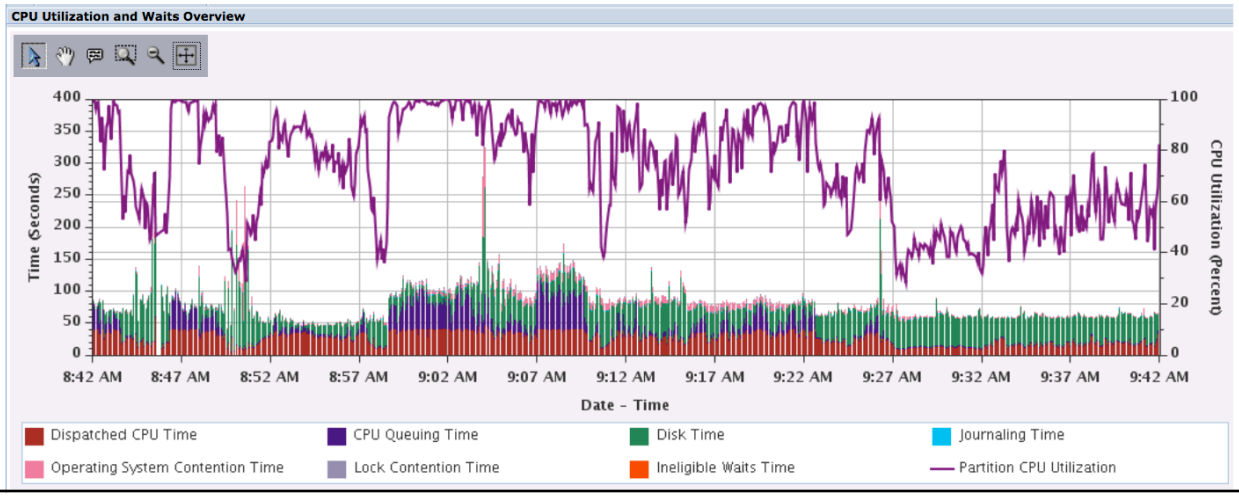
Example of Machine Gate Serialization

Cognitive Systems

Let's look at the entire collection

- Job Watcher
  - CPU Utilization and Waits Overview
  - CPU Utilization by Thread or Task
  - Resource Utilization Overview

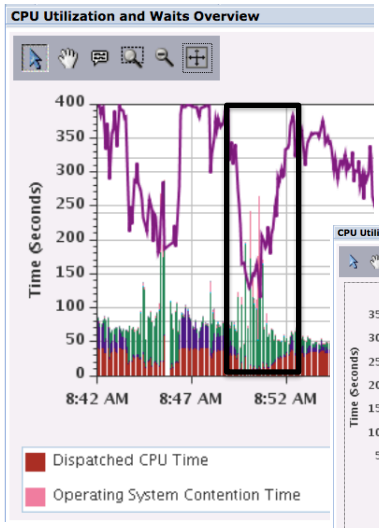
- CPU Utilization and waits gives an excellent overview of the entire partition
- There are a lot of interesting things to investigate.....



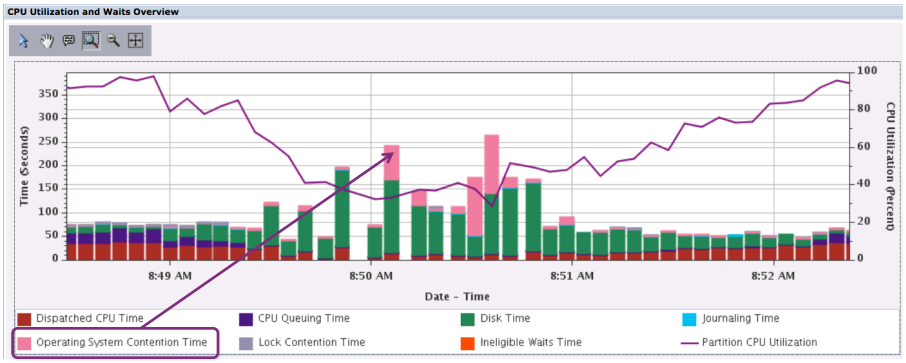
Cognitive Systems

IBM

Zoom into the time where we see a large drop in CPU Utilization



We can see operating system contention occurred during the time when the CPU Utilization dropped.

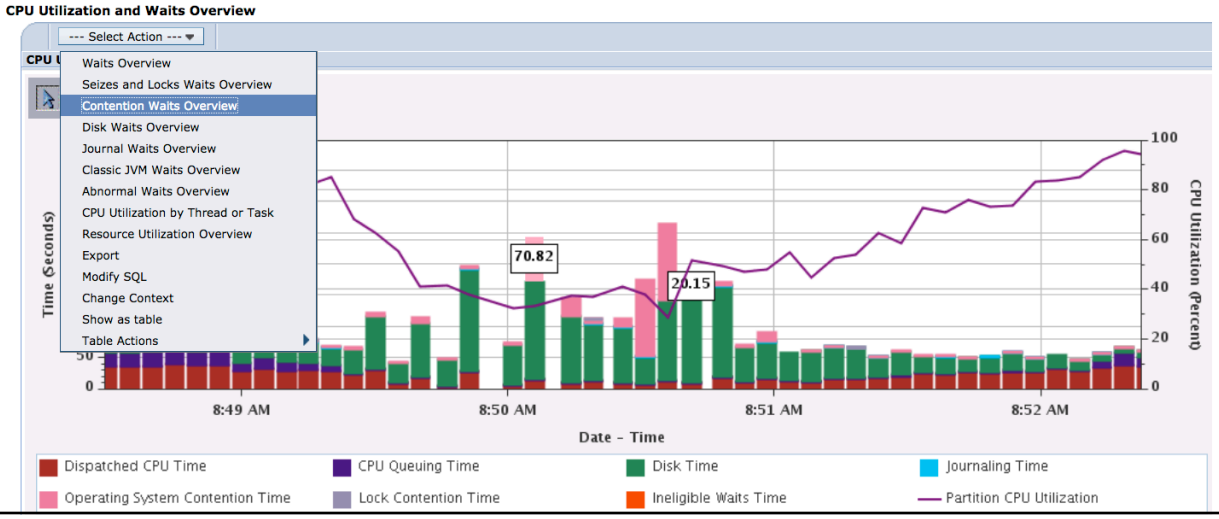


Cognitive Systems

IBM

Drill-down based upon what you see

Select the beginning and ending intervals to investigate and then drill into Contention Waits Overview
Be sure to select the metric you are investigating.

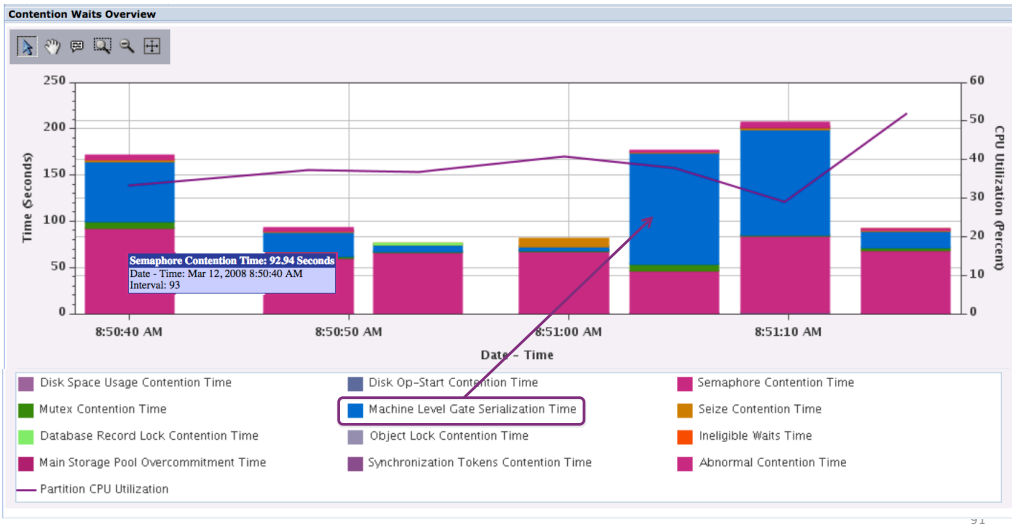


Cognitive Systems

IBM

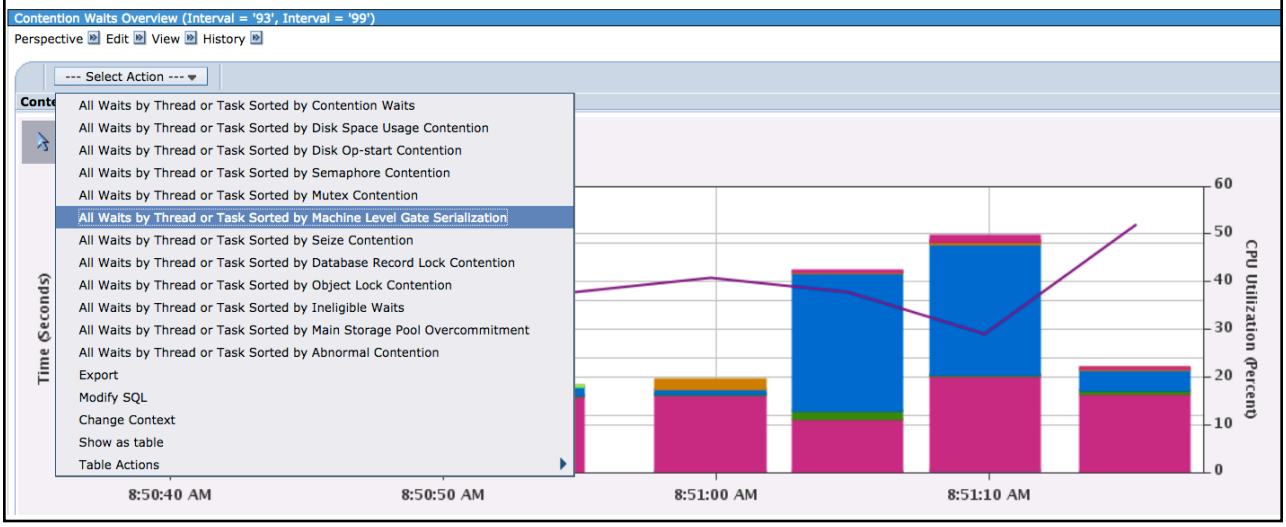
Machine level gate serialization is a major reason for the contention waits

The pink is semaphore contention (see the flyover). This is usually an uninteresting wait.

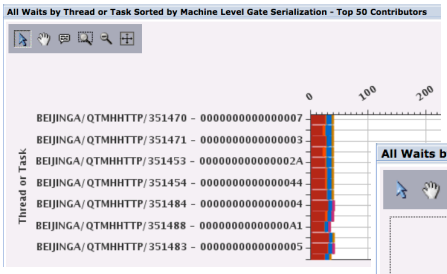


We want to see if we can figure out who might be causing the contention.
Drill into **All Waits by Thread or Task Sorted by Machine Level Gate Serialization** so we can see the jobs/threads/tasks that are all waiting.
We selected **Machine Level Gate Serialization** because that's the predominate "bad" wait our graph shows us.

Note: Drilling into **waits by thread or task** can take some time…. be patient.



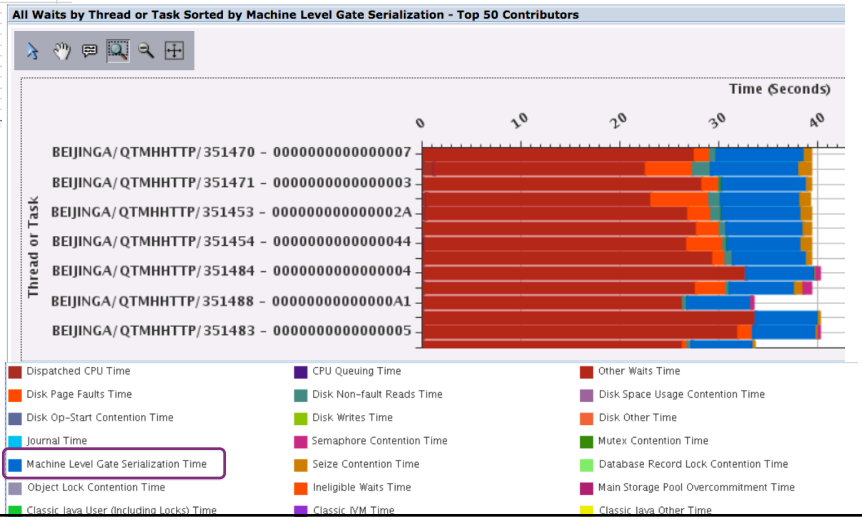
Zoom into see more detail



We can't see the machine level gate serialization details at first;

Zoom in and we can see it appear in many threads. This tells us many threads were waiting.

But why?

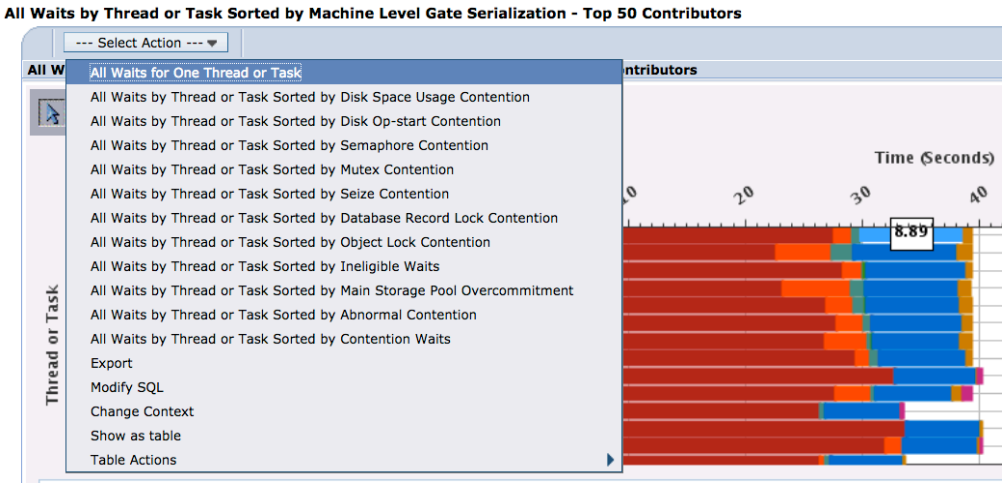


Cognitive Systems

IBM

Select a thread and look at the waits for that one thread

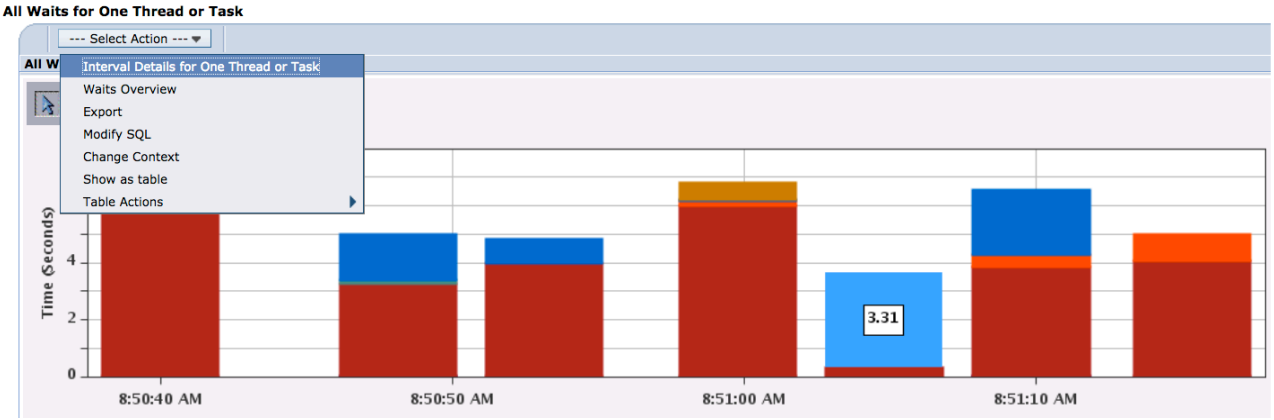
It may be necessary to drill down into interval details for several threads to find the one with the information we need...



Cognitive Systems

IBM

Select an interval
View Interval details for one thread or task



Cognitive Systems

IBM

And here is where we discover Job Watcher's power….

Interval Details for One Thread or Task (Interval = '97', Initial Thread Task Count = '319696')

Perspective Edit View

Thread or Task Details

Job information:	BEIJINGA/QTMHHTTP/351470 - 0000000000000007	Priority	25
Current user profile:	QTMHHTP1	Pool:	2
Object waited on:	QAUDJRN	Type description:	JOURNAL
Wait duration:	3,307 milliseconds	Segment type description:	JOURNAL RESERVED
Current or last wait:	Qu gate - high performance	SQL client job:	None detected this interval
Holding job or task:	QDBSRV02/QSYS/345313	Interval timestamp:	Mar 12, 2008 8:51:05 AM
Show Holder		Interval (1 to 710):	< 97 >

We can review the call stack to see how we got to this wait point.

This thread is waiting for the QAUDJRN journal at 8:51:05.

In the call stack you will see an entry that shows the job is creating an audit journal entry.

Note that access to the audit journal is serialized by a "gate". So why is this job blocked and waiting to create the audit record?

Call Stack

--- Select Action ---

Call Level	Program	Module	Procedure	Offset
1			qutde_block_trace	000000E4
2			slowLock__10QuGateCodeFQ2_2Qu8Lock	000003F0
3			#journal	0000066C
4			auditIt__19CfCreateAuditRecordFv	00000444
5			validateDescQueue__11LoDescEntryFRtP	00000390
6			recvDescriptors__12LoSocketUnixFR15Lo	000002B8
7			recv__19LoReceiveStreamUnixFR15LoSo	000006A0
8			recvmsg__8LoSocketFR15LoSocketMana	0000018C
9			recvmsg__FtP6msghdrT1PtP7timeval15L	00000680
10			recvMsgHandler__FP19LoSocketRecvMsg	00000448
11			LoSocketOp__FUtP13LoSocketOpHdr	00000254
12			socketop	000001DC

Total: 28

97

Cognitive Systems

IBM

We can easily go look at the thread that is holding the resource

The call stack shows a "Change Journal" program (QJOCHJN)

Show Holder

Interval Details for One Thread or Task (Interval = '97', Initial Thread Task Count = '1476')

Perspective Edit View

Thread or Task Details

Job information:	QDBSRV02/QSYS/345313 - 0000000000000001	Priority	16
Current user profile:	QSYS	Pool:	2
Object waited on:	QAUDJRN	Type description:	JOURNAL
Wait duration:	104 milliseconds	Segment type description:	JOURNAL SPACE IOCB
Current or last wait:	MAINSTORE/LOGICAL-DASD-IO: PAGE FAULT	SQL client job:	None detected this interval
Holding job or task:	None detected this interval	Interval timestamp:	Mar 12, 2008 8:51:06 AM
Show Holder		Interval (1 to 710):	< 97 >

Call Stack

--- Select Action ---

Call Level	Program	Module	Procedure
1			qutde_stackless_block
2			qu_dasd_fault_on_res_stack
3			#jomodjp
4			#cfmir
5			syscall_A_portal
6	QJOCHGJN		
7			cblabranch
8			aiuser_program_call_portal
9	QJORETRY		
10			cblabranch
11			aiuser_program_call_portal
12	QDBSERVE		

Total: 15

Since we know the object that was causing the contention was QAUDJRN, let's have a look….

Note the audit journal entries from the matching time period. Observe the job name…

NR is Next Receiver
PR is Previous Receiver

```
                    Display Journal Entries

Journal . . . . . :   QAUDJRN        Library . . . . . :   QSYS
Largest sequence number on this screen . . . . . : 00000000000088885894
Type options, press Enter.
  5=Display entire entry


Opt    Sequence  Code  Type  Object     Library     Job         Time
       88885883   T     GS                         BEIJINGA     8:51:02
       88885884   T     SK                         QSYSARB      8:51:02
       88885885   J     NR                         QDBSRV02     8:51:02
       88885886   J     PR                         QDBSRV02     8:51:06
       88885887   T     GS                         BEIJINGA     8:51:07
       88885888   T     GS                         BEIJINGA     8:51:07
       88885889   T     GS                         BEIJINGA     8:51:07
       88885890   T     SK                         QSYSARB      8:51:07
       88885891   T     GS                         BEIJINGA     8:51:07
       88885892   T     GS                         BEIJINGA     8:51:07
       88885893   T     GS                         BEIJINGA     8:51:07
       88885894   T     GS                         BEIJINGA     8:51:07
                                                                    More...
F3=Exit   F12=Cancel
```


Cognitive Systems

IBM

What next?

- We observed significant system-wide machine gate serialization contention
- With Job Watcher, we determined that at that point in time, the QAUDJRN was changing journal receivers
- Perhaps we should review our security audit settings
- In this example, it was a stress-test System Test partition with a heavy workload and extreme settings

IBM

Cognitive Systems

Viewing Waits with Job Watcher

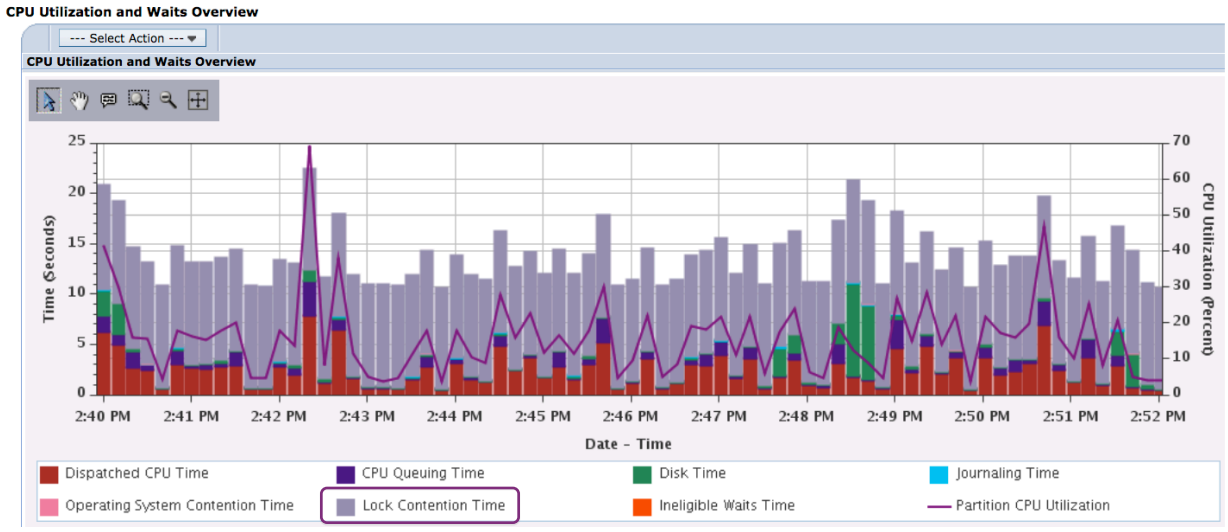
Example of Object Lock Contention

Cognitive Systems

IBM

Once again, we start with CPU Utilization and Waits Overview

Observe the gray … "Lock Contention Time"



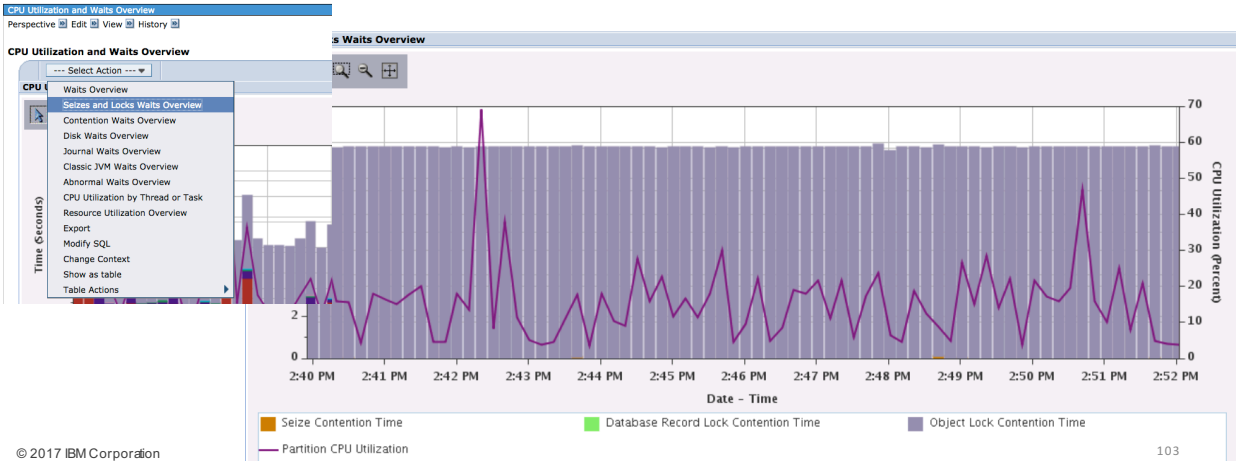
Cognitive Systems

IBM

Drill into Seizes and Locks Waits Overview

Tip: Drill-downs are named to drill into what you see

- This confirms we have some job holding a lock that another job is waiting on

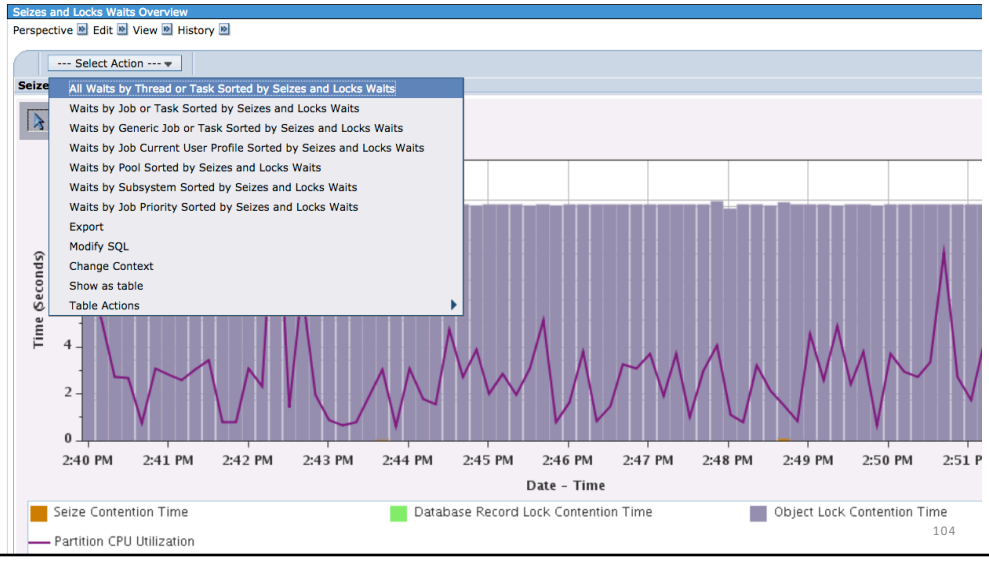


Cognitive Systems

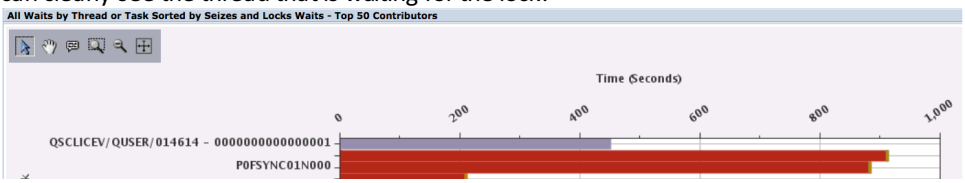
IBM

Seizes and Lock Waits Overview →All Waits by Thread or Task…

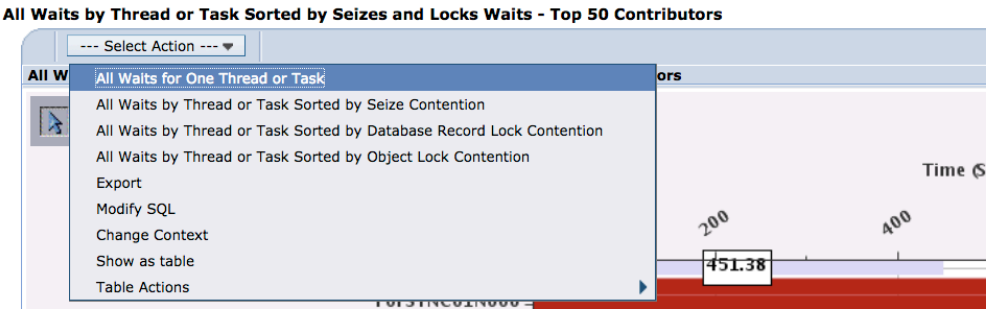
Drill down into **All Waits by Thread or Task Sorted by Seizes and Lock Waits**



We can clearly see the thread that is waiting for the lock.



Select that thread and drill down to **All Waits for One Thread or Task**



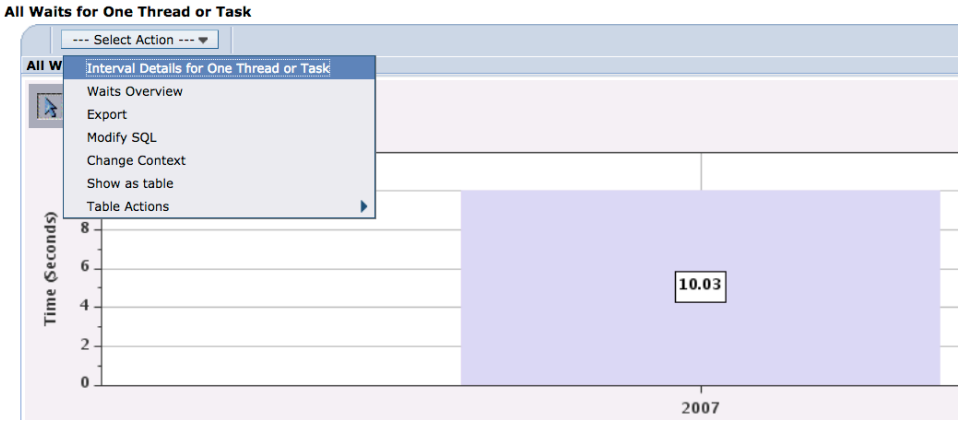
Cognitive Systems

IBM

We are now to the magic drill-down…

Select the interval for that job and drill down to

Interval Details for One Thread or Task



Cognitive Systems

IBM

Interval Details

Interval Details for One Thread or Task (Interval = '47', Initial Thread Task Count = '2545')

Perspective Edit View

Thread or Task Details

Job information:	QSCLICEV/QUSER/014614 - 0000000000000001	Priority	25
Current user profile:	QSYS	Pool:	2
Object waited on:	WATCHEVENTSPACE	Type description:	TEMPORARY - SPACE
Wait duration:	4,482 milliseconds	Segment type description:	BASE MI SYSTEM OBJECT
Current or last wait:		SQL client job:	None detected this interval
Holding job or task:	QZRCSRVS/QUSER/014097	Interval timestamp:	Dec 13, 2007 2:48:00 PM
Show Holder		Interval (1 to 91):	< 47 >

Call Stack

IBM

Cognitive Systems

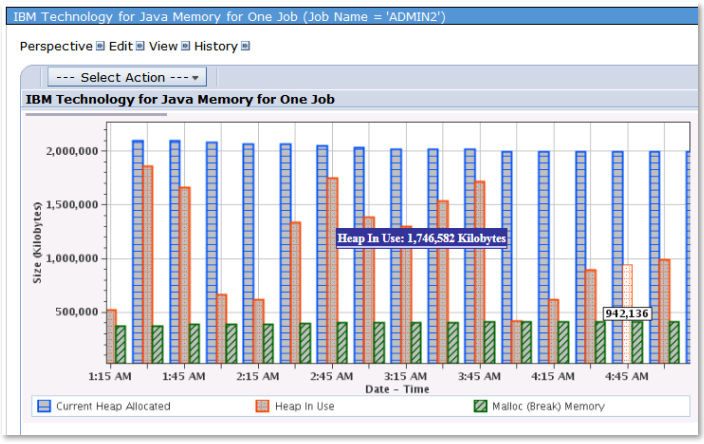
More PDI Examples

Cognitive Systems

IBM

Java Perspectives

Drilldown for one job -
Look at the heap and memory usage over time for one selected job.



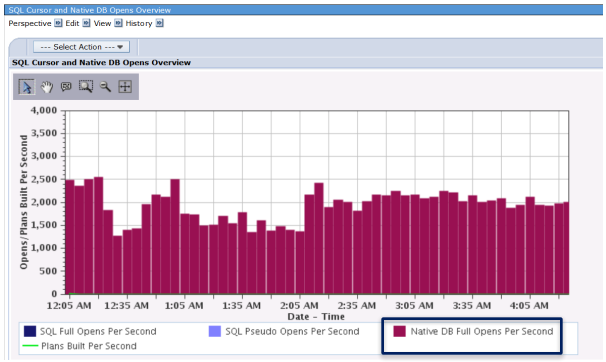
IBM Technology for Java Memory for One Job

Cognitive Systems

IBM

Database Full Opens

Full Opens are expensive resource-wise



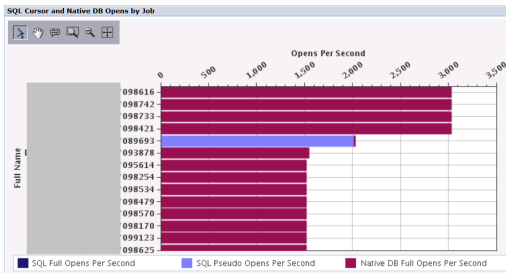
General recommendation is to keep Native Full Opens per second < 1000

Next, find jobs doing full opens…

Cognitive Systems

IBM

Database Full Opens



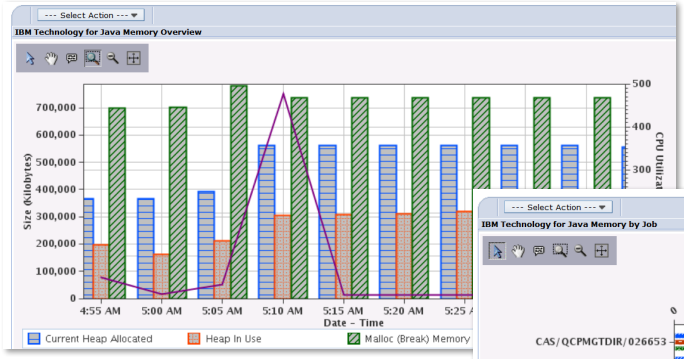
In an RPG program, full opens are caused by the use of SETON *LR instead of RETRN. Avoid LR if possible. Not setting on LR (in OPM programs) will keep the program in memory, keeps file open and pointers set, retains variable values, etc. Also, avoid *NEW for the ACTGRP

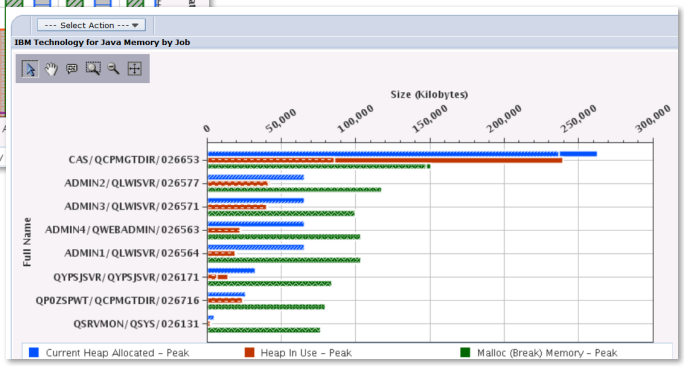
Shared file opens are far less expensive than full file opens. They consume less CPU, less storage and are faster than full opens. There are some implications of changing full file opens to shared file opens, but implementation of shared opens is typically easier to manage and implement than changing SETON LR to RETRN in RPG programs. The following links provide explanation, usage and considerations of using shared ODPs:

Sharing database files in the same job or activation group
http://pic.dhe.ibm.com/infocenter/iseries/v7r1m0/topic/dbp/rbafosfile.htm?resultof=%22%4f%44%50%22%20%22%6f%64%70%22%20
Open considerations for files shared in a job or an activation group
http://pic.dhe.ibm.com/infocenter/iseries/v7r1m0/topic/dbp/rbafoopench.htm
Input/output considerations for files shared in a job or an activation group
http://pic.dhe.ibm.com/infocenter/iseries/v7r1m0/topic/dbp/rbafoiocon.htm
Close considerations for files shared in a job or an activation group
http://pic.dhe.ibm.com/infocenter/iseries/v7r1m0/topic/dbp/rbafoclose.htm

Cognitive Systems

Java Perspectives in Collection Services





Find that job using a lot of heap…

Cognitive Systems

IBM

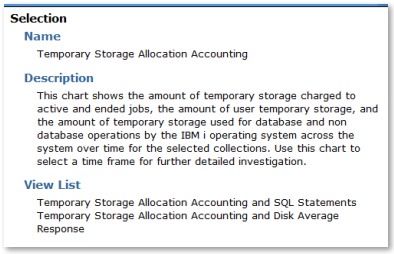
Temporary Storage Allocation/Deallocation perspectives

7.2

Storage Allocation Perspectives

Where is my temporary storage going?

Expand Collection Services

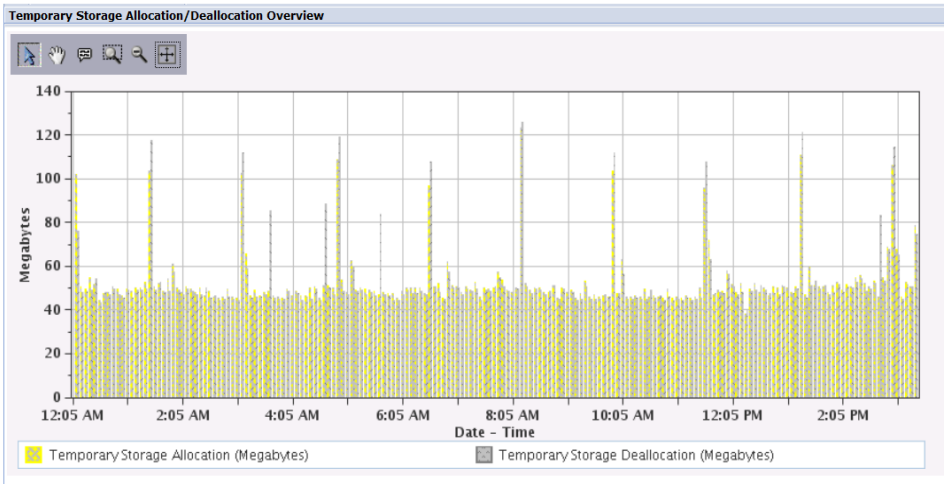


Cognitive Systems

IBM

Temporary Storage Allocation / Deallocation Overview

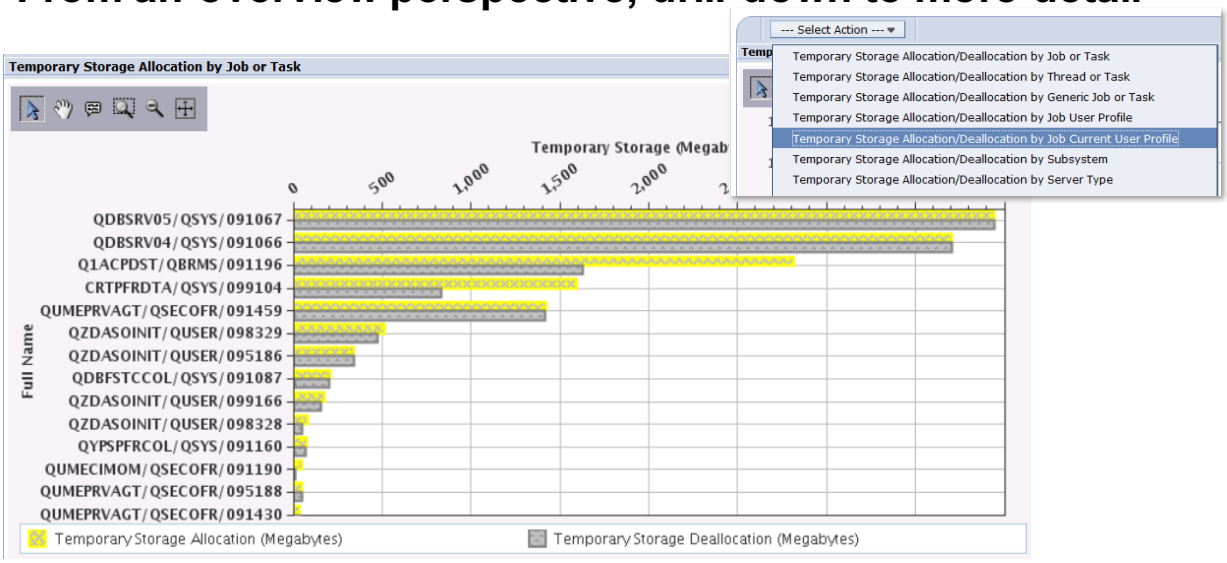
Generally, allocations and deallocations following a similar pattern



Cognitive Systems

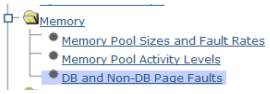
IBM

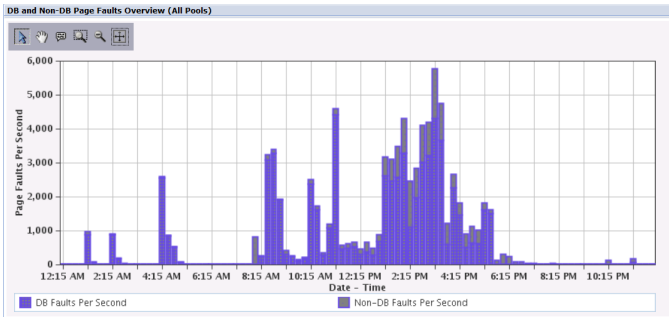
From an overview perspective, drill down to more detail

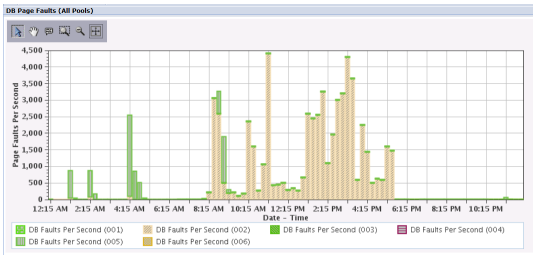


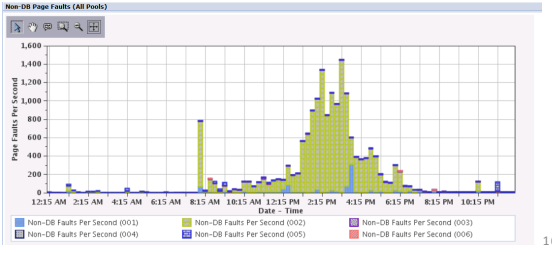
Cognitive Systems

What does the faulting look like when I was testing?









IBM

Cognitive Systems

iDoctor

Cognitive Systems

IBM

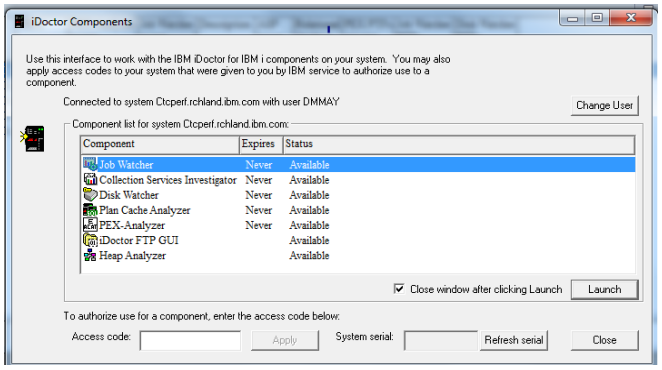
IBM iDoctor for i

- Product developed by the IBM Rochester Support Center for deep, detailed performance analysis
- Major components
  - Job Watcher
    - Job Watcher
    - Collection Services Investigator
    - Disk Watcher
    - Plan Cache Analyzer - graphical analysis of the system's SQL Plan Cache
  - PEX Analyzer
  - VIOS Investigator
- https://www-03.ibm.com/systems/services/labservices/iDoctor/

Cognitive Systems

IBM

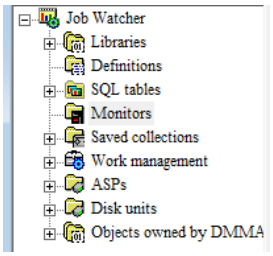
IBM iDoctor for i

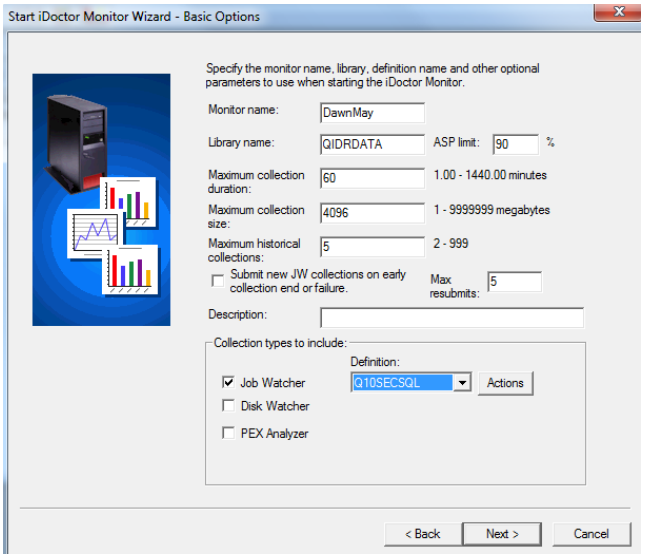


Cognitive Systems

IBM

iDoctor Monitors





Cognitive Systems

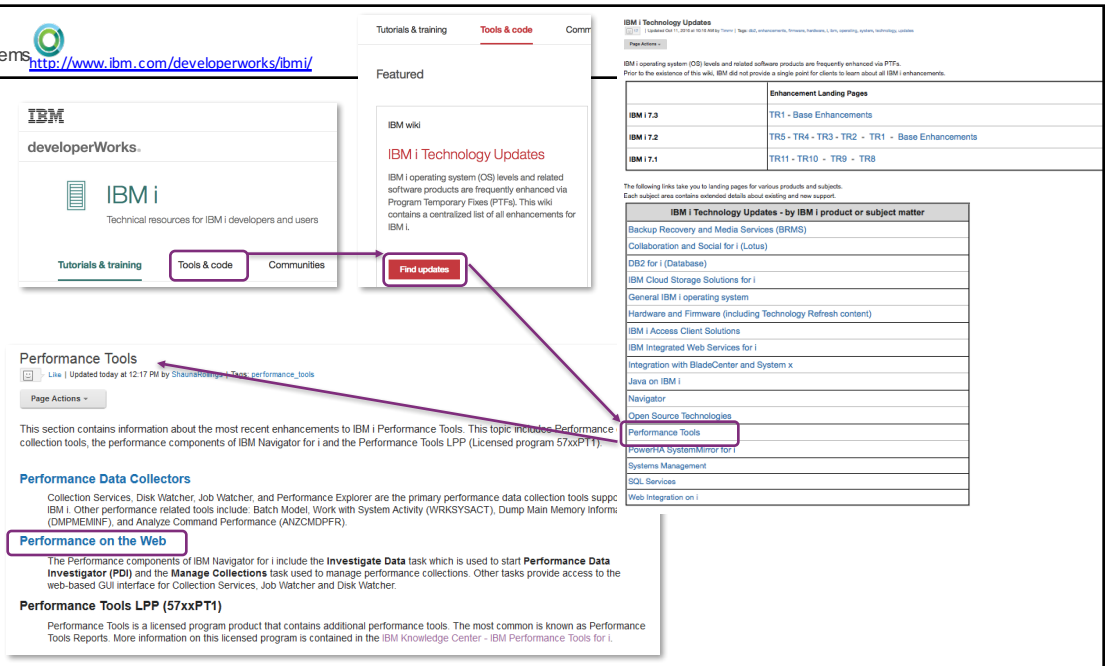
IBM

iDoctor versus Performance Data Investigator

- You have two graphical interfaces for performance data analysis…
  Which should you use? It depends….

Feature	iDoctor	PDI
Interface	Windows client	Browser
Wait Analysis	Yes	Yes
Collection Services	Yes	Yes
Job Watcher	Yes	Yes
Disk Watcher	Yes	Yes
Performance Explorer	Yes	Profile collections only
Database	Yes	Yes
Job Watcher Monitors	Yes	No
Customizable	Yes	Yes
User Defined graphs and queries	Yes	Yes
Update Frequency	Monthly	Twice Yearly
Support	Defect only	Standard SWMA
Chargeable	Yearly license	•Collection Services at no additional charge with i •Disk Watcher, Database, and Performance Explorer included with base PT1 product •Job Watcher is an additional option of PT1 and has an additional charge
Experimental Features	Yes (e.g., VIOS Investigator)	No
Multinational language support	No	Yes

Cognitive Systems http://www.ibm.com/developerworks/ibmi/



ithankyou

www.ibm.com/power/i

References

Cognitive Systems

IBM

IBM i Performance FAQ - a MUST read!

http://www.ibm.com/common/ssi/cgi-bin/ssialias?subtype=WH&infotype=SA&appname=STGE_PO_PO_USEN&htmlfid=POW03102USEN&attachment=POW03102USEN.PDF

IBM i on Power - Performance FAQ

April 3, 2017

Cognitive Systems

IBM

IBM i Web Sites with Performance Information

- IBM i Knowledge Center
  http://www.ibm.com/support/knowledgecenter/ssw_ibm_i/welcome
  - 7.1 - https://www.ibm.com/support/knowledgecenter/ssw_ibm_i_71/welcome.html
  - 7.2 - http://www-01.ibm.com/support/knowledgecenter/ssw_ibm_i_72/rzahg/ic-homepage.htm
  - 7.3 - https://www.ibm.com/support/knowledgecenter/ssw_ibm_i_73/rzahg/welcome.htm
- IBM i Performance Management
  http://www-03.ibm.com/systems/power/software/i/management/#tab2
- Performance Management for Power Systems
  http://www-03.ibm.com/systems/power/support/pm/index.html
- IBM Workload Estimator
  http://www.ibm.com/systems/support/tools/estimator
- iDoctor
  http://www-912.ibm.com/i_dir/idoctor.nsf
- Job Waits Documentation
  - http://www.ibm.com/support/knowledgecenter/en/ssw_ibm_i_73/rzahx/rzahxbasicwaitaccounting.htm
  - http://www.ibm.com/developerworks/ibmi/library/i-ibmi-wait-accounting/
  - http://public.dhe.ibm.com/services/us/igsc/idoctor/Job_Waits_White_Paper.pdf

Cognitive Systems

IBM

Understanding "Time Dispatched on a CPU"

- **Time dispatched on a CPU (Bucket 1)**
  - Thread or task has been assigned to a processor and is NOT waiting
  - Complicated by certain features
    - Hardware Multi Threading (HMT)
      - Allows multiple threads/tasks to be assigned to a single physical processor
      - Causes bucket 1 time to be greater than actual CPU time
    - Background assisting tasks
      - Promote their CPU usage back into the client job/thread
      - Causes client thread's bucket 1 time to be smaller than measured CPU time
    - LPAR shared/partial processors
      - Bucket 1 records time dispatched to the virtual processor
      - Bucket 1 time may be greater than CPU time because it may include time the thread/task is waiting for the physical processor behind the virtual processor

Bucket 1 does NOT equal CPU Time

Cognitive Systems

IBM

Understanding "CPU Queuing"

- **CPU Queuing (Bucket 2)**
  - Thread or task has been assigned to a processor and is waiting for the CPU to become available
    - Too much work on the partition causing threads to need to wait for the processors
    - Spiky workloads
    - Workload Groups
    - Shared processors

      Latency due to hypervisor sharing the physical processors among multiple partitions

Cognitive Systems

CPU Queueing … technical details

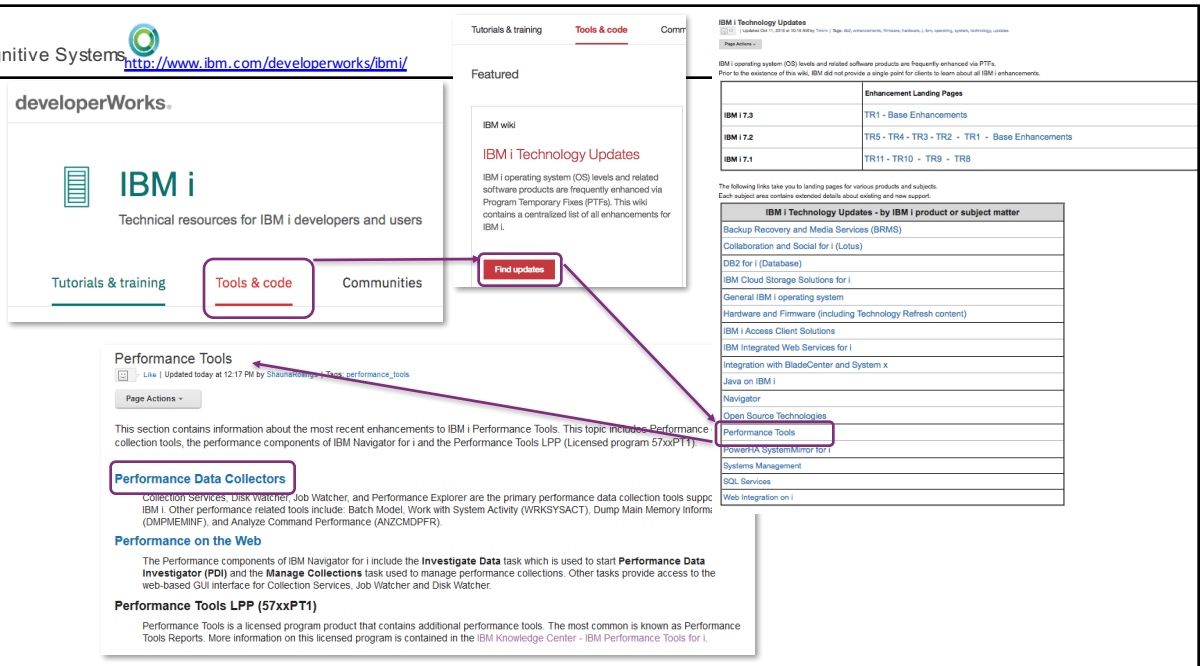
- A spikey workload.
  I/O completing in batches can cause this but so can software design. For instance, there are some seize-lock scenarios that unblock herds of tasks. A task switch trace can be used to instrument how the workload is behaving. This is a common reason.

- Shared processors.
  When there are idle processors, tasks are enqueued directly to a special per-processor queue until they overflow to a shared queue. On a dedicated processor system, the time it takes for the wait-state task to dequeue the task from this special queue and dispatch it is very low, but non-zero. The latency can be slightly higher when donation is used. In a SPLPAR, the latency can vary depending on outside factors, namely shared pool utilization, because the hypervisor is sharing the physical processors among multiple partitions. The separate components of VCPU latency is instrumented in Collection Services as of 7.1.

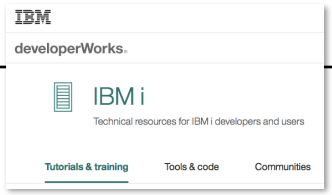
  Nodal dispatching.
  In a multi-node system, load imbalance could cause tasks to be queued in some nodes while other nodes are idle. This is generally not a problem with the default nodal affinity scheduling policy. However, a QTHDRSCAFN value set to *HIGH instead of the default *NORMAL can cause this problem. Node imbalance can be instrumented in 6.1 and later.

- Workload capping.
  The workload group can be over-committed even though the system is under-committed. There is separate instrumentation of workload group dispatch latency in Collection Services.

Cognitive Systems http://www.ibm.com/developerworks/ibmi/



IBM i Performance on developerWorks



- developerWorks
  http://www.ibm.com/developerworks/ibmi/
- Performance Tools
  https://www.ibm.com/developerworks/mydeveloperworks/wikis/home?lang=en#/wiki/IBM i Technology Updates/page/Performance Tools
- IBM i Performance Data Investigator
  http://www.ibm.com/developerworks/ibmi/library/i-pdi/index.html
- IBM i Performance Data Investigator – Edit Perspectives
  http://www.ibm.com/developerworks/ibmi/library/i-pdiedit/index.html
- IBM i Wait Accounting
  http://www.ibm.com/developerworks/ibmi/library/i-ibmi-wait-accounting/
- How to use the Batch Model performance tool
  https://www.ibm.com/developerworks/ibmi/library/i-how-to-use-the-batch-model-performance-tool/
- A new way to analyze historical performance data on IBM i
  https://www.ibm.com/developerworks/ibmi/library/i-new-way-to-analyze-historical-performance-data-trs/

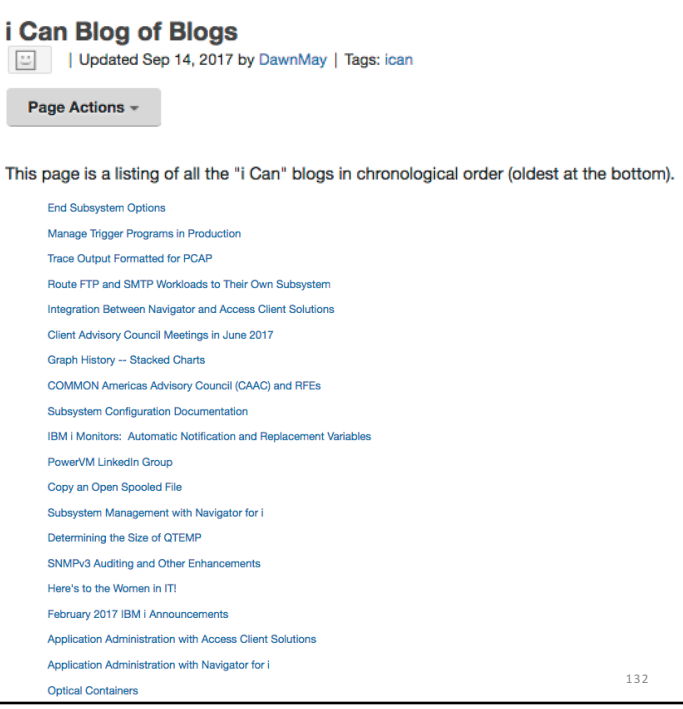
Cognitive Systems

i Can Technical Tips for i

i Can
http://www.ibmsystemsmag.com/Blogs/i-Can/

For a simple list of all blogs on one page:

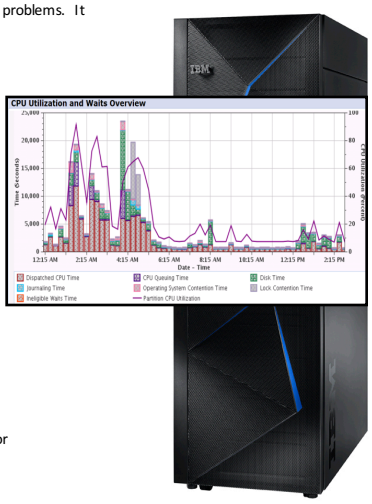
"i Can" Blog of Blogs
https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Power Systems/page/i Can Blog of Blogs



IBM i Performance Analysis Workshop

Learn the science and art of performance analysis, methodology and problem solving

Managing and analyzing the data can be quite complex. During this workshop, the IBM Systems Lab Services IBM i team will share useful techniques for analyzing performance data on key IBM i resources, and will cover strategies for solving performance problems. It will aid in building a future foundation of performance methodology you can apply in your environment.



Overview:

- Topics covered include:
  - Key performance analysis concepts
  - Performance tools
  - Performance data collectors (Collection Services, Job Watcher, Disk Watcher, and Performance Explorer)
  - Wait accounting
- Core methodology and analysis of:
  - Locks
  - Memory
  - I/O subsystem
  - CPU
- Concept reinforcement through case studies and lab exercises
- Discussions on theory, problem solving, prevention and best practices

Workshop details:

- Intermediate IBM i skill level
- 3-4 day workshop, public or private (on-site)
  - For public workshop availability and enrollment: IBM i Performance Analysis Workshop
  - For additional information, including private workshops, please contact Eric Barsness at ericbar@us.ibm.com or Stacy Benfield at stacylb@us.ibm.com, members of Systems Lab Services Performance Optimization team.

IBM Systems Lab Services Power Systems Delivery Practice - ibm.com/systems/services/labservices - ibmsls@us.ibm.com

Cognitive Systems

IBM

IBM i Performance and Optimization Services

The IBM i Performance and Optimization team specializes in resolving a wide variety of performance problems. Our team of experts can help you tune your partition and applications, including:

- Reducing batch processing times
- Resolving SQL query and native IO performance problems
- Tuning RPG, COBOL, C, and Java (including WebSphere Application Server) programs
- Removing bottlenecks, resolving intermittent issues
- Resolving memory leaks, temporary storage growth problems, etc.
- Tuning memory pools, disk subsystems, system values, and LPAR settings for best performance
- Optimizing Solid State Drive (SSD) performance
- Tuning client interfaces such as ODBC, JDBC, .Net and more

Skills transfer and training for performance tools and analysis also available!

Contact Eric Barsness at ericbar@us.ibm.com for more details.

Cognitive Systems

IBM

Performance and Scalability Services

- The IBM i Performance and Scalability Services Center can provide facilities and hardware IN ROCHESTER to assist you in testing hardware or software changes
  - "Traditional" benchmarks
  - Release-to-release upgrades
  - Assess and tune application and database performance
  - Stress test your system
  - Determine impact of application changes
  - Proofs of Concept (e.g. HA alternatives; SSD analysis, external storage, etc.)
  - Evaluate application scalability
  - Capacity planning
- … all with the availability of Lab Services IBM i experts and development personnel
- To learn more about Power offerings:
  https://www.ibm.com/it-infrastructure/services/lab-services/power

Cognitive Systems

IBM

Special notices

Cognitive Systems

IBM

Special notices (cont.)